# Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder Boulder, CO 80309-0430

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Department of Computer Science University of Colorado at Boulder 430 UCB Boulder, CO 80309-0430 Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

)100

0101

# Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder Boulder, CO 80309-0430

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Department of Computer Science University of Colorado at Boulder 430 UCB Boulder, CO 80309-0430 Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

)100

0101

## **Signatures**

### **Tom Lookabaugh**

University of Colorado at Boulder Department of Computer Science 430 UCB Boulder, CO 80309

#### **Bruce Sanders**

University of Colorado at Boulder Department of Computer Science 430 UCB Boulder, CO 80309

## Acknowledgements

We would like to thank a number people for their contributions and aid throughout the course of this project. The first is Dr. Tom Lookabaugh, who sponsored our project. Dr. Lookabaugh took the time to teach us about selective encryption and the JPEG compression. He has guided this project through every phase and we hope our final product satisfies his expectations. We would also like to thank Dr. Bruce Sanders for keeping the project on track, his invaluable input, and for believing in the success of Team ISE. Third, we thank Martin Cochran our TA for the Senior Project. He was required to read every one of our documents (multiple times) and provided critical insight into each. We thank David Keaton, who aided us in the lab and with the insides of the Linux systems. Then we would like to thank Cecilia M. Girz who aided us in proofreading our research paper. Lastly, we could not have made it this far without Jamie Griffith. Thank you for your wonderful support.

## Contents

- 1. Project Proposal
- 2. Initial Requirements
- 3. System Architecture
- 4. Overview Presentation
- 5. Requirements Specification
- 6. Design Specification
- 7. Design Presentation
- 8. Test Plan
- 9. ISE Man Pages
- 10. ISE Reference
- 11. Manipulator Tutorial
- 12. Manipulator Reference
- **13. Final Demo Presentation**
- 14. Developer's Reference
- 15. Research Paper
- 16. Source Code

# Project Proposal

# Team ISE Image Selective Encryption

#### Project Proposal September 2003

## Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

100

10

### **Project Proposal**

A constant amount of traffic flows between computers connected to the Internet. A large volume of information may take a long time traveling from destination to destination. The resulting speed reduction makes it desirable to compress the file as much as possible in order to send the smallest amount of data. Compression of data has allowed for the high-speed data transfers that have made Internet communication and business very workable.

In addition to sending the smallest amount of information possible, users also attempt to maintain a certain level of security upon their information. Due to the fact that common encryption methods generally manipulate an entire file, most encryption algorithms tend to make the transfer of information more costly in terms of time and bandwidth. Thus, users pay a price for security relative to their desired level of security. One possible solution would be a system of encryption that works cooperatively with the standard compression schemes. *Selective Encryption* of only a small percentage of the file's bits will facilitate this solution. Because most encryption schemes will make the file larger, selective encryption seeks only to encrypt portions of the file that will make it unusable. In other words, if a user does not have the proper decryption device, the file should not be usable. Selective encryption will minimize the necessary increase in file size due to encryption while maintaining a maximum level of uselessness, or damage, to the product.

An image could be encrypted with any of the sufficiently secure encryption algorithms available to the open source community, but this will usually result in a dramatic increase in file size that will severely increase transfer time over the Internet. However, selecting key parts of a file for encryption and only encoding those bits can actually render an image unusable. The initial statistical analysis done by the team will consist of specifically breaking down the standard JPEG compression scheme into its usable parts and evaluate which of these, if encrypted, will cause a potential user to pay for rights to the image or force subscription to the provider service.

Team ISE (Image Selective Encryption) will deliver a package for selectively encrypting JPEG (Joint Photographic Experts Group) still image files. The package will provide the tools necessary to encrypt the critical information of a JPEG file in cooperation with existing standard compression tools. This package will handle JPEG files in such a way that only a small percentage of the total file will be encrypted. Selective Encryption security will not extend to the level of military secrecy, but rather a level that would deter all but brute force attacks, allowing users to securely protect private JPEG images.

An additional aspect of the encryption analysis will be the determination of the specific targets in the file for encryption. For example in an MPEG file there are headers that contain a small portion of the overall number of bits but which are extremely vital to the reproduction of the movie by the user. So, if certain headers were to be encrypted the percentage of the file being manipulated would be less than ten percent of the total number of bits in the file. Although only a small portion will be encrypted, the resulting

damage experienced by an unauthorized user would be sufficient to cause the user to pay for the decryption package. However, there are other targets that, while they can be encrypted and will do sufficient damage, can be guessed by an attacker. The attacker could, with some degree off effort, render the file useful without use of the decryption software. For example, if the frame rate of an MPEG file was encrypted, an attacker could try all three of most common frame rates and one of these is certain to produce the correct rate for the particular video. In the case of JPEG Selective Encryption, Team ISE will have to balance the targets for encryption against ease of simple attacks.

A permanent website will be constructed by the team to make the software package available to anyone interested in the software process. As it is vital to the world of cryptography to let the community view the approach, the first form of the working prototype will be made available on the website. From this, feedback can be received not only from the team itself, but also from the cryptography community at large.

So, following the guidelines of the ongoing MPEG research (also being guided by the sponsor), the team will study the JPEG process and earlier attempts at encryption. With the sponsor's assistance, Team ISE will devise a workable approach to handling individual JPEG images following the concept of selective encryption.

It is possible that the team will complete the JPEG process early enough in the year that they will able to apply the same approach to other types of compressed files (text, audio, etc.) However, this initial specifications document applies only to the envisioned JPEG project.

# Initial Requirements

# Team ISE Image Selective Encryption

#### Initial Requirements September 19th, 2003

## Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Associate Professor of Computer Science Shinya Daigaku Geoffery Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

### **Project Proposal**

A constant amount of traffic flows between computers connected to the Internet. A large volume of information may take a long time traveling from destination to destination. The resulting speed reduction makes it desirable to compress the file as much as possible in order to send the smallest amount of data. Compression of data has allowed for the high-speed data transfers that have made Internet communication and business very workable.

In addition to sending the smallest amount of information possible, users also attempt to maintain a certain level of security upon their information. Due to the fact that common encryption methods generally manipulate an entire file, most encryption algorithms tend to make the transfer of information more costly in terms of time and bandwidth. Thus, users pay a price for security relative to their desired level of security. One possible solution would be a system of encryption that works cooperatively with the standard compression schemes – *Selective Encryption* of only a small percentage of the file's bits. Because most encryption schemes will make the file larger, selective encryption seeks only to encrypt portions of the file that will make it unusable. In other words, if a user does not have the proper decryption device, the file should not be usable. Selective encryption will seek to balance the necessary increase in file size, or bandwidth, due to encryption while maintaining a maximum level of uselessness, or damage, to the product.

An image could be encrypted with any of the sufficiently secure encryption algorithms available to the open source community, but this will usually result in a dramatic increase in file size that will prohibit transfer over the Internet. However, selecting key parts of a file for encryption and only encoding those bits can actually render an image unusable. The initial statistical analysis done by the team will consist of specifically breaking down the standard JPEG compression scheme into its usable parts, and evaluating which of these, if encrypted, will cause a potential user to pay for or subscribe to the decryption service.

Team ISE (Image Selective Encryption) will deliver a package for selectively encrypting JPEG still image files. The package will provide the tools necessary to encrypt the critical information of a JPEG file in cooperation with existing standard compression tools. This package will handle JPEG files in such a way that only a small percentage of the total file will be encrypted. The level of encryption will not reach to the height of military secrecy, but rather a level that would thwart most simple attacks while causing potential users to pay for viewing the image.

An additional aspect of the encryption analysis will be the determination of the specific targets in the file for encryption. For example in an MPEG file there are headers that contain a small portion of the overall number of bits but which are extremely vital to the reproduction of the movie by the user. So, if certain headers were to be encrypted the

percentage of the file being manipulated would be less than 10% of the total number of bits in the file. Although only a small portion will be encrypted, the resulting damage experienced by an unauthorized user would be sufficient to cause the user to pay for the decryption package. However, there are other targets that, while they can be encrypted and will do sufficient damage, can be guessed at by an attacker. The attacker could, with some degree off effort, render the file useful without use of the decryption software. For example if one encrypted the frame rate of an MPEG file, an attacker could just guess at the 3 most common frame rates, and one is certain to produce a correct copy of the video. Again, Team ISE will have to balance the targets for encryption against ease of simple attacks.

A permanent website will be constructed by the team to make the software package available to anyone interested in the software process. As it is vital to the world of cryptography to let the community view the approach, the first form of the working prototype will be made available on the website. From this, feedback can be received not only from the team itself, but also from the cryptography community at large.

So, following the guidelines of the ongoing MPEG research (also being guided by the Sponsor), the team will study the JPEG process and earlier attempts at encryption. With the Sponsor's assistance, Team ISE will devise a workable approach to handling individual JPEG images following the concept of selective encryption.

It is possible that the team will complete the JPEG process early enough in the year that they will able to apply the same approach to other types of compressed files (text, audio, etc.) However, this initial specifications document applies only to the envisioned JPEG project.

#### **Table of Contents**

#### **1. INTRODUCTION**

#### 2. RESEARCH PATH 2.1. Research and Analysis Requirements

#### **3. REQUIREMENTS**

#### **3.1. Supporting Environment**

- 3.1.1. Software
- 3.1.2. Hardware

#### **3.2. Functional Requirements**

- **3.2.1. Required Operations**
- **3.2.2. Interface to Generator**
- 3.2.3. Control of Software Event Collection

#### **3.3. Documentation and Release Requirements**

- 3.3.1. Documentation Requirements
- **3.3.2. Release Requirements**

#### 4. SUMMARY

## **1. INTRODUCTION**

The goal of selective encryption is to minimize the amount of encryption applied to a file while maximizing the damage done to the image being viewed by a user not in possession of the authorized decryption package. Complete encryption is not a requirement of the process, nor is rendering the file to useless to the level of complete military secrecy. It is acceptable for an attacker to be able to view portions of the file; however the file should be distorted enough that an attacker would not wish to use the encrypted file but would rather purchase or subscribe to the decryption method for access to the original files.

Multimedia files prove to be a good subject for selective encryption. This is due to the fact that the multimedia files tend to be very large and their compression algorithms concentrate critical information in small portions of this bit stream. If the critical information is encrypted, the remaining information becomes useless to those without the proper decoder. There are many types of compression algorithms that fit this description. Examples of such are MPEG 1, 2 and 4 video, AAC audio, G.723 and G.729 video, and JPEG and JPEG2000 image.

The focus of this project is to research and develop an algorithm for selective encryption of a standard baseline compressed JPEG image file. This process must encrypt a file in such a way that the amount of the file being encrypted is relatively small, yet the damage done to the file is on a scale that would render the file useless without a proper decryption device. This process will be delivered in a package that will include an encrypter for JPEG files and a decrypter that will reverse the operation. This package will be made available in a fully open source form on the website that will be constructed by the team.

The website is to be constructed on a server being purchased by the Sponsor in an environment that will match the other computers in the working lab. The team will acquire a fixed IP address from the proper University authorities and will set up a simple website capable of informing viewers about the possibilities of the technology of Selective Encryption and to provide them with a package they can download and test. The site will provide links to important information and will remain up permanently even once the project is complete.

The envisioned software package will accomplish a seemingly simple result while being extremely effective and usable to the appropriate users. Below is a flow chart showing the general picture of the package's operations. (Figure 1.1.)



Figure 1.1: Conceptual Overview of ISE Software

The ISE website displayed in the flow chart will be used to distribute the ISE software and will also contain information on the product as well as the research behind it.

The list of requirements for ISE follows. As there is a degree of research that must be done by the team under the Sponsor's guidance and supervision, the general path of the research is given as a precursor to the actual final product requirements. Further, as this research will to some degree determine the final necessary requirements, this document will serve as a starting point for the project, but will be refined later.

## 2. RESEARCH PATH

#### 2.1. Research and Analysis Requirements

The research and analysis will be the initial part of this project. The final product of this process is essentially a completely determined approach.

- Proportional Analysis of a large quantity of JPEG images to define what might be acceptable targets within the JPEG file structure for encryption.
- Analysis of earlier methods of encryption for performance and effectiveness.

- Analysis of different encryption methods and targets in the JPEG image file for percentage of file encryption vs. image corruption.
- Analysis of different encryption methods and targets in the JPEG image file for the encryption target's susceptibility to attack.
- Final stage of the research analysis will evaluate and get approved by the Sponsor an acceptable performance evaluation taking into account all necessary factors that the research will review.

## **3. REQUIREMENTS**

The requirements have been divided into several logical sections. These sections include the requirements of the Supporting Environment, Functional Requirements, Performance Requirements, and Documentation and Release Requirements.

#### ➢ 3.1. Supporting Environment

The supporting environment includes specification of both the expected environments that the package should be able to perform in and the form in which the package will be written. There is also a basic specification of the hardware environment the package will require to be run in.

#### • 3.1.1. Software

- Package to be operational in Linux Red Hat 9.0, Windows XP and Mac OS X.
- Package to be written in ANSI C/C++ incorporating the Independent JPEG Group (IJG) package.
- Package should not change IJG's claim of wide portability (see <u>http://www.ijg.org</u> for specific environments.)
  - Web page will be built on a server and OS supplied by the Sponsor.
    - ✓ Web page to be viewable on Internet Explorer 6 and Safari 1.0.
    - ✓ Web page will use HTML version 4.01.

#### • 3.1.2. Hardware

0

- Package should be able to be run on any computer system supporting color graphics.
- Generic color monitor and JPEG image viewing system outlined above.
- Mouse, and Keyboard.
- Hardware supports the software environment outlined above.

#### > 3.2. Functional Requirements

Functional Requirements specify all of the functionality that ISE is required to provide. This includes functionality interfacing to the software package, and the

commands supported by the software package. The requirements of the web page created to support the package will also be listed in this section.

#### • 3.2.1. Required Operations

- Encrypt a standard image file in cooperation with the standard JPEG encoding format.
- Maintain compliance to the JPEG compression standard.
- Decrypt a standard compressed JPEG image file in cooperation with the Standard JPEG decoding format.
- Level of encryption is not "secretive/military" but only to level of damage that would force subscription to image viewing.
- Time permitting; the package will also selectively encrypt audio files, possible mp3 or AC3, and/or text files, such as zip files. However, these are secondary options. The main goal of the project is to deliver a package that selectively encrypts JPEG files.
  - Any attempt at these secondary projects would follow the same line of research into implementation.

#### • 3.2.2. Interface to Package

- Must be able to read in either .jpg or .bmp files for encryption.
- Research will determine what file type the encryption module will output.
- Decrypt module will input the appropriate file type.
- Decrypt will output a standard .jpg file.
- Final product will be a software package with command line user interface and appropriate incorporation into the standard JPEG tools.

#### • 3.2.3. Commands for software package

- Encrypt -- take a standard .jpg or .bmp file and convert to an encrypted JPEG file.
- Decrypt -- take an encrypted JPEG file and convert to a standard .jpg file.

#### • 3.2.4 Supporting Web Page

- To be built on server and OS provided by Sponsor.
- Contain links explaining the purpose of the software package provided by Team ISE.
- Contain links to downloadable version of the software package.
- Contain links to the software documentation as well as providing the user with the ability to download the documentation.
- Contain open source files of the software package.
- Contain links to other sources of related information.

#### ➢ 3.3. Documentation and Release Requirements

The following requirements specify the documentation that is to be provided, along with issues related to the release and delivery of the final product.

#### • 3.3.1. Documentation Requirements

- Man Page -- standard UNIX man page.
- User Tutorial -- presentation of system for first-time user.
- Research paper written up in style of the Sponsor's MPEG reference paper.
- Web site to include all code and documentation and supporting links.

#### • 3.3.2. Release Requirements

- Delivered as zipped files for Unix, Windows and Mac users.
- File will include entire source tree of software
- File will include installation programs for automatic generation and installation of executable and preview/evaluation programs.
- O Documentation provided only on the website for download.

## 4. SUMMARY

The purpose of this document was to give an initial outline for the path of research and the set of requirements for the ISE software package. These requirements include the software and hardware environments the application will run on, the functional requirements, the research and analytic requirements, and the supporting and research document's requirements that will be included along with the software package. These requirements will be modified at a later date when more information is known about completing the software package.

# System Architecture

# Team ISE Image Selective Encryption

#### System Architecture September 30th, 2003

## Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffery Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

10

## **Project Proposal**

The selective encryption project (Team ISE) is being sponsored by Assistant Professor of Computer Science, Tom Lookabaugh. Dr. Lookabaugh teaches and researches in the technology and practice of video communication, high technology businesses, and the intersection of policy, innovation, and management. His website contains a great deal of information on his research projects and responsibilities: http://itd.colorado.edu/lookabaugh/.

While many compression techniques have allowed an increase in the flow of traffic across the lines of the Internet, the files they produce are largely unprotected by efficient security measures. They are generally unencrypted and susceptible to unauthorized viewing. Team ISE will be working to incorporate encryption into common compression schemes starting with the JPEG image standard. While the final product is not required to provide more than the classes that would define the encryption and decryption methods, the initial portion of the project is oriented around the research and analysis of the most workable methods for securing compressed files. For this we will be developing a preview and testing suite with a simple graphical interface providing the ability to attack different portions of the compression standard.

The immediate efforts of the team will focus on developing selective encryption for the JPEG standard. If that portion of the project is able to be finished in a reasonable period of time, the team will venture into developing schemes for audio and text compression standards (MP3, zip, etc.)

Therefore the design of the test suite will first be for JPEG development. The test suite will utilize a pattern or process that can easily be extended to other desired formats.

Finally, the team will construct a permanent website which will allow anyone to download the team's previews, products, code and documentation. The site will be constructed on a computer and operating system provided by the Sponsor.

## **Table of Contents**

- 0. TITLE (COVER)
  - Project Proposal (p. i)
  - Table of Contents (p. 4)
- 1. INTRODUCTION (p. 1-2)
  - Figure 1.1 (p. 1)
- 2. INVOCATION (p. 2-3)
  - 2.1. Production Code (p. 2-3)
    - Parameters
  - 2.2. Test Suite (p. 3)
    - **o** Graphical User Interface
  - 2.3. Website (p. 3)
- 3. USER INTERFACE (p. 3-6)
  - 3.1. Production Code (p. 3)
    - Parameters
  - 3.2. Test Suite (p. 3-6)
    - Figure 3.2.1 (p. 6)
    - Graphical User Interface (p. 4-6)
  - 3.3. Website (p. 6)
    - Figure 3.3.1 (p. 6)
- 4. HIGH-LEVEL MODULAR DECOMPOSITION (p. 7-9)
  - Figure 4.1 (p. 7)
  - 4.1. ISE Website (p. 7)
  - 4.2. ISE Encryptor (p. 8)
  - 4.3. ISE Decryptor (p. 8)
  - 4.4. ISE Test Suite (p. 9)
- 5. FILE DESCRIPTIONS (p. 9-10)
  - 5.1. Input Files (p. 9)
  - 5.2. Output Files (p. 9)
  - 5.3. Test Suite Files (p. 9)
  - 5.4. Optional Project Extension Files (p. 10)
- 6. SUMMARY (p. 10)

## **1. INTRODUCTION**

Team ISE is being sponsored by Assistant Professor of Computer Science, Tom Lookabaugh: http://itd.colorado.edu/lookabaugh/.

Selective encryption is intended to utilize the standard formatting of commonly used compression schemes. Targeting small portions of a file that has been or will be divided into pieces defined by the standard algorithm can allow encryption of only a tiny portion of the file. If the target is chosen with care, the encryption can have the effect of damaging the usability of the file for the user who does not have the compatible decryption package.

Team ISE will first be developing a selective encryption scheme for the JPEG image standard. A standard encryption algorithm will be used to encrypt target portions of the file. However, because it is not the goal of Team ISE's project, the team will not be developing or implementing the encryption algorithm. However, the team will include a freely available encryption implementation with the software package. Current encryption candidates are the RC4 stream cipher algorithm and the AES block cipher. However, the team is not limited to these options.

The final product that the team will be providing to the open source community will be methods or classes that will provide the ability to encrypt and decrypt a file created by or used with a standard compression method. These methods or classes will be written in ANSI C/C++. See Figure 1.1 for an overview of the usage of the team's final product. Given a reasonable amount of time the team will also attempt to create selective encryption schemes for other compression standards, such as audio and text.



Figure 1.1: Conceptual Overview of ISE Software

To develop this and possibly other products, the team will be creating a test suite for use in establishing a workable encryption scheme. Again, there will be no work by the team to create an encryption algorithm; the target is only the development of a scheme for <u>selective</u> encryption. The intention of selective encryption is that it be such a system that it is possible to use any standard encryption algorithm. The test suite will effectively simulate an end user product. It will utilize a standard encryption algorithm but the end user would not be required to use any algorithm chosen by the team.

There are several necessary functions that the test suite must have. It will first be able to preview a standard file. Each compressed file is divided into separate pieces of information as per the compression standard. Therefore, the test suite will provide the ability to manipulate the various portions of the compression standard in each compressed file. Having manipulated the file, the test suite will be able to preview the encryption attempt without the benefit of compatible decryption. It will also have the ability to preview a standard file that has been both encrypted and decrypted. The decryption options will allow the user try to defeat the encryption methods (let the user put on a black hat.) Any selective encryption scheme could be developed using a package that implemented these features.

The test suite will be developed with Visual Studio C#.

The test suite will use the encryption and decryption classes or methods that the team is developing. The methods will be developed in standard ANSI C/C++, as per the specifications document, and will be able to be called by the test suite.

The website that will be constructed by the team will be on a computer and operating system provided by the Sponsor. It will have a simple home page with links to previews, final product code and to documentation.

This document will primarily define the high-level design architecture for the final product, the test suite, and for the website to be used and developed by the team. For each element of the project, this document will outline the design of invocation, user interface, high-level modular decomposition and file description.

### 2. INVOCATION

Throughout this document design specifications will be laid out for the final product, the test suite and the website. There will be more or less detail depending on the necessary complexity of the object being described.

#### 2.1. Production Code:

- On invocation, the ISE encrypter will be given an image file path, a flag indicating the target portion(s) of the image for encryption, a step size value for the quality of encryption (the amount or portion of the file to be encrypted), the encryption key file path, (optional) the output file name and path.
- On invocation, the ISE decrypter will be given an encrypted image file path, a flag indicating the target portion(s) of the image for decryption, a step size value for the quality of decryption (the amount or portion fo the file to be decrypted), the decryption key file path, (optional) the output file name and path.

#### 2.2. Test Suite:

 The Previewer will start up as a version 1.1 .NET windowed application using a default test image and an appropriate set of default parameters for the encryption and decryption modules.

#### 2.3. Web Site:

 The web site will be accessed through a fixed IP on the University of Colorado network and will have a home page that will identify the project and provide links to previews, final product code and documentation.

## **3. USER INTERFACE**

The general high-level design for the various user interfaces will be laid out as follows.

#### 3.1. Production Code:

• The user interface will be strictly a command line environment where the command encryption or decryption is given along with the necessary parameters.

#### Parameters:

- int file\_type\_encryption (image\_file\_path, target\_flag(s), encryption\_quality, key\_file\_path, output\_file\_path)
- int file\_type\_decryption (image\_file\_path, target\_flag(s), decryption\_quality, key\_file\_path, output\_file\_path)

#### 3.2. Test Suite:

 The test suite will be constructed in Visual Studio C# (see Figure 3.1 below) and will attempt to make development of the selective encryption scheme very organized and straightforward.

- There will be a tab corresponding to each major portion of the compression standard. In the JPEG standard, the compressed file is stored in easily parsible portions and each has a clear identification and purpose (Huffman encoding tables, Quantizer tables, etc..)
- The test suite will have a modular design which will allow the team to scale it to work for other compression formats, such as MP3 and ".zip".

🛃 ISE JPEG Manipulat	tor - Version 1.0.0 - C:\test3.jpg	- 🗆 🗙
Elle About		
Console Original Picture	e   Changed Picture	
File Information		
Original File Name:	C/test3.jpg	
Changed File Name:	C:\testchanged3jpg	
File Size:	102262 bytes	
File Comments:		
	Update P	icture

Figure 3.1: Screen Shot of ISE Testing Suite being prototyped

#### Graphical User interface:

(Comments will only be placed under the items not obvious.)

- Menu Bar
  - o File
    - Open project
    - Save project
    - Save project as
    - Exit
  - o Help
    - Help
    - About (Will include versioning information.)
- Image Display section
  - Console tab
    - Original image preview
      - Displays the original unmanipulated image.
    - Final output image preview

- Displays the final image after encrypt and decrypt.
- File information tab
  - Will include the original file name and a button to allow opening another file.
  - Will display the output file name and will include a browse button so that the output name can be changed without overwriting any existing files.
  - ➤ Will display file size.
  - Will include a section for adding comments to the output file.
- Huffman Encoded Scan Data tab
  - > Will display the Scan header.
  - > Will display the encoded data (start of scan).
  - If the file is manipulated, this tab will display the original header and the original encoded data for comparison.
- Quantizer Table tab
  - Will display up to 5 quantizer tables and if any are modified will also display the unmanipulated table.
- Huffman Table tab
  - Will display up to 5 Huffman tables and, if any are modified, will also display the unmanipulated tables.
- Application Data tab
  - Fields for up to 10 of the available application data flags.
- Miscellaneous data tab
  - Fields for:
    - Restart Interval
    - Number of Lines
    - □ Expand Image
    - Restart modulo 8 occurred at byte index
    - Hierarchicial Progression
    - Program Errors
- Encryption tab
  - > Will have check boxes for all possible flags.
  - Will have radio buttons for any implemented encryption methods.
  - Will display the path to the encryption key and will allow this path to be set or browsed.

- > Will have a field to define the step size for the quality of encryption.
- Will call the encrypt function outlined in section 3.1.
- Decryption tab
  - > Will have check boxes for all possible flags.
  - Will have radio buttons for any implemented decryption methods.
  - Will display the path to the decryption key and will allow this path to be set or browsed.
  - Will have a field to define the step size for the quality of the decryption.
  - Will call the decrypt function outlined in section 3.1.
- Project comments tab
  - Will have a field for project comments to be entered and saved with the project.
- Original picture tab
  - Will display the original picture without the size alterations made in the display window.
- Final Image tab
  - Will display the encrypted image without the size alterations made in the display window.

#### 3.3. Web Site:

 The web site will be a very simple construction with a home page directing users to previews, final product code, documentation and test suite.



Figure 3.3.1: Screenshot of ISE Web Page

## 4. HIGH-LEVEL MODULAR DECOMPOSITION

A high-level modular decomposition of Team ISE's software project is presented in Figure 1.1. The project consists of four main modules:

- ISE Website
- ISE Encryptor
- ISE Decryptor
- Test Suite



Figure 4.1 High level modular decomposition of ISE

Any comments in Sections 4.2 through 4.4 that seem to apply only to JPEG images can and will be adopted to any other compression standards that the team may attempt during the project.

#### 4.1 ISE Website

- The website will serve as the distributor for Team ISE's software package.
- It will also include links to all documentation provided by the team about the software package and the research behind the implementation.
- The website will display the product and output. This will either be done with a screen shot or a possible test encryption service found at the site.

#### 4.2 ISE Encryptor

- The ISE encryptor will be invoked by a command line call.
- The encryptor will take a few parameters, namely the JPEG filepath, a flag indicating the target portion of the file for encryption, a step size value to define the quality of encryption, the pathname of the location of the key used in the encryption, and an optional output file name and path.
- The encryptor will gracefully terminate if the file imported to it does not end in a ".jpg" extension.
- If the optional file name and path is not included, the encryptor will produce an encrypted file in the current directory with the same name as the orignal JPEG file, however it will contain a ".ise" extension.
- The encryptor module will allow the user to determine which portion(s) of the JPEG file they would like to be encrypted. The portions of the file that can be targeted are defined by the compression standard (these are outlined in Section 3.2 for the JPEG standard.)
- The Module will allow adjustment of the desired quality of encryption. This will vary in implementation between standards. For some formats this will be a percentage of the file to be encrypted. For other formats this might define what portions of the file are to be encrypted.

#### 4.3 ISE Decryptor

- The ISE decryptor will also be invoked by a command line call.
- The decryptor will take the following parameters: encrypted image file path, a flag indicating the target portion of the image for decryption, a step size to define the quality of decryption, the decryption key file path, (optional) the output file name and path.
- Like the encryptor, if an output file name and path is not specified, the decryptor will produce a standard JPEG file with the same name as the encrypted file, however the ".ise" extension will revert back to a ".jpg" extension and a number will be assigned to the end of the file name string("dog.ise" will become "dog001.jpg".)
- The decryptor will gracefully terminate if it is run on a file without the ".ise" extension.
- The decryptor module will allow the user to determine which portion(s) of the file they would like to be decrypted. The portions of the file that can be targetted are defined by the compression standard (these are outlined in Section 3.2 for the JPEG standard.)
- The Module will allow adjustment of the desired quality of decryption. In most cases this would be required to match the encryption step size setting.

#### 4.4 ISE Test Suite

- The ISE Test Suite will provide the team with valuable information about the contents of the compressed JPEG file before and after encryption.
- The Test Suite will also be available to users who wish to view the file changes that can be made to JPEG files using selective encryption.
- It will also display the original and final JPEG images side by side allowing the user to visually compare the differences in image quality. The test suite will implement and include all of the functionality described in section 3.2 for the JPEG standard.

## **5. FILE DESCRIPTIONS**

There are several files that will be used by Team ISE's software package. They will be divided into the following categories:

- Input Files
- Output Files
- Test Suite Files
- Optional Project Extension Files

Again, any comments in the following sections that seem to apply only to the JPEG image standard will be adopted by the team and applied to any other compression standards attempted by the team during the project.

#### 5.1 Input Files

- The encryptor will require standard JPEG files. The file will have to end in ".jpg" and will have to be a standardly recognizable JPEG image.
- The decryptor will require files that have been output by the ISE encryptor ending in the ".ise" extension.

#### 5.2 Output Files

- The encryptor will produce encrypted JPEG files ending in a ".ise" extension
- The decryptor will produce standard JPEG files ending in a ".jpg" extension

#### 5.3 Test Suite Files

• The test suite will require standard JPEG files. The file will have to end in ".jpg" and will have to be a standardly recognizable JPEG image.

#### 5.4 Optional Project Extension Files

 Time permitting, Team ISE will provide encryption and decryption modules to selectively encrypt other file formats, for example MP3 or ".zip" files. In this case, the encryptor will work on files ending in the standard extensions for these compression methods, and will produce selectively encrypted files with ."ise" extensions. The decryptor module will work on files with this new extension and reproduce the original files with their standard file extensions. Again, this is potential additional work to be performed by Team ISE. The main goal of the project is the production of JPEG encryption and decryption modules.

## 6. SUMMARY

This has been a very high-level view of the initial thoughts on a system architecture for Team ISE's selective encryption project. The document includes information on the architecture of the final encryption and decryption modules, the team's distribution website, as well as extensive planning on important tools which will be implemented for the team's reseach, development and testing. These tools will be available to users who wish to view the inner workings of the selective encryption methods. Thought was also given to the scalability of the project, specifically the inclusion of encryption and decryption modules for compression standards other than JPEG. The design and architecture allows for the extension of the available modules to other formats. It should serve as a strong beginning from which the team can start prototyping. It will also allow the team to begin the formulation of a more detailed design of the product.

# Overview Presentation

# Team ISE Image Selective Encryption



# Team ISE Image Selective Encryption

**Joe Jarchow** Shinya Daigaku **Joseph Kadhim Andrew Pouzeshi Geoffrey Griffith**
#### **Sponsor**

- Tom Lookabaugh
  - Assistant Professor in Computer Science Department
  - Faculty Director of Interdisciplinary Telecommunications
  - Research into areas such as
    - Selective Encryption
    - Broadband
    - Multimedia and Distance Learning



### Problem:

- Multimedia files are often very large
- Encrypting can require extensive processing time
- Can also increase the file size

 No current intelligent method for securing multimedia information without encrypting an entire file



# Solution:

- <u>Selective Encryption</u>
- <u>Team ISE</u>





### **Definition of Selective Encryption:**

 Selective encryption applies encryption to a subset of a file with the expectation that the entire file will be rendered useless to anyone who cannot decrypt that subset.

### **Successful Selective Encryption:**

 The right subset must be chosen or the file may not very secure

Example:

For ixstanxe, obsxurinx everx fiftx lettxr of ax Englxsh sextencx does xot maxe it pxrticxlarlx hard xo read.

Joe J

### **Sponsor's current research into MPEG**

- Target for degraded image rather than secretive
- Encryption of less than 10% of file
- Allow only a small reduction in efficiency (nominal increase in bandwidth)
- Provide solution to the cryptography community for review and testing





# 

# Joe J



# 

Joe J

### **Team ISE Presentation Outline:**

- General introduction to compressed media
- Requirements for JPEG image approach
- Architecture and Design
  - C++ Production Code
  - Website
- Demonstrate research
  - C# JPEG Manipulator



# Compressed Multimedia Overview



Multimedia file examples:

- MPEG 1, 2, and 4 video
- MP3 (MPEG 1 Layer 3)
- JPEG, zip and voice

Each uses a standard compression scheme



## **Multimedia File Components**

- Compressed files are partitioned into standard pieces
- Some parts are *Descriptive*
- Others are Mathematical
- These components are referred to as frames within a compressed media file



### Frame:

- Consists of a marker, header and data
- Example frame (piece of a JPEG file):

ff e0 00 10 4a 46 49 46 00 01 01 01 00 48 00 48 00 00



### Marker:

- Indicates what kind of data follows.
- Example marker:

ff e0 (indicates Application Data)



### Header:

- Describes the data to follow
- Size, parameters, descriptor, etc.
- Example header:

00 10 -- (16 bytes of data will follow)



### Data:

- The information itself
- Example data:

4a 46 49 46 00 01 01 01 00 48 00 48 00 00 (16 bytes of information indicating what application created the file.)



Real cryptography approaches a solution from both a white hat and black hat view

White hat -- Designs a secure system
Black hat -- Attempts to break into the

system



## **Closed encryption method**

- NSA has been faulted in not publishing their algorithms
- Only people on the inside actually know the algorithm

### **Open encryption method**

- Algorithm is published publicly
- People not involved in producing the algorithm can review and attempt to break the encryption
- If the encryption is broken, we can improve the algorithm



Why do we want to use selective encryption?

Drawbacks to encrypting an entire fileTakes time to encrypt the data

Sometimes makes the file bigger

Drawbacks to selective encryption
Slightly more complex than encrypting an entire file
Have to find the right target to encrypt



 Many multimedia compression algorithms concentrate critical information in a small portion of the bit stream

 Encrypting this portion could render the remaining information useless

 Selective encryption involves selecting which pieces of a bit stream to encrypt in order to minimize the amount of encryption applied and maximize the amount of damage



# **JPEG Specification and Approach**



### **JPEG Selective Encryption**

 JPEG is a compressed image format consisting of frames (markers, headers, data)

 Through a process of analysis, we were able to find a target appropriate for successful selective encryption

• We are only required to handle Baseline JPEG Images, the most commonly used compression mode for JPEGs

### **Criteria For Bad Targets**

- Optional markers
- Markers not used in Baseline JPEG images
- Markers that contain information that does not affect visibility of the image
- Markers that contain information that can be easily guessed or forged by a hacker

Markers immediately eliminated:

- APP Application
- COM Comments
- DAC Define Arithmetic Conditioning Tables(Not part of Baseline Compression)
- DHP Define Hierarchical Progression (Not part of Baseline Compression)
- DNL Define Number of Lines
- DRI Define Restart Interval
- EOI End of Image
- EXP Expand (Not part of Baseline Compression)

Markers immediately eliminated:

- JPG Reserved for Future Extensions
- RES Reserved
- RST Restart (Not part of Baseline Compression)

• TEM - Temporary (Not used in Baseline Compression)

- Markers themselves are predictable
- Scan Header easily reconstructed



# Remaining Target List for Selective Encryption

- Encoded Data Stream
- Quantizer Tables
- Huffman Tables

### **Target Analysis**

 Two C++ programs were written for target analysis:

**Convert and Analyze** 

### Convert

- Binary to Hexadecimal
- File information for a single JPEG
  image



This is an ASCII representation (in hexadecimal) of the binary values found in the file : Dust.jpg

Markers Found:=========

ff d8 -- Start of Image

ff e0 - Application Data - 00 10 - (16 bytes) - 4a 46 49 46 00 01 01 01 00 48 00 48 00 00

ff db -- Define Quantization Table -- 00 43 -- (67 bytes) -- 00 06 04 05 06 05 04 06 06 05 06 07 07 06 08 0a 10 0a 0a 09 09 0a 14 0e 0f 0c 10 17 14 18 18 17 14 16 16 1a 1d 25 1f 1a 1b 23 1c 16 16 20 2c 20 23 26 27 29 2a 29 19 1f 2d 30 2d 28 30 25 28 29 28

ff c0 -- Huffman Table -- Baseline DCT -- 00 11 -- (17 bytes) -- 08 01 cb 02 4a 03 01 22 00 02 11 01 03 11 01

ff c4 -- Huffman Table -- 00 1f -- (31 bytes) -- 00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09 0a 0b

### Analyzer

- Compute file information for multiple
  JPEG images
- Average file size
- Average number of Huffman tables
- Average size of Huffman tables combined
- Average number of Quantizer tables
- Average size of Quantizer tables
   combined



### Analyzer

- Average size of the encoded stream
- Average number of markers
- Number of files processed
- Percent of the file dedicated to:
  - Huffman tables
  - Quantizer tables
  - Encoded Stream



### **Test Cases for JPEG Analysis**

- Over 2500 JPEG images randomly selected from the Internet
- Digital Photograph vs. Manmade
- Size ranges: 10-19KB, 100 KB, 1 MB, and larger
- Resolution Ranges: 320x240, 640x480, and 800x640 Pixels



### **Encoded Data Stream**

- SOI (start of image) marker
- Compressed data stream
- Takes up a large portion of the file
- Averaged 90% of the file!





### **Quantizer Tables**

- DQT (Define Quantization Table) markers
- Averaged 0.88% of the file
- Unpredictable affects on image
- Might not visually damage the image at all!
- Could be replaced with another Quantizer table!



### **Huffman Tables**

- DHT (Define Huffman Table) markers
- Averaged 1.84% of the file
- Considerable damage to image
- Mathematically derived from the image
- This makes the Huffman Tables a
  perfect target for Selective Encryption



# **Architecture and Design**


# **ISE ARCHITECTURE**

- Invocation
- ISE Class Inheritance
- User Interface
- High-Level Modular Decomposition
- File Description



# **ARCHITECTURE: INVOCATION**

- Test Suite
  - Application designed to aid the research into JPEG images
  - Easy to test the effects of manipulating each type of frame
- Production Code
  - Final release of C++ package implementing selective encryption for successful targets



# **ARCHITECTURE: INVOCATION**

# **Test Suite:**

• Invoked as a windowed Version 1.1 .NET application





**Production Code:** 

- ISE ENCRYPTOR and ISE DECRYPTOR
  - Input File Name and Path
  - Output File Name and Path
  - Encryption Key
  - Flags



## **ARCHITECTURE: ISE CLASS INHERITANCE**

## **ISE Super Class**

• The ise class will define our process and will be inherited by our specific encryption classes:

class ise
{
 public:
 virtual int selectiveEncryption(. . .) = 0;
 virtual int selectiveDecryption(. . .) = 0;
};



**ARCHITECTURE: ISE CLASS INHERITANCE** 

**JPEG Encryptor Class** 

 The jpeg\_file class is the immediate goal of project ISE

```
class jpeg_file : public ise
{
    public:
    virtual int selectiveEncryption(. . .);
    virtual int selectiveDecryption(. . .);
};
```





## **ARCHITECTURE: ISE CLASS INHERITANCE**

# **MP3 Encryptor Class**

• This class will be an extension to the project.

```
class mp3_file : public ise
{
    public:
        virtual int selectiveEncryption(. . .);
        virtual int selectiveDecryption(. . .);
};
```



# **ARCHITECTURE: USER INTERFACE**

- Test Suite
- Production Code





ARCHITECTURE: ISE USER INTERFACE Production Code:

- Utilized through API
  - Encryptor API

int selectiveEncryption(
 ifstream &input\_file\_stream,
 ofstream &output\_file\_stream,
 char\* key\_material,
 char\* encryption\_flags,
 int num\_flags, int quality);



ARCHITECTURE: ISE USER INTERFACE Production Code:

- Utilized through API
  - Decryptor API

int selectiveDecryption(
 ifstream &input\_file\_stream,
 ofstream &output\_file\_stream,
 char\* key\_material,
 char\* encryption\_flags,
 int num\_flags, int quality);



# **FILE DESCRIPTIONS**

**Input Files:** 

- Encryptor:
  JPEG file with '.jpg' extension
- Decryptor:
  - Encrypted file with '.ise' extension





# **FILE DESCRIPTIONS**

**Output Files:** 

- Encryptor:
  - Encrypted file ending in '.ise' extension
- Decryptor:
  - JPEG file with '.jpg' extension



## **ISE WEBSITE**

Website is temporarily Available at:
 ucsub.colorado.edu/~pouzeshi/ISE

### Includes links to:

- Sponsor
- Documentation
- Test Suite/Production Code
- Other Relevant sites
- Team ISE info



PROJECT EXTENSIONS/CODE MODIFICATION

**Compression Type Extension:** 

- Time Permitting, the project will be extended to include:
  - MP3 Compression
  - ZIP Compression

• The Modular Design of the JPEG selective encryptor makes these extensions possible.



# **Manipulator Demonstration:**





# Requirements Specification

# Team ISE Image Selective Encryption

Requirement Specification

# Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

1 (3()

100

01

### **Project Proposal**

A constant amount of traffic flows between computers connected to the Internet. A large volume of information may take a long time traveling from destination to destination. The resulting speed reduction makes it desirable to compress the file as much as possible in order to send the smallest amount of data. Compression of data has allowed for the high-speed data transfers that have made Internet communication and business very workable.

In addition to sending the smallest amount of information possible, users also attempt to maintain a certain level of security upon their information. Due to the fact that common encryption methods generally manipulate an entire file, most encryption algorithms tend to make the transfer of information more costly in terms of time and bandwidth. Thus, users pay a price for security relative to their desired level of security. One possible solution would be a system of encryption that works cooperatively with the standard compression schemes. *Selective Encryption* of only a small percentage of the file's bits will facilitate this solution. Because most encryption schemes will make the file larger, selective encryption seeks only to encrypt portions of the file that will make it unusable. In other words, if a user does not have the proper decryption device, the file should not be usable. Selective encryption will minimize the necessary increase in file size due to encryption while maintaining a maximum level of uselessness, or damage, to the product.

An image could be encrypted with any of the sufficiently secure encryption algorithms available to the open source community, but this will usually result in a dramatic increase in file size that will severely increase transfer time over the Internet. However, selecting key parts of a file for encryption and only encoding those bits can actually render an image unusable. The initial statistical analysis done by the team will consist of specifically breaking down the standard JPEG compression scheme into its usable parts and evaluate which of these, if encrypted, will cause a potential user to pay for rights to the image or force subscription to the provider service.

Team ISE (Image Selective Encryption) will deliver a package for selectively encrypting JPEG (Joint Photographic Experts Group) still image files. The package will provide the tools necessary to encrypt the critical information of a JPEG file in cooperation with existing standard compression tools. This package will handle JPEG files in such a way that only a small percentage of the total file will be encrypted. Selective Encryption security will not extend to the level of military secrecy, but rather a level that would deter all but brute force attacks, allowing users to securely protect private JPEG images.

An additional aspect of the encryption analysis will be the determination of the specific targets in the file for encryption. For example in an MPEG file there are headers that contain a small portion of the overall number of bits but which are extremely vital to the reproduction of the movie by the user. So, if certain headers were to be encrypted the percentage of the file being manipulated would be less than ten percent of the total number of bits in the file. Although only a small portion will be encrypted, the resulting

damage experienced by an unauthorized user would be sufficient to cause the user to pay for the decryption package. However, there are other targets that, while they can be encrypted and will do sufficient damage, can be guessed by an attacker. The attacker could, with some degree off effort, render the file useful without use of the decryption software. For example, if the frame rate of an MPEG file was encrypted, an attacker could try all three of most common frame rates and one of these is certain to produce the correct rate for the particular video. In the case of JPEG Selective Encryption, Team ISE will have to balance the targets for encryption against ease of simple attacks.

A permanent website will be constructed by the team to make the software package available to anyone interested in the software process. As it is vital to the world of cryptography to let the community view the approach, the first form of the working prototype will be made available on the website. From this, feedback can be received not only from the team itself, but also from the cryptography community at large.

So, following the guidelines of the ongoing MPEG research (also being guided by the sponsor), the team will study the JPEG process and earlier attempts at encryption. With the sponsor's assistance, Team ISE will devise a workable approach to handling individual JPEG images following the concept of selective encryption.

It is possible that the team will complete the JPEG process early enough in the year that they will able to apply the same approach to other types of compressed files (text, audio, etc.) However, this initial specifications document applies only to the envisioned JPEG project.

#### Table of Contents

- 1. Introduction (p. 1-2)
  - Figure 1.1: Conceptual Overview of ISE Software (p. 2)
- 2. Research Path (p. 3)
  - 2.1 Research and Analysis Requirements (p. 3)
  - 2.2 Research Related Products (p. 3)

#### 3. Requirements (p. 4-7)

- 3.1 Supporting Environment (p. 4)
  - 3.1.1 Software (p. 4)
    - 3.1.1.1 Runtime Environment (p. 4)
    - 3.1.1.2 Development Environment (p. 4)
  - o 3.1.2 Hardware (p. 4)
- 3.2 Functional Requirements (p. 5-7)
  - 3.2.1 Required Operations (p. 5)
  - 3.2.2 Package Functionality (p. 5)
  - 3.2.3 ISE Function Interfaces (p. 5)
  - 3.2.4 Test Suite Requirements (p. 6)
    - 3.2.4.1 Test Suite Display (p. 6)
    - ✓ Figure 3.2.4.1 (p. 6)
    - 3.2.4.2 Test Suite Functionality (p 6.)
  - 3.2.5 Supporting Web Page (p.7)
- 3.3 Documentation and Release Requirements (p. 7)
  - 3.3.1 Documentation Requirements (p. 7)
  - 3.3.2 Release Requirements (p. 7)

#### 4. Future Enhancements (p. 8)

- 4.1 Selective Encryption of MP3 Files (p. 8)
- 4.2 Selective Encryption of ZIP Files (p. 8)
- 5. Summary (p. 9)
- 6. Glossary (p. 10-11)
- 7. Related Documents (p. 12)

### **1. INTRODUCTION**

Team ISE is being sponsored by Assistant Professor of Computer Science, Tom Lookabaugh, at the University of Colorado: <u>http://itd.colorado.edu/lookabaugh/</u>. Tom Lookabaugh is currently involved in selective encryption research on standard MPEG (Moving Picture Experts Group) files and is interested in researching the application of Selective Encryption for other multimedia formats.

The goal of selective encryption is to minimize the amount of encryption applied to a file while maximizing the damage done to the image being viewed by a user not in possession of the authorized decryption package. Complete encryption is not a requirement of the process, nor is rendering the file useless to the level of complete military secrecy. It is acceptable for an attacker to be able to view portions of the file; however, the file should be distorted enough that an attacker would not wish to use the encrypted file, but would rather purchase or subscribe to the decryption method for access to the original files.

Multimedia files prove to be a good subject for selective encryption, as these files tend to be very large and employ compression algorithms that concentrate critical information in small portions of their bit stream. If the critical data in certain multimedia standards is encrypted properly, the remaining information becomes useless to those without the appropriate decrypter. There are many types of compression algorithms that fit this description, such as MPEG 1, 2 and 4 video, G.723 and G.729 video, AAC audio, MP3 audio, JPEG and JPEG2000 image formats. Applying a Selective Encryption security solution to selected multimedia formats will greatly increase the protection level of important information.

The focus of the ISE project is to research and develop an algorithm for selectively encrypting the JPEG *baseline* compression image standard. The product of the research and development will be a package that will encrypt a file so that the amount of the file being encrypted is relatively small (on the order of 1-2% of the total file). The product will be delivered in a package that will include an encrypter and a decrypter for JPEG files, a website to facilitate the delivery of the product and documentation about the process. The encrypter and decrypter will encrypt and decrypted selected targets contained within JPEG files. The ISE project will employ the AES (Advanced Encryption Standard) for our Selective Encryption algorithm. This package will be made available in a purely open source form on our final website.

In addition to the package containing the decrypter and encrypter, Team ISE will also provide a test suite available to prospective users. The test suite will be used to aid in the research, development and testing of the team's final product. The test suite will provide the functions necessary to completing this project. First, it will allow the user to preview a standard JPEG image. Second, the test suite break down the various portions of a JPEG image and provide the ability to manipulate the data of all of the pieces the particular file. Third, after altering the data in any particular file, the test suite will provide the capability to preview the encryption attempt without the benefit of compatible decryption. Forth, the suite will have the ability to decrypt an encrypted file. The decryption options will allow the user try to defeat the encryption methods (let the user put on a black hat). Any selective encryption scheme could be developed using a package that implemented these features, however, the delivered test suite will only employ the AES encryption scheme chosen by the team. The test suite will be available to download from the team website.

The final website will be deployed on a sponsor provided Apache web servers. The machine facilitating the web server will use the Linux Red Hat 9.0 operating system platform. The team will acquire a fixed IP address from the proper University of Colorado authorities and will develop a simple website capable of delivering information to viewers about the benefits and application of Selective Encryption technology. The site will provide users the option to download and use the final software package. The site will also provide links to important information and will remain in place as long as the sponsor deems necessary.

The envisioned software package will accomplish the complex task of selectively encrypting a JPEG baseline standard image, while providing a simple user interface to users. Team ISE has identified three specific types of users: high-end art users, typical Internet image users, and small, low-end image users. The research and software will be tailored to these users' needs. Figure 1.1 is a flow chart showing the general logic design of the team's final product.



Figure 1.1: Conceptual Overview of ISE Software

Information regarding the research required by the sponsor is further outlined in the next section. Details regarding the requirements of Team ISE's Selective Encryption project are then presented, followed by short discussions of possible requirements alternatives and future enhancements. These details are concluded with a summary followed by a glossary of important terms and a list of related readings.

### 2. RESEARCH PATH

#### > 2.1. Research and Analysis Requirements

The research and analysis will be the initial part of this project. The final product of the research will allow the team to determine a specific approach to this form of Selective Encryption. The sponsor considers the research done by the team equally as important as the delivery of the final product. The research portion of this project will include:

- A proportional analysis of a large quantity of JPEG images to define what might be acceptable targets within the JPEG file structure for encryption.
- An analysis of earlier methods of encryption for performance and effectiveness.
- The analysis of potential encryption methods and targets in the JPEG image file for percentage of file encryption vs. image corruption.
- The analysis of different encryption methods and targets in the JPEG image file for the encryption target's susceptibility to potential attack.
- The final stage of the research analysis will conclude with approval from the sponsor on the useful approaches and corresponding performance issues.

#### > 2.2. Research Related Products

Following significant discoveries throughout the project the Sponsor requires that the team will produce research documentation to be presented at applicable security and compression conferences.

### **3. REQUIREMENTS**

The requirements have been divided into several sections based upon the category of the requirement. These categories consist of the Supporting Environment, Functional, Performance, Documentation and Release Requirements. Each of the requirements is defined below.

#### ➢ 3.1. Supporting Environment

The supporting environment includes specification of both the expected environments that the package should be able to perform in and the form in which the package will be written. There is also a specification of the minimum hardware environment the package will require to be run on. The package referred to in the requirements consists of the encryption and decryption package. The test suite is not a portion of this package, and has its own runtime and language requirements.

#### • 3.1.1. Software

The supporting software environment includes the runtime environment as well as the development requirements.

- 3.1.1.1. Runtime Environment
  - Package to be operational in Linux Red Hat 9.0, Windows XP and Mac OS X.
  - ✤ The test suite is to be operational in a .NET environment.
  - Web page is to be viewable on Internet Explorer 6.0 and Safari 1.0.

#### • 3.1.1.2. Development Environment

- ◆ Package to be written in ANSI C/C++ specification.
- Package should not change IJG's claim of wide portability (see <u>http://www.ijg.org</u> for specific environments).
- CVS will be used for software versioning.
- Test suite to be written in the C# (C-sharp) programming language.
- The web page will be built on a server utilizing Linux Red Hat 9.0 operating system, supplied by the sponsor.
  - ✓ Web page will use HTML version 4.01.

#### • 3.1.2. Hardware

- Package should be able to be run on any computer system supporting color graphics.
- Generic color monitor and JPEG image viewing system outlined above.
- Keyboard as part of the user interface.
- Hardware supports the software environment outlined above.

#### ➢ 3.2. Functional Requirements

The functional requirements specify all of the functionality Team ISE's product is required to provide. These requirements will include the interface to our production code and outline the functionality that must be supported. The requirements of the test suite and web page will also be listed in this section.

#### • 3.2.1. Required Operations

- Encrypt a selected portion of a JPEG baseline standard image compression format.
- Decrypted files must maintain compliance to the JPEG compression standard.
- Level of encryption is not "secretive/military" but level to provide sufficient protect against all but brute force attacks.

#### • 3.2.2. Package Functionality

- Must be able to read in .jpg or files for encryption.
  - Encryption software must gracefully terminate upon receipt of other file types.
- Must output a file ending with an .ise extension. The .ise extension denotes a Team ISE selectively encrypted file.
- Decrypt module will input and process .ise file types.
  - Decryption software must gracefully terminate upon receipt of other file types.
- Decrypt will output a standard .jpg file.
- Final product will be a software package that provides simple interface methods.

#### • 3.2.3. ISE Function Interfaces

This section illustrates the pseudo code form that the Team ISE Selective Encryption package interface methods must have.

- int selectiveEncryption(ifstream &input\_file\_stream, ofstream &output\_file\_stream, char\* key\_material, char\* encrypt\_flags, int num\_flags, int quality);
- int selectiveDecryption(ifstream &input\_file\_stream, ofstream &output\_file\_stream, char\* key\_material, char\* decrypt\_flags, int num\_flags, int quality);

#### • 3.2.4. Test Suite Requirements

- **3.2.4.1** Test Suite Display
  - ✤ The test suite must have a standard windows display.
  - The test suite must include buttons and tabs to display different portions of the JPEG image file data.
  - ✤ Figure 3.2.4.1 displays a screenshot of the test suite.

🔚 ISE JPEG Manipulat	or - Version 1.0.0 - C:\test3.jpg
Elle About	
Console Original Picture	Changed Picture
File Information	
Original File Name	
Original File Name.	C/Nest3jpg
Changed File Name:	C:testchanged3jpg
File Size:	102262 bytes
File Comments:	
	Update Picture

Figure 3.2.4.1 Test Suite Screen Shot.

#### • 3.2.4.2 Test Suite Functionality

- ✤ The test suite must be able to parse compressed JPEG files.
- The test suite must divide up and display the hexadecimal values of the different pieces of a JPEG file.
- Display the manipulated image of the file alongside the original image. The image size will be altered to fill the display windows.
- The test suite must allow the user to make changes to the data contained in each of the pieces of a JPEG file.
  - ✓ These changes must be incorporated into the encoding of the file, and the image displayed must be updated to be the image produced by the changes in the encoding.
- ✤ Include tabs to display the image without altering its size.
- Incorporate the encryption and decryption software methods provided in the final software product.
  - ✓ The test suite will provide a graphical user interface for the final Team ISE product package.

 Provide options for the user to save all of the information and changes to the current image.

#### • 3.2.5 Supporting Web Page

- To be deployed on machine provided by the sponsor.
- Contain links explaining the purpose of the software package provided by Team ISE.
- Contain links to downloadable version of the software package.
- Contain links to downloadable version of the test suit package.
- Contain links to the software documentation as well as providing the user with the ability to download the documentation.
- Contain open source files of the software package.
- Contain links to other sources of related information.
- Contain information about the sponsor.
- Contain information about Team ISE.

#### ➢ 3.3. Documentation and Release Requirements

The following requirements specify the documentation that is to be provided, along with issues related to the release and delivery of the final product.

#### • 3.3.1. Documentation Requirements

- Man Page A standard UNIX man page.
- User Tutorial A presentation of system for first-time user.
- Research paper(s) written in a style specified by the sponsor.
- A website to include all code and documentation and supporting links.
- Documents will be made available in Adobe PDF file format.

#### • 3.3.2. Release Requirements

- Product will be delivered as series of ZIP files for Unix, Windows and Mac users.
- Files will include installation programs for automatic generation and installation of executable and preview/evaluation programs.
- Documentation will be provided on the website for download.

### 4. Future Enhancements

Pending the early completion of the JPEG selective encryption methods and software, the following enhancements may be incorporated into the final product. These enhancements illustrate the development of selective encryption, and its spread into other areas in the future.

#### ➤ 4.1 Selective Encryption of MP3 Files

The project may be extended to include the MP3 file format. The team will research MP3 file formats and devise ways of selectively encrypting MP3 files. Completion of this will entail expanding the encrypter/decrypter software to include MP3 files. The software will output selectively encrypted MP3 files. For security and consistency, these encrypted files will have the same ".ise" extension as the encrypted JPEG files. The test suite will also be expanded to parse MP3 files and display the encoding to the user. Documentation will be updated to include descriptions of the MP3 encoder/decoder. Research papers involving MP3 selective encryption will be produced upon the Sponsor's request. The website will be updated to include all pertinent MP3 information.

#### ➤ 4.2 Selective Encryption of ZIP Files

Upon completion of both the JPEG and MP3 selective encryption targets, the project will be further extended to the ZIP file format. Team ISE will update the encrypter/decrypter software to perform upon ZIP file types. Again, for security and consistency, selectively encrypted ZIP files will end in the ".ise" extension. Research will be done to determine how to best perform selective encryption upon ZIP files. The test suite will again be expanded to parse ZIP files and display the encoding to the user. Documentation will be updated to include descriptions of the ZIP encoder/decoder. Research papers involving ZIP selective encryption will be produced upon the Sponsor's request. The website will be updated to include all pertinent ZIP information.

### **5. SUMMARY**

The purpose of this document was to give an outline for the path of research and the set of requirements for the ISE software package. These requirements include the software and hardware environments the application will run on, the functional requirements of the software, test suite and website, the research and analytic requirements, and the supporting and research document's requirements that will be included along with the software package. This document includes all necessary information for designing all of the necessary aspects of the Team ISE software.

### 6. Glossary

#### **AES (Advanced Encryption Standard)**

An encryption method that uses block ciphering.

#### ANSI C/C++

The standard C and C++ programming languages as defined by the American National Standards Institute.

#### **Black Hat**

The process of testing an encryption algorithm by trying to break the encryption using several different methods.

#### **Baseline JPEG**

A subset mode of sequential JPEG where the number of tables is restricted and the sample precision must be eight bits.

#### **C**#

A modern, object-oriented language that enables programmers to quickly build a wide range of applications for the new Microsoft .NET platform.

#### **Compression Algorithm**

An algorithm designed to compress a file, that is, utilizes patterns in a file to reduce the size of the file.

#### **CVS** (Concurrent Versioning System)

A code management system. CVS provides the ability to track (and potentially revert) incremental changes to files, reporting them to a mailing list as they are made, and can be used concurrently by many developers.

#### Decryption

The act of rendering an encrypted file into a know format.

#### Encryption

To convert computer data or messages to something incomprehensible by means of a key, so that only an authorized recipient holding the matching key can recover the original.

#### IJG (Independent JPEG Group)

An informal group that writes and distributes a widely used free library for JPEG image compression. IJG is not affiliated with the ISO committee. www.ijg.org/

#### **ISO (International Organization for Standardization)**

The world's largest developer of standards, particularly the development of technical standard.

#### JPEG (Joint Photographic Experts Group)

A compression technique for color images and photographs that balances compression against loss of detail in the image. The greater the compression, the more information is lost (this is called Lossy compression).

#### **Military Secrecy**

A level of secrecy where all information is hidden.

#### MP3 (MPEG-1 Audio Layer-3)

A standard technology and format for compression a sound sequence into a very small file (about one-twelfth the size of the original file) while preserving the original level of sound quality when it is played.

#### **MPEG (Moving Picture Experts Group)**

A standard for digital video and audio compression.

#### **Selective Encryption**

A method of encryption that exploits the relationship between encryption and compression to reduce encryption requirements, saving in complexity and facilitating new system functionality. Selective Encryption only encrypts a small portion of a file.

#### Visual Studio .NET

Microsoft's visual programming environment for creating web services based on use of the Extensible Markup Language (XML).

#### ZIP

A method of compressing text files.

### 7. Related Readings

#### [Lookabaugh and Sicker and Keaton and Gua and Vedula 2003]

Lookabaugh, T., and Sicker, D., and Keation, D., and Guo, W., and Vedula, I. *Security Analysis of Selectively Encrypted MPEG-e Streams*. 2003.

Tom Lookabaugh's description of the methods and results of applying selective encryption to MPEG-2 streams.

#### [Miano 99]

Miano, J. *Compressed Image File Formats*. Addison Wesley Longman, Inc., Reading, Massachusetts, 1999.

# Design Specification

# Team 1SE Image Selective Encryption

#### Design Specification 5 December 2003

# Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

100
# **Project Proposal**

Traffic constantly flows between computers connected to the Internet. Large volumes of information may take a long time traveling from destination to destination. Such a reduction in speed makes it desirable to compress the file as much as possible in order to send the smallest amount of data required. Thus, compression of data has allowed for the high-speed data transfers that have made Internet communication and business more feasible.

In addition to sending the smallest amount of information possible, users also attempt to maintain a certain level of security upon their information. Due to the fact that common encryption methods generally manipulate an entire file, most encryption algorithms tend to make the transfer of information more costly in terms of time and bandwidth. Thus, users pay a price for security relative to their desired level of security. One possible solution would be a system of encryption that works cooperatively with the standard compression schemes. *Selective Encryption* of only a small percentage of the file's bits will facilitate this solution. Because most encryption schemes will make the file larger, selective encryption seeks only to encrypt portions of the file that will make it unusable. In other words, if a user does not have the proper decryption device, the file should not be usable. Selective encryption will minimize the necessary increase in file size due to encryption while maintaining a maximum level of uselessness, or damage, to the product.

Team ISE (Image Selective Encryption) will deliver a package for selectively encrypting JPEG (Joint Photographic Experts Group) still image files. The package will provide the tools necessary to encrypt the critical information of a JPEG file in cooperation with existing standard compression tools. This package will handle JPEG files in such a way that only a small percentage of the total file will be encrypted. Selective Encryption security will not extend to the level of complete encryption, but rather to a level that would deter all but brute force attacks, allowing users to securely protect private JPEG images.

A JPEG image could be encrypted with any of the sufficiently secure encryption algorithms available to the open source community, but this can result in an increase in file size or can require a large amount of processing time. However, by selecting small but vital portions of a file and encrypting only those few bytes can render an image unusable. The initial statistical analysis done by the team will consist of specifically breaking down the standard JPEG compression scheme into its usable parts and evaluate which of the parts, if encrypted, will cause a potential user to pay for rights to the image or force subscription to the provider service.

An additional aspect of the encryption analysis will be the determination of the specific targets in the file for encryption. For example in an MPEG file there are headers that contain a small portion of the overall number of bits but which are extremely vital to the reproduction of the movie by the user. So, if certain headers were to be encrypted the percentage of the file being manipulated would be less than ten percent of the total number of bits in the file. Although only a small portion will be encrypted, the resulting damage experienced by an unauthorized user would be sufficient to cause the user to pay for the decryption package. However, there are other targets that, while they can be encrypted and will do sufficient damage, can be guessed by an

attacker. The attacker could, with some degree of effort, render the file useful without use of the decryption software. For example, if the frame rate of an MPEG file was encrypted, an attacker could try all three of most common frame rates and one of these is certain to produce the correct rate for the particular video. In the case of JPEG Selective Encryption, Team ISE will have to balance the targets for encryption against ease of simple attacks.

A permanent web site will be constructed by the team to make the software package available to anyone interested in the Team's project. As it is vital to the world of cryptography to let the community view the approach, the first form of the working prototype will be made available on the web site. From this, feedback can be received not only from the team itself, but also from the cryptography community at large.

So, following the guidelines of the ongoing MPEG research (also being guided by the sponsor), the team will study the JPEG process and earlier attempts at encryption. With the sponsor's assistance, Team ISE will devise a workable approach to handling individual JPEG images following the concept of selective encryption.

It is possible that the team will complete the JPEG process early enough in the year that they will able to apply the same approach to other types of compressed files (text, audio, etc.). However, this specifications document applies only to the envisioned JPEG project

# **Table of Contents**

1. INTRODUCTION	1
2. USER INTERFACE	3
2.1. ISE Class Production Code User Interface 3	3
2.1.1. Instantiation of the JPEG ISE Class 3	3
2.1.2. Usage of the JPEG ISE Class Methods 3	3
2.2. The JPEG Manipulator User Interface 5	5
2.2.1. JPEG Manipulator Invocation 5	5
2.2.2. The Manipulator's Menu Bar 5	5
2.2.3. The Manipulator's Console Tab 8	8
2.2.4. The Original Picture Tab1	2
2.2.5. The Manipulated Picture Tab1	.3
2.3. Team ISE Web Site User Interface1	4
2.3.1. ISE Web Site Invocation1	4
2.3.2. ISE Web Site Navigation1	4
3. DESIGN OVERVIEW	.8
3.1. High-Level Modular Decomposition1	.8
3.2. ISE Class Production Code Modules1	.8
3.2.1. ISE Encryptor Module 1	.8
3.2.2. ISE Decryptor Module 1	.8
3.3. JPEG Manipulator Test Suite Module 1	9
3.3.1. Standard Windows Form Application Methods 1	9
3.3.2. Manipulator Graphical Interface Methods 1	9
3.3.3. Manipulator Common Methods 1	9
3.3.4. Methods to Convert from Binary to ASCII 1	9
3.3.5. Methods to Convert from ASCII to Binary 1	.9
<b>3.3.6. Methods to Encrypt and Decrypt</b>	20
3.4. Team ISE Web Site Module	20
4. DESIGN	1
4.1. ISE Class Production Code Design	1
4.1.1. ISE Class Invocation	1
4.1.2. Public Methods of the ISE Class	2
4.1.3. Protected Methods of the ISE Class	6
4.1.4. Data Members of the ISE Class	8
4.1.5. JPEG ISE Class Invocation	8
4.1.6. Public Methods of the JPEG ISE Class	0
4.1.7. Protected Methods of the JPEG ISE Class	4
4.1.8. Data Members of the JPEG ISE Class	6
4.1.9. Algorithms Developed by Team ISE Used in the ISE Class 3	6
4.2. The JPEG Manipulator Design	8
4.2.1. Standard Windows Form Application Methods	8
4.2.2. ISE Manipulator Graphical Interface Methods	iy 
4.2.3. ISE Manipulator Common Methods	7
4.2.4. ISE Methods to Convert from Binary to ASCII	2

4.2.5. ISE Methods to Convert from ASCII to Binary	. 64
4.2.6. Data Members of the JPEG Manipulator	. 66
4.3. Team ISE Web Site Design	. 76
4.3.1. The ISE Web Site Index Page	. 76
<b>4.3.2. The ISE Menu</b>	. 76
4.3.3. The ISE Project Proposal Document	. 77
4.3.4. The ISE Documentation Page	. 77
4.3.5. The ISE Project Sponsor Page	. 77
4.3.6. The Team ISE Info Page	. 77
4.3.7. The ISE Download Page	. 77
4.3.8. The ISE Links Page	. 77
5. FILE DESCRIPTIONS	. 78
5.1. JPEG Standard Image Files	. 78
5.2. JPEG ISE Encrypted Files	. 78
5.3. Test Suite Manipulated Images	. 78
5.4. Test Suite Project Files	. 78
6. SUMMARY	. 80
7. GLOSSARY	. 81
8. RELATED READINGS	. 84

# **1. INTRODUCTION**

Team ISE is sponsored by Assistant Professor of Computer Science, Tom Lookabaugh, at the University of Colorado: http://itd.colorado.edu/lookabaugh/. Tom Lookabaugh is currently involved in selective encryption research on standard MPEG (Moving Picture Experts Group) files and is interested in researching the application of Selective Encryption for other multimedia formats.

The goal of selective encryption is to minimize the amount of encryption applied to a file while maximizing the damage done to the image being viewed by a user not in possession of the authorized decryption package. Complete encryption is not a requirement of the process, nor is rendering the file useless to the level of complete military secrecy. It is acceptable for an attacker to be able to view portions of the file; however, the file should be distorted enough that an attacker would not wish to use the encrypted file, but would rather purchase or subscribe to the decryption method for access to the original files.

Multimedia files prove to be good subjects for selective encryption, as these files tend to be very large and employ compression algorithms that concentrate critical information in small portions of their bit stream. If the critical data in certain multimedia standards is encrypted properly, the remaining information becomes useless to those without the appropriate decryptor. There are many types of compression algorithms that fit this description, such as MPEG 1, 2 and 4 video, G.723 and G.729 video, AAC audio, MP3 audio, JPEG and JPEG2000 image formats. Applying a Selective Encryption security solution to selected multimedia formats will greatly increase the protection level of important information.

The focus of the ISE project is to research and develop an algorithm for selectively encrypting the JPEG *baseline* compression image standard. The product of the research and development will be a package that will encrypt a file so that the amount of the file being encrypted is relatively small (on the order of 1-2% of the total file). The product will be delivered in a package that will include an encryptor and a decryptor for JPEG files and a testing suite. A web site will be constructed to facilitate the delivery of the product and documentation about the process. The encryptor and decryptor will encrypt and decrypt selected targets contained within JPEG files. The ISE project will employ the AES (Advanced Encryption Standard) for our Selective Encryption algorithm. This package will be made available in a purely open source form on our final web site.

In addition to the package containing the decryptor and encryptor, Team ISE will also provide a test suite available to prospective users. The test suite will be used to aid in the research, development and testing of the team's final product. The test suite will provide the functions necessary to complete this project. First, it will allow the user to preview a standard JPEG image. Second, the test suite will break down the various portions of a JPEG image and provide the ability to manipulate the data in all of the portions. Third, after altering the data in any particular file, the test suite will provide the capability to preview the encryption attempt without the benefit of compatible decryption. Forth, the suite will have the ability to decrypt an encrypted file. The decryption options will allow the user try to defeat the encryption methods. Any selective encryption scheme could be developed using a package that implemented these

features, however, the delivered test suite will only employ the AES encryption scheme chosen by the team. The test suite will be available to download from the team web site.

The final web site will be deployed on a web server provided by the Sponsor. The machine facilitating the web server will use the Linux Red Hat 9.0 operating system platform. The team will acquire a fixed IP address from the proper University of Colorado authorities and will develop a simple web site capable of delivering information to viewers about the benefits and application of Selective Encryption technology. The site will provide users the option to download and use the final software package. The site will also provide links to important information and will remain in place as long as the sponsor deems necessary.

The final software package will accomplish the complex task of selectively encrypting a JPEG baseline standard image while providing a simple user interface. Team ISE has identified three specific types of users: high-end art users, typical Internet image users, and small, low-end image users. The research and software will be tailored to these users' needs. Figure 1.1 is a flow chart showing the general logic design of the team's final product.



Figure 1.1: Conceptual Overview of ISE Software

Information regarding the user interfaces for all of the ISE products are described in the next section of this document. Following the user interface sections, the design overview for the project is presented with a high-level modular decomposition of each of the project modules and sub-modules. After the design overview, an in-depth explanation of the design and its low-level functionality for each module is presented. Immediately following the low-level design is an explanation of all of the valid file types used by the ISE products. Lastly, a summary of this document is provided, followed by a complete glossary of terms, and finally, a listing of readings directly related to this project. For a full description of the project requirements or the system architecture document, please refer to the online documentation located on the ISE web site at http://128.138.75.184. This document outlines the full design of the ISE project and will be referred to as a "road map" for development during the implementation process.

# **2. USER INTERFACE**

During the course of this project, Team ISE will develop three separate products: the ISE class production code, the JPEG Manipulator test suite and the Team ISE web site. Each of these products will have a different user interface. This section outlines the design of each of the three final products.

# 2.1. ISE Class Production Code User Interface

The user interface to the ISE production code is a series of C++ classes designed for use in application development. A software developer can create a new instance of the jpeg\_ise type, input the pertinent information, and then make calls to the class APIs to encrypt and decrypt images. This section defines how a programmer may employ this functionality.

# 2.1.1. Instantiation of the JPEG ISE Class

To create a new instance of this class, the user may choose from three different constructors. The default constructor allows the user to create the object without having to pass any arguments. An example of using the default constructor is shown below:

// Creating a JPEG ISE object with default ctor
jpeg\_ise MyEncryptionClass;

In addition to the default constructor, the jpeg\_ise class also provides two overloaded constructors for passing the Key, and one (or optionally both) of the file names. If third parameters are not passed, a default name will be created based upon the input file name. An example of using the two overloaded constructors is shown below:

# 2.1.2. Usage of the JPEG ISE Class Methods

Once an instance of the class has been declared, the user can then begin to make calls to the various functions. There are two major uses of the class: encrypting and decrypting. This section outlines the steps necessary to complete both of these tasks.

# 2.1.2.1. Encrypting with the JPEG ISE Class

There are number of steps required before the user can call the encrypt\_file() method to encrypt a JPEG image. The programmer is required to set up both a key and the JPEG input file name. If desired, the user may also specify the file name for the intermediate ISE file created during the encryption process, but if none is specified, a default file name will be created based upon the original JPEG input file name. The following is an example of one way a user can encrypt using the jpeg\_ise class:

```
// Objects needed to encrypt a JPEG file
jpeg_ise MyEncryptClass;
char JPEG_File_Name [256] = "C:\\MyImage.jpg";
char ISE_File_Name [256] = "C:\\MyImage.ise";
// The Key can be up to 320 chars long
char My_Key_Password [320] = "MyPassword123";
// Set the file names and key information
MyEncryptClass.set_input_file_name(JPEG_File_Name);
MyEncryptClass.set_ise_file_name(ISE_File_Name);
MyEncryptClass.set_key(My_Key_Password);
// Encrypt the JPEG file
MyEncryptClass.encrypt_file();
```

**Note:** This is one way in which a programmer can use the encrypt methods, but there are several ways to accomplish this task from using this class. We will talk about the design of all of the methods in section 5.1 of this document.

#### **2.1.2.2. Decrypting with the JPEG ISE Class**

The decryption process is virtually identical to the encryption process, with a few subtle differences. There are still a number of steps required before the user can call the decrypt\_file() method to decrypt an ISE file. The programmer is required to set up both a key and the ISE intermediate file name. If desired, the user may also specify the name for the decrypted file, but if none is specified, a default file name will be created based upon the ISE file name. The following is an example of one way a user can decrypt using the jpeg\_ise class:

```
// Objects needed to encrypt a JPEG file
jpeg_ise MyEncryptClass;
char Output_File_Name [256] = "C:\\MyDecryptedImage.jpg";
char ISE_File_Name [256] = "C:\\MyImage.ise";
// The Key can be up to 320 chars long
// The Key must match the key used to encrypt the file
char My_Key_Password [320] = "MyPassword123";
// Set the file names and key information
MyEncryptClass.set_ise_file_name(ISE_File_Name);
MyEncryptClass.set_output_file_name(Output_File_Name);
MyEncryptClass.set_key(My_Key_Password);
// Decrypt the ISE file
MyEncryptClass.encrypt_file();
```

**Note:** As with encrypt, this is only one way in which a programmer can use the decrypt methods, but there are several ways to accomplish this task using this class. We will talk about the design of all of the methods in section 5.1 of this document.

# 2.2. The JPEG Manipulator User Interface

The JPEG Manipulator's user interface is outlined within this section. The Manipulator provides an easy-to-use graphical user interface. The GUI allows the user to view all of the various pieces of a JPEG image with a familiar Windows style application interface. The following is a description of the GUI interface that will be developed for the JPEG Manipulator.

# 2.2.1. JPEG Manipulator Invocation

The Manipulator will come prepackaged with a fully functional installation script to provide ease of use for any user to quickly install. This package will also include an uninstaller script, to provide the user with the ability to fully remove all data installed with the program, should the need arise. Once the program has been installed by the user, they can then invoke the program from their start menu by choosing:

### Start -> Programs -> ISE -> JPEG -> JPEG Manipulator

Once the user has invoked the application, it will open to the default main screen, which is the "Console" tab within the application. This will appear as a standard Windows user interface as shown below:

ISE JPEG Manipulator - Version 1.0.1	_ [D] ×
Ele Help	
Console Original Picture Manipulated Picture	
Project Ele Information Encoded Data Quantizer Table Hoffman Tables 2 Annihoston Data Mico	
Project File Pubr	
Project Notes:	New Project
	Load Picture Save Picture
×	Update Picture

Figure 2.2.1: The JPEG Manipulator on entry into the application.

# 2.2.2. The Manipulator's Menu Bar

The Manipulator will provide a standard menu bar with the application. This menu bar will consist of a File, an Edit and a Help menu.

# 2.2.2.1. The File menu

The File menu will provide the user with a number of standard functions for interacting with the application. Figure 2.2.2.1 shows an example of the File menu. The functionality of this menu is described below.

ISE JPEG Manip	ulator - Ve	ersion 1.0.1
File Help		
New Project Open Project Save Project	Picture	Manipulated Picture
Open Picture		
Update Picture		
Exit		

Figure 2.2.2.1: Illustration of the File menu.

# 2.2.2.1.1. New Project option

This option allows the user to create a new selective encryption project within the Manipulator.

# 2.2.2.1.2. Open Project option

This option allows the user to open a previously created selective encryption project within the Manipulator.

# 2.2.2.1.3. Save Project option

This option allows the user to save the current selective encryption project that is currently in progress within the Manipulator.

# 2.2.2.1.4. Open Picture option

This option allows the user to open a new original JPEG image within the Manipulator.

# 2.2.2.1.5. Update Picture option

This option allows the user to create a manipulated image JPEG image within the Manipulator.

# 2.2.2.1.6. Exit option

This option allows the user to quickly and easily exit the Manipulator. If there is an unsaved project open, then the user will be prompted to save before the application has exited.

### 2.2.2.2. The Edit menu

The Edit menu will provide the user with the ability to do common editing functions such as Cut, Copy, and Paste. The functionality of this menu is described below.

# 2.2.2.1. Cut option

This option allows the user to cut selected text from any of the TextBox fields within the manipulator. The cut text will be copied to the system clipboard for future retrieval.

# **2.2.2.2.2. Copy Option**

This option allows the user to copy selected text from any of the TextBox fields within the manipulator. The copied text will be copied to the system clipboard for future retrieval.

### **2.2.2.3.** Paste Option

This option allows the user to paste the most recently copied or cut text from the system clipboard to the selected text box.

### 2.2.2.3. The Help menu

The Help menu will provide the user with a Help option and an About option. Figure 2.2.2.2 shows an example of the Help menu. The functionality of this menu is described below.

ISE	ISE JPEG Manipulator - Version 1.0.1			
File }	Help			
foD	Help About	hal Picture	Manipulated Picture	
		_		

Figure 2.2.2.2: Illustration of the Help menu.

# 2.2.2.3.1. Help option

This option allows the user to enter the self-help portion of the program. This program will provide the user with a user-guide for the Manipulator and other information about using this program.

# 2.2.2.3.2. About option

This option allows the user to view the About screen included with the Manipulator. The About screen will display information about the creators, version and other minor information about the program.

# 2.2.3. The Manipulator's Console Tab

The Console Tab of the Manipulator consists of several different controls. The Console Tab is the main work area within the application and provides access to all of the data stored for a particular JPEG image. This access is provided via a series of Data Tabs located on the bottom half of the Console Tab. In addition, the Console Tab provides two picture box controls to view both an original image and a manipulated image, for a side-by-side comparison of the two pictures. The original picture is located on the left side and the manipulated picture is located on the right. The figure below illustrates an example of the Console Tab:



Figure 2.2.3: Example of the Console Tab

# 2.2.3.1. Project Tab

This tab allows the user to access project data for the currently loaded project. From here, the user can create a new project, save a project, load a project, load a picture, save a picture, create a manipulated picture or enter in notes about the particular project. The figure below shows an example of this data tab:

Project File Informa	tion Encoded Data Quantizer Table Huffman Tables 1 Huffman Tables 2 Application Data Mis	c	
Proiect File Path:	C:\TestProject.SEP		New Project
Proiect Notes:	This is a test of the Project notes field in the JPEG Manipulator.	~	Load Project
	1. Altering the Huffman table drastically distorts the image.		Save Project
			Load Picture
			Save Picture
		×	Update Picture

Figure 2.2.3.1: Example of the Project Tab

# 2.2.3.2. File Information Tab

This tab allows the user access to the File Information data for the currently loaded pictures. From here, the user can specify the manipulated picture name and path, view the original picture name and path, view the file size of the original image and view file comments included with the JPEG image. The figure below shows an example of this data tab:

Project File Information	n Encoded Data Quantizer Table Huffman Tables 1 Huffman Tables 2 Application Data Misc
Original File Name:	C:\photo1.jpg
Manipulated File:	C:\photo1changed0.jpg
File Size:	7687 bytes
File Comments:	

Figure 2.2.3.2: Example of the File Information Tab.

# 2.2.3.3. Encoded Data Tab

This tab allows the user access to the Encoded Data information for the currently loaded JPEG image. From here, the user can view and manipulate the first 10,000 bytes of the encoded data frame and the Scan Header information for the JPEG file. All of the data under this tab is displayed in hexadecimal format. The figure below shows an example of this data tab:

Encoded Data:	Scan Header:	03 01 00 02 11 03 11 00 3f 00
9d13778214916c40aae606e20d15l c2b6d746672e01156a62dd841735	odbc01ec35d57b2cb95d7e878d d9153e76779ae288e26d0bb <del>f11</del> Th	D5a1954d8cb5c41c8a6ef2dda632466ac9f6e637508512ea37d083f0514c92d34 5527_02055_04600_040k060000_067729f7d6ee249a5698aaa96473584157195d is is the entropy encoded data stream.
Oriainal Encoded Data:	Original Header:	

Figure 2.2.3.3: Example of the Encoded Data Tab.

# 2.2.3.4. Quantizer Table Tab

This tab allows the user access to the Quantizer Frame data for the currently loaded JPEG image. From here, the user can view and manipulate the Quantizer tables and restore any manipulated table to their original values. All of the data under this tab is displayed in hexadecimal format. The figure below shows an example of this data tab:

Project File Information Encoded Data Quantizer Table Huffman Table	les 1 Huffman Tables 2 Application Data Misc
Quantizer 1: ffdb         00 08 06 06 06 06 06 06 08 06 06 08 0c 08 07 08 0c 0e 0a           08 08 0a 0e 10 0d 0d 0e 0d 0d 10 11 0c 0e 0d 0d 0e	Quantizer 2:
Original 1:	Original 2:
Quantizer 3:	Quantizer 4:
Original 3:	Original 4:

Figure 2.2.3.4: Example of the Quantizer Table Tab.

# 2.2.3.5. Huffman Table Tabs

This tab allows the user access to the Huffman Frame data for the currently loaded JPEG image. From here, the user can view and manipulate the Huffman tables and restore any manipulated table to their original values. Because there are up to eight Huffman tables allowed in a JPEG file, we require two tabs to present all of the data. All of the data under this tab is displayed in hexadecimal format. The figure below shows an example of one of the Huffman tabs:

Project File Information Encoded Data Quantizer Table Huffman Ta	es 1 Huffman Tables 2 Application Data Misc
Huffman 1: ffc0 08 00 e1 00 e1 03 01 22 00 02 11 01 03 11 01	Huffman 2: ffc4 00 00 00 00 00 01 11 e9 ea eb ec ed ee ef f0 f1 f2 f3 f4 f5 f6 f7 f8 f9 fa fb fc fd fe ff
Original 1:	Original 2: ffc4 00 00 01 05 01 01 00 00 00 00 00 00 00 00 00 00 00
Huffman 3:	Huffman 4:
Original 3:	Original 4:

Figure 2.2.3.5: Example of a Huffman Table Tab.

# **2.2.3.6.** Application Data Tab

This tab allows the user access to the Application Data for the currently loaded JPEG image. From here, the user can view and manipulate the Application Data contained within the image, even though this data will not affect the visual display of the JPEG image. All of the data under this tab is displayed in hexadecimal format. The figure below shows an example of the Application Data Tab:

Project File Information Encoded Data Quantizer Table Huffmar	n Tables 1 Huffman Tables 2 Application Data Misc
App Data 1: ffe0 4a 46 49 46 00 01 02 00 00 64 00 64 00 00	App Data 2: ffec 44 75 63 6b 79 00 01 00 04 00 00 00 32 00 00
App Data 3: ffee 41 64 6f 62 65 00 64 c0 00 00 00 01	App Data 4:
App Data 5:	App Data 6:
App Data 7:	App Data 8:
App Data 9:	App Data 10:

Figure 2.2.3.6: Example of the Application Data Tab.

# 2.2.3.7. Misc Data Tab

This tab allows the user access to the all other JPEG file data for the currently loaded JPEG image. From here, the user can view and manipulate the data for a number of data frames including: the Restart Interval, the Number of Lines, the Expand Image, the Restart Mod 8 and the data for the Hierarchical Progression. In addition to this data, the user can view any errors encountered during the use of the program. All of the data under this tab is displayed in hexadecimal format, except for the Program Errors data. The figure below shows an example of the Miscellaneous Data Tab:

Project File Information Encoded Data Quantizer	Fable   Huffman Tables 1   Huffman Tables 2   Application Data   Misc
Restart Interval:	Restart Modulo 8 occured at. Hierarchial Progression:
Expand Image: Program Errors:	

Figure 2.2.3.7: Example of the Misc Data Tab.

This sums up all of the data tabs contained under the JPEG Manipulator Console Tab. Note that all of the JPEG image data is contained within the tabs are located on the Console Tab. In addition to the Console Tab, there are two other tabs: the Original Picture Tab and the Manipulated Picture Tab. These additional tabs allow the user to view each picture in a larger form.

# 2.2.4. The Original Picture Tab

The Original Picture tab allows the user to view the currently loaded original JPEG image in a larger form. This tab is located directly to the right of the Console tab. The figure below illustrates an example of the Original Picture Tab:



Figure 2.2.4: Example of the Original Picture Tab.

# 2.2.5. The Manipulated Picture Tab

The Manipulated Picture tab allows the user to view the currently loaded manipulated JPEG image in a larger form. This tab is located directly to the right of the Original Picture tab. The figure below illustrates an example of the Manipulated Picture Tab:



Figure 2.2.5: Example of the Manipulated Picture Tab.

# 2.3. Team ISE Web Site User Interface

This section outlines the user interface of the Team ISE web site. The web site will be a very simple construction with a home page directing users to previews, final product code, all final documentation and JPEG Manipulator test suite.

# 2.3.1. ISE Web Site Invocation

Once the project has been completed, all pertinent information will reside on the ISE web pages. This web site will be located on a server maintained by the sponsor at the University of Colorado at Boulder. The IP address of this web site is: 128.138.75.184. A user can access this site by going to their web browser and entering the following in their browser address bar:

### http://128.138.75.184/

# 2.3.2. ISE Web Site Navigation

The user will have access to a menu at the top of the page. The buttons in the menu will redirect the user to the different sections of the web site. Users can jump directly to a specific document by using the menu's pull down menus. Figure 2.3.2.1 displays an image of the web site.



Figure 2.3.2.1: Screenshot of ISE Web Page

The Documentation button directs the user to a page where they can download the PDF version of any documents produced by Team ISE. The user can download a desired document by clicking on the document's button. Figure 2.3.2.2 displays an image of the document download page.



Figure 2.3.2.2: Screenshot of Documentation Page

The user can access the download page by clicking on the Download button in the menu. Upon clicking this button, the user will be directed to the download page where they can download the production code, the Manipulator, and the Microsoft .NET Framework version 1.1. The user can download these products by clicking on the buttons on the download page. Figure 2.3.2.3 displays an image of the download page.



The user can access relevant links by clicking on the Links button in the menu bar. The Links button will direct the user to a page containing links to web pages relevant to the ISE project. The user can visit these pages by clicking on the buttons on the links page. These links will redirect the user to other web pages. Figure 2.3.2.4 displays an image of the links page.

💁 ISE Related Links - Microsoft Internet Explorer	
Ele Edit Yew Favorites Look	Beb 🥂
🕒 Back + 🕞 - 💌 😰 🦿	🔓 🔎 Search 👷 Favorites 🜒 Media 🥝 🔗 - 🦕 🖾 - 🛄 🍪
Address 👔 http://128.138.75.184/Lin	is.html 💌 🛃 Go 🛛 Links 🎢 📆 🗸
www.ijg.org/	▲ Clicking on this button will open the Independent JPEG Group's (IJG) site in a new browser window. IJG writes and distributes a free JPEG image compression library.
rijndael algo	Clicking on this button will open a web page containing information on the Rijndael block cipher in a new browser window. This block cipher is used in the C++ class for selectively encrypting JPEG Baseline Compression files. Implementations of the Rijndael AES block cipher can be downloaded from this page.
Project Sponsor	Clicking on this button will open project sponsor Dr. Tom Lookabaugh's site in a new browser window.
Univ of Colorado	Clicking on this button will open the University of Colorado home page in a new browser window.
CU Dept of CS	Clicking on this button will open the University of Colorado Computer Science Department's home page in a new browser window.
Dept of JTD	Clicking on this button will open the University of Colorado Interdisciplinary Telecommunications Department's home page in a new browser window.
This project was done by University of Colorado students under the supervision of the Computer Science Department.	
e) Done	💿 Internet 🖉

Figure 2.3.2.4: Screenshot of Links Page

The user can always return to the ISE home page, displayed in Figure 2.3.2.1, by clicking on the Home button in the menu bar. The user can access information on the Project Sponsor, Tom Lookabaugh, by clicking on the Project Sponsor button, and can read the project proposal by clicking on the Project Proposal button.

# **3. DESIGN OVERVIEW**

This section of the document provides an overview of the design of all of Team ISE's products. This overview is given as a high-level design scheme for the ISE class production code, the JPEG Manipulator and the Team ISE web site.

# 3.1. High-Level Modular Decomposition

Team ISE's project breaks down into several high-level modules, each fulfilling a specific purpose for the project. The project consists of 4 main modules: the Encryptor, the Decryptor, the JPEG Manipulator test suite and the Team ISE web site. A high-level modular decomposition of Team ISE's software project is presented in the following figure:



Figure 3.1: High level modular decomposition of the ISE project.

# 3.2. ISE Class Production Code Modules

The ISE class production code contains both the Encryptor and the Decryptor. Both of these modules are outlined below.

# **3.2.1. ISE Encryptor Module**

The Encryptor will be invoked as an API, as described is section 2.1.2 in this design document. The purpose of this module is to selectively encrypt a JPEG image, based upon the algorithm developed by Team ISE. The Encryptor is called by invoking one of the encrypt\_file() methods found within the JPEG ISE class. The Encryptor will be included along with the Decryptor in the ISE class production code.

# **3.2.2. ISE Decryptor Module**

The Decrpytor will be invoked as an API, as described is section 2.1.2 in this design document. The purpose of this module is to decrypt a selectively encrypted JPEG image, based upon the algorithm developed by Team ISE. The Decryptor is called by invoking

one of the decrypt\_file() methods found within the JPEG ISE class. The Decryptor will be included along with the Encryptor in the ISE class production code.

# 3.3. JPEG Manipulator Test Suite Module

The design of the Manipulator application breaks down into six high-level sub-modules that in turn break down into a series of supporting object methods. These sub-modules are defined as:

- 1. Standard Windows Form Application Methods.
- 2. Manipulator Graphical Interface Methods.
- 3. Manipulator Common Methods.
- 4. Methods to Convert from Binary to ASCII.
- 5. Methods to Convert from ASCII to Binary.
- 6. Methods to Encrypt and Decrypt.

The following is a brief explanation of each of the Manipulator sub-modules. For a detailed description of each of the individual methods included in these sub-modules, refer to section 4 of this document.

# **3.3.1. Standard Windows Form Application Methods**

This sub-module contains the methods necessary to support the Windows form instantiation and disposal, the main entry point of the application and any other functionality necessary to operate in the Windows environment.

# 3.3.2. Manipulator Graphical Interface Methods

This sub-module contains the methods necessary to support the Windows form operations and resolve events generated by Windows form components. Specifically, these methods will execute events like button clicks or provide menu option functionality.

# 3.3.3. Manipulator Common Methods

This sub-module contains the methods related to the core functionality of the Manipulator. These methods will be the engine for the Manipulator, implementing the low-level functionality for loading files, writing files, processing data and performing common tasks within the application.

# 3.3.4. Methods to Convert from Binary to ASCII

This sub-module consists of methods written to convert the byte values found within a JPEG file to ASCII characters between '0' and 'F' that represents the binary data value in hexadecimal format.

# 3.3.5. Methods to Convert from ASCII to Binary

This sub-module consists of methods written to convert the ASCII character value currently loaded in the Manipulator back to binary values. These functions are the reverse of the functions found in the previous section.

# **3.3.6.** Methods to Encrypt and Decrypt

This sub-module consists of methods responsible for providing all of the encryption and decryption functionality for the Manipulator.

# 3.4. Team ISE Web Site Module

The web site will serve as the distributor for Team ISE's software packages, research materials and information pertaining to the final products. The site will provide links to all documentation created by the team for the software packages, the research behind implementation, sponsor information and all documents related to the ISE project.

# 4. DESIGN

This section of the document provides an in-depth view of design of the ISE class production code, the JPEG Manipulator and the ISE web site.

# 4.1. ISE Class Production Code Design

The ISE class production code will be implemented in C++ and will consist of two classes and several methods which will be outlined within this section. The ISE class is an abstract base class from which other selective encryption classes are derived. The ISE class by itself is never instantiated but is inherited by other classes and is used as an interface to define the inheriting classes. The class will implement non-file-type-specific methods and will initialize class data members.

The JPEG ISE class will inherit the ISE class and all of its non-file-type-specific methods and data members. The class will implement the selective encryption and decryption methods inherited from the ISE class, specifically designed to selectively enrypt standard baseline JPEG images.

This section details the design of the ISE class production code, including a full description of all methods and data members for both the ISE and JPEG ISE classes. The algorithm designed to selectively encrypt and decrypt JPEG files using the Huffman table information is also included in this section.

# 4.1.1. ISE Class Invocation

The ISE class will provide a number of different user interfaces that developers will use to employ the ISE class. The ISE class can be constructed in one of three ways:

# 4.1.1.1. protected ise()

# **Default Constructor**

Pre-conditions: None. Post-conditions: A default ISE object is created. Parameters: None. Return values: Constructor, no return type.

Description:

An ISE object can be constructed with no arguments using the default constructor on a protected level, and is not intended to be used explicitly.

# 4.1.1.2. ise(char \* key, char \* input\_file\_name, char \* ise\_file\_name = NULL)

# **Encryption Overloaded Constructor**

Pre-conditions:

The **key** must be a pointer to a character string with a maximum length of 320 characters.

Post-conditions:

An ISE object is created containing the specified data members.

Parameters:

The first argument is a pointer to the encryption key. The second argument is the name and path of the input file to be encrypted. The third argument is the ISE file name for the file generated by encryption.

Return values:

Constructor, no return type.

Description:

An ISE object may also be constructed with the data necessary to encrypt a file. This overloaded constructor only requires that the first two arguments be provided. The third argument is optional and will be set to a default value based upon the input file if it is not specified.

# 4.1.1.3. ise(char \* key, char \* ise\_file\_name, char \* output\_file\_name = NULL)

### **Decryption Overloaded Constructor**

**Pre-conditions:** 

The **key** must be a pointer to a character string with a maximum length of 320 characters.

**Post-conditions:** 

An ISE object is created containing the specified data members.

Parameters:

The first argument is a pointer to the encryption key. The second argument is the name and path of the ISE file to be decrypted. The third argument is the output file name for the file generated by decryption.

Return values:

Constructor, no return type.

Description:

An ISE object may also be constructed with the data necessary to decrypt an ISE file. This overloaded constructor only requires that the first two arguments be provided. The third argument is optional and will be set to a default value based upon the input file if it is not specified.

### 4.1.2. Public Methods of the ISE Class

There are a number of public interface exposed by the ISE class to the developer. These are the methods used to by the developer to perform the major functions of this class. These public interfaces are as follows:

### **4.1.2.1. virtual int encrypt\_file()**

Pre-conditions:

The **input\_file\_name** and **key** must be set using either the encryption overloaded constructor or the **set\_input\_file\_name(char\* name)** and **set\_key(char\* key)** functions prior to calling this method.

**Post-conditions:** 

The **ise\_file\_name** file will contain the selectively encrypted file data.

Parameters: None.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The encrypt\_file method will take a file and selectively encrypt the pertinent data within the file. This is a virtual method and must be implemented in the class inheriting from ISE.

# 4.1.2.2. virtual int encrypt\_file(char \* key, char \* input\_file\_name, char \* ise\_file\_name = NULL)

Pre-conditions: None.

**Post-conditions:** 

An encrypted file will be created with the name and path specified by the value within the **ise\_file\_name** data member. If this data member is NULL, then a default file name will be created based upon the **input\_file\_name** data member. The **key**, **input\_file\_name** and **ise\_file\_name** data members within the class will be set to parameter values.

Parameters:

The first argument is a pointer to the encryption key. The second argument is the name and path of the input file to be encrypted. The third argument is the ISE file name for the file generated by encryption.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The encrypt\_file method will take a file and selectively encrypt the pertinent data within the file. This is a virtual method and must be implemented in the class inheriting from ISE.

### **4.1.2.3. virtual int decrypt\_file()**

Pre-conditions:

The **ise\_file\_name** and **key** must be set using either the decryption overloaded constructor or the **set\_ise\_file\_name(char\* name)** and **set\_key(char\* key)** functions prior to calling this method. The **key** used in this method must be the same as the one used to encrypt the ISE file.

Post-conditions:

The **output\_file\_name** file will contain the selectively decrypted file data. Parameters: None. Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

### Description:

The decrypt method will take an instance of an ISE file and selectively decrypt the correct portion(s) of the file. This is a virtual method and must be implemented in the class inheriting from ISE.

# 4.1.2.4. virtual int decrypt\_file(char \* key, char \* ise\_file\_name, char \* output\_file\_name = NULL)

**Pre-conditions:** 

The **ise\_file\_name** and **key** must be set using either the decryption overloaded constructor or the **set\_ise\_file\_name(char\* name)** and **set\_key(char\* key)** functions prior to calling this method. The **key** used in this method must be the same as the one used to encrypt the ISE file.

### Post-conditions:

The **output\_file\_name** file will contain the selectively decrypted file data.

Parameters:

The first argument is a pointer to the encryption key. The second argument is the name and path of the ISE file to be decrypted. The third argument is the file name for the file generated by the decryption process.

#### Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

### Description:

The decrypt method will take an instance of an ISE file and selectively decrypt the correct portion of the file. This is a virtual method and must be implemented in the class inheriting from ISE.

### **4.1.2.5. int set\_key(char** \* key)

Pre-conditions:

The **key** must point to a character string with a maximum length of 320 characters.

Post-conditions:

The **key** will be set using the new string specified. Any previous information in **key** will be lost.

Parameters:

The only argument to this method is a pointer to a character string containing the key information for either encryption or decryption.

### Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The method will use the specified character string to create a valid key to be used by the encryption or decryption methods. This method must be called prior to calling **encrypt\_file()** or **decrypt\_file()** if the default constructor is used to create the ISE object.

### 4.1.2.6. int set\_input\_file\_name(char \* name)

Pre-conditions:

The **name** must be a pointer to a valid file type supported by ISE selective encryption.

Post-conditions:

The **input\_file\_name** will be set using the new string specified. Any previous data in **input\_file\_name** will be lost.

#### Parameters:

The only argument to this method is a pointer to a character string containing the **input\_file\_name**, specifying the file to be selectively encrypted.

#### Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

This method is used to set the **input\_file\_name**. The method must be called prior to the encryption method if the default constructor was used to create the ISE object.

### 4.1.2.7. int set\_ise\_file\_name(char \* name)

Pre-conditions:

The **name** must be a pointer to a valid ISE file.

Post-conditions:

The **ise\_file\_name** will be set using the new string specified. Any previous data in **ise\_file\_name** will be lost.

Parameters:

The only argument to this method is a pointer to a character string containing the **ise\_file\_name**, specifying the file to be selectively decrypted or the resulting selectively encrypted file.

#### Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

This method is used to set the **ise\_file\_name**. This method must be called prior to calling the decryption method if the default constructor was used to create the ISE object.

### 4.1.2.8. virtual int set\_output\_file\_name(char \* name)

### Pre-conditions:

The **name** must be a pointer to a valid file type supported by ISE selective encryption.

Post-conditions:

The **output\_file\_name** will be set using the new string specified. Any previous **output\_file\_name** in the object will be lost.

Parameters:

The only argument to this method is a pointer to a character string containing the **output\_file\_name**, specifying the file to be created during selective decryption. Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

This method is used to set the **output\_file\_name**. If the **output\_file\_name** is not specified by this method or the decrypt overloaded constructor, the program will automatically create a name based on the **ise\_file\_name**. The created name will be one that does not exist in the current directory. For example the string decrypted might be concatenated to the end of the **ise\_file\_name**. The function must be implemented in the class inheriting the ISE base class.

# 4.1.2.9. char \* get\_input\_file\_name()

Pre-conditions: None. Post-conditions: None.

Parameters: None.

Return values:

The method will return the input\_file\_name character string. If the

**input\_file\_name** is not set, the method will return an empty string. The string the returned pointer points to is owned by the class. The user need not worry about deallocating this string.

Description:

This is the accessor method for the input file name.

# 4.1.2.10. char \* get\_ise\_file\_name()

- Pre-conditions: None.
- Post-conditions: None

Parameters: None.

Return values:

The method will return the **ise\_file\_name** character string. If the **ise\_file\_name** is not set, the method will return an empty string. The string the returned pointer points to is owned by the class. The user need not worry about deallocating this string.

Description:

This is the accessor method for the **ise\_file\_name.** 

# 4.1.2.11. char \* get\_output\_file\_name()

Pre-conditions:None.Post-conditions:None.Parameters:None.

Return values:

The method will return the **output\_file\_name** character string. If the **output\_file\_name** is not set, the method will return an empty string. The string the returned pointer points to is owned by the class. The user need not worry about deallocating this string.

Description:

This is the accessor method for the output file name.

### **4.1.3.** Protected Methods of the ISE Class

In addition to the public interfaces, this ISE class will also contain a number of protected methods that will only be used by the classes. These methods are specific to the low-level functionality of the class and thus will not be exposed to users. These private methods are as follows:

### 4.1.3.1. int get\_ise\_file\_type(char \* name)

Pre-conditions:

The **name** must be a pointer to a valid ISE file.

Post-conditions: None

Parameters:

The only argument for this method is a pointer to a character string indicating the name of a valid ISE file.

#### Return values:

The function will return an integer indicating the type of the original file from which the specified ISE file was created.

0 will indicate an unknown or unimplemented file type.

1 will indicate a jpeg file.

The return values may be extended to accommodate other file types.

Description:

This method will return an integer corresponding to the original file type of an encrypted ISE file.

### 4.1.3.2. int make\_ise\_file\_name()

**Pre-conditions:** 

The user of the class has previously set the **input\_file\_name**. Post-conditions:

The **ise\_file\_name** data member points to a string with a file name and file path, based upon the string pointed to by the **input\_file\_name**.

Parameters: None.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The file name and path created will be the same as the string pointed to by the **input\_file\_name** data member, except that the extension of the file will be changed to .ise. If this file already exists, then a 0 will be added on to the end of the file name, just before the extension. If this file already exists, we will keep incrementing this number and checking, until the new file name does not previously exist.

### 4.1.3.3. int make\_output\_file\_name()

Pre-conditions:

The user of the class has previously set the **ise\_file\_name**. Post-conditions:

The **output\_file\_name** data member points to a string with a file name and file path, based upon the string pointed to by the **ise\_file\_name**.

Parameters: None.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The file name and path created will be the same as the string pointed to by the **ise\_file\_name** data member, except that the extension of the file will be changed to .jpg. If this file already exists, then a 0 will be added on to the end of the file name, just before the extension. If this file already exists, we will keep incrementing this number and checking, until the new file name does not previously exist.

# **4.1.4. Data Members of the ISE Class**

# 4.1.4.1. char \* input\_file\_name

This data member defines the file to be encrypted.

### 4.1.4.2. char \* ise\_file\_name

This data member defines the ISE file created after encryption.

### 4.1.4.3. char \* output\_file\_name

This data member defines the file created after decryption.

### 4.1.4.4. char \* key

This data member defines the key to be used in both encryption and decryption.

### **4.1.5. JPEG ISE Class Invocation**

The JPEG ISE class will provide a number of different user interfaces that developers will use to employ the JPEG ISE class. The JPEG ISE class can be constructed in one of three ways:

# 4.1.5.1. protected jpeg\_ise()

### **Default Constructor** Pre-conditions: None.

Pre-conditions: Post-conditions:

A default JPEG ISE object instance is created.

Parameters: None.

Return values:

Constructor, no return type.

Description:

A JPEG ISE object can be constructed with no arguments under the default constructor on a protected level, and is not intended to be used explicitly.

# 4.1.5.2. jpeg\_ise (char \* key, char \* input\_file\_name, char \* ise\_file\_name = NULL)

# **Encryption Overloaded Constructor**

Pre-conditions:

The **key** must be a pointer to a character string with a maximum length of 320 characters.

Post-conditions:

A JPEG ISE object is created containing the specified data members.

Parameters:

The first argument is a pointer to the encryption key. The second argument is the name and path of the input JPEG file to be encrypted. The third argument is the ISE file name for the file generated by encryption.

### Return values:

Constructor, no return type.

Description:

A JPEG ISE object may also be constructed with the data necessary to encrypt a JPEG file. This overloaded constructor only requires that the first two arguments be provided. The third argument is optional and will be set to a default value based upon the input JPEG file if it is not specified.

# 4.1.5.3. jpeg\_ise(char \* key, char \* ise\_file\_name, char \* output\_file\_name = NULL )

# **Decryption Overloaded Constructor**

Pre-conditions:

The **key** must be a pointer to a character string with a maximum length of 320 characters.

Post-conditions:

A JPEG ISE object is created containing the specified data members. Parameters:

The first argument is a pointer to the encryption key. The second argument is the name and path of the ISE file to be decrypted. The third argument is the output file name for the file generated by decryption.

Return values:

Constructor, no return type.

Description:

A JPEG ISE object may also be constructed with the data necessary to decrypt an ISE file. This overloaded constructor only requires that the first two arguments be provided. The third argument is optional and will be set to a default value based upon the input file if it is not specified.

### 4.1.6. Public Methods of the JPEG ISE Class

There are a number of public interface exposed by the JPEG ISE class to the developer. These are the methods used to by the developer to perform the functions of this class. Note that some of these methods will be implemented in and inherited from the ISE base class. These public interfaces are as follows:

### 4.1.6.1. int encrypt\_file()

Pre-conditions:

The **input\_file\_name** and **key** must be set using either the encryption overloaded constructor or the **set\_input\_file\_name(char\* name)** and **set\_key(char\* key)** functions prior to calling this method.

Post-conditions:

An encrypted file will be created with the name and path specified by the value within the **ise\_file\_name** data member. If this data member is NULL, then a default file name will be created based upon the **input\_file\_name** data member. Parameters: None.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The encrypt\_file method will take a standard baseline compression JPEG file and selectively encrypt the Huffman Table frames found within the file, as well as delete all application and file comment data. If the file already exists, the existing file will be overwritten. If there is not enough space, the partial file will be deleted, and an error message will be provided telling the user that there is not enough disk space. The exact algorithm used for this method is fully explained in section 4.1.9.1 of this document. A new, encrypted file will be created for this selectively encrypted JPEG image.

# 4.1.6.2. int encrypt\_file(char \* key, char \* input\_file\_name, char \* ise\_file\_name = NULL)

Pre-conditions: None.

Post-conditions:

An encrypted file will be created with the name and path specified by the value within the **ise\_file\_name** data member. If this data member is NULL, then a default file name will be created based upon the **input\_file\_name** data member.

Parameters:

An encrypted file will be created with the name and path specified by the value within the **ise\_file\_name** data member. If this data member is NULL, then a default file name will be created based upon the **input\_file\_name** data member. The **key**, **input\_file\_name** and **ise\_file\_name** data members within the class will be set to parameter values.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The encrypt\_file method will take a standard baseline compression JPEG file and selectively encrypt the Huffman Table frames found within the file, as well as delete all application and file comment data. If the file already exists, the existing file will be overwritten. If there is not enough space, the partial file will be deleted, and an error message will be provided telling the user that there is not enough disk space. The exact algorithm used for this method is fully explained in section 4.1.9.1 of this document. A new, encrypted file will be created for this selectively encrypted JPEG image.

### 4.1.6.3. int decrypt\_file()

**Pre-conditions:** 

The **ise\_file\_name** and **key** must be set using either the decryption overloaded constructor or the **set\_ise\_file\_name(char\* name)** and **set\_key(char\* key)** functions prior to calling this method. The **key** used in this method must be the same as the one used to encrypt the JPEG image.

#### **Post-conditions:**

A decrypted file will be created with the name and path specified by the value within the **output\_file\_name** data member. If this data member is NULL, then a default file name will be created based upon the **ise\_file\_name** data member. Parameters: None.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The decrypt\_file method will take a standard ISE file and selectively decrypt the Huffman Table frames found within the file. The exact algorithm used for this method is fully explained in section 4.1.9.2 of this document. A new, decrypted standard JPEG image file will be created from this ISE file. If the file already exists, the existing file will be overwritten. If there is not enough space, the partial file will be deleted, and an error message will be provided telling the user that there is not enough disk space.

# 4.1.6.4. int decrypt\_file(char \* key, char \* ise\_file\_name, char \* output\_file\_name = NULL)

Pre-conditions: None.

**Post-conditions:** 

A decrypted file will be created with the name and path specified by the value within the **output\_file\_name** data member. If this data member is NULL, then a default file name will be created based upon the **ise\_file\_name** data member. The **key**, **input\_file\_name** and **ise\_file\_name** data members within the class will be set to parameter values.

Parameters:

The first argument is a pointer to the encryption key. The second argument is the name and path of the ISE file to be decrypted. The third argument is the file name for the file generated by decryption the process.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

The decrypt\_file method will take a standard ISE file and selectively decrypt the Huffman Table frames found within the file. The exact algorithm used for this method is fully explained in section 4.1.9.2 of this document. A new, decrypted standard JPEG image file will be created from this ISE file. If the file already exists, the existing file will be overwritten. If there is not enough space, the partial file will be deleted, and an error message will be provided telling the user that there is not enough disk space.

# 4.1.6.5. int set\_key(char \* key)

Pre-conditions:

The **key** must point to a character string with a maximum length of 320 characters.

**Post-conditions:** 

The **key** will be set using the new string specified. Any previous information in **key** will be lost.

Parameters:

The only argument to this method is a pointer to a character string containing the key information for either encryption or decryption.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

Implemented in class ISE. The method will use the specified character string to create a valid key to be used by the encryption or decryption methods. This method must be called prior to calling **encrypt\_file(**) or **decrypt\_file(**) if the default constructor is used to create the JPEG ISE object.
### 4.1.6.6. int set\_input\_file\_name(char \* name)

**Pre-conditions:** 

The **name** must be a pointer to a standard baseline JPEG image file. Post-conditions:

The **input\_file\_name** will be set using the new string specified. Any previous data in **input file name** will be lost.

Parameters:

The only argument to this method is a pointer to a character string containing the **input\_file\_name**, specifying the JPEG file to be selectively encrypted.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

Implemented in class ISE. This method is used to set the **input\_file\_name**. The method must be called prior to the encryption method if the default constructor was used to create the JPEG ISE object.

#### 4.1.6.7. int set\_ise\_file\_name(char \* name)

Pre-conditions:

The **name** must be a pointer to a valid ISE file.

Post-conditions:

The **ise\_file\_name** will be set using the new string specified. Any previous data in **ise file name** will be lost.

Parameters:

The only argument to this method is a pointer to a character string containing the **ise\_file\_name**, specifying the JPEG file to be selectively decrypted.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

Implemented in class ISE. This method is used to set the **ise\_file\_name**. This method must be called prior to calling the decryption method if the default constructor was used to create the ISE object.

# 4.1.6.8. int set\_output\_file\_name(char \* name)

Pre-conditions:

The **name** must be a pointer to a standard baseline JPEG file.

Post-conditions:

The **output\_file\_name** will be set using the new string specified. Any previous data in **output\_file\_name** will be lost.

Parameters:

The only argument to this method is a pointer to a character string containing the **output\_file\_name**, specifying the JPEG file to be created during selective decryption.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

Implemented in class ISE. This method is used to set the **output\_file\_name**. If the **output\_file\_name** is not specified by this method or the decrypt overloaded constructor, the program will automatically create a name based on the **ise\_file\_name**. The created name will be one that does not exist in the current directory. For example the string "decrypted" might be concatenated to the end of the **ise\_file\_name**.

# 4.1.6.9. char \* get\_input\_file\_name()

Pre-conditions: None. Post-conditions: None.

Parameters: None.

Return values:

The method will return the input\_file\_name character string. If the

input\_file\_name is not set, the method will return an empty string.

Description:

Implemented in class ISE. This is the accessor method for the input file name.

### 4.1.6.10. char \* get\_ise\_file\_name()

Pre-conditions: None. Post-conditions: None Parameters: None.

Return values:

The method will return the **ise\_file\_name** character string. If the **ise\_file\_name** is not set, the method will return an empty string.

Description:

Implemented in class ISE. This is the accessor method for the ise\_file\_name.

### 4.1.6.11. char \* get\_output\_file\_name()

Pre-conditions: None.

Post-conditions: None. Parameters: None.

Return values:

The method will return the **output\_file\_name** character string. If the

output\_file\_name is not set, the method will return an empty string.

Description:

Implemented in class ISE. This is the accessor method for the output file name.

### 4.1.7. Protected Methods of the JPEG ISE Class

In addition to the public interfaces, this JPEG ISE class will also contain a number of private methods. These methods are specific to the low-level functionality of the class

and thus will not be exposed to users. Some of this functionality will be implemented and inherited from the in the ISE base class. These private methods are as follows:

### 4.1.7.1. int get\_ise\_file\_type(char \* name)

**Pre-conditions:** 

The **name** must be a pointer to a valid ISE file.

Post-conditions:

None

Parameters:

The only argument for this method is a pointer to a character string indicating the name of an ISE file.

Return values:

The function will return an integer indicating the type of the original file from which the specified ISE file was created. The return value will be a one to indicate a JPEG ISE file.

Description:

Implemented in class ISE. This method will return an integer corresponding to the original file type of an encrypted ISE file.

#### 4.1.7.2. int make\_ise\_file\_name()

Pre-conditions:

The user of the class has previously set the **input\_file\_name**.

Post-conditions:

The **ise\_file\_name** data member points to a string with a file name and file path, based upon the string pointed to by the **input\_file\_name**.

Parameters: None.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

Description:

Implemented in class ISE. The file name and path created will be the same as the string pointed to by the **input\_file\_name** data member, except that the extension of the file will be changed to .ise. If this file already exists, then a 0 will be added on to the end of the file name, just before the extension. If this file already exists, we will keep incrementing this number and checking, until the new file name does not previously exist.

#### **4.1.7.3.** int make\_output\_file\_name()

Pre-conditions:

The user of the class has previously set the **ise\_file\_name**.

Post-conditions:

The **output\_file\_name** data member points to a string with a file name and file path, based upon the string pointed to by the **ise\_file\_name**.

Parameters: None.

Return values:

An integer is returned indicating a success or failure.

A zero will indicate a success.

A one will indicate a failure.

#### Description:

Implemented in class ISE. The file name and path created will be the same as the string pointed to by the **ise\_file\_name** data member, except that the extension of the file will be changed to .jpg. If this file already exists, then a 0 will be added on to the end of the file name, just before the extension. If this file already exists, we will keep incrementing this number and checking, until the new file name does not previously exist.

# 4.1.8. Data Members of the JPEG ISE Class

# 4.1.8.1. char \* input\_file\_name

Inherited from ISE base class. This data member defines the standard baseline JPEG file to be encrypted.

### 4.1.8.2. char \* ise\_file\_name

Inherited from ISE base class. This data member defines the ISE file created after encryption.

## 4.1.8.3. char \* output\_file\_name

Inherited from ISE base class. This data member defines the standard baseline JPEG file to be created after decryption.

### 4.1.8.4. char \* key

Inherited from ISE base class. This data member defines the key to be used for encryption and decryption.

### 4.1.9. Algorithms Developed by Team ISE Used in the ISE Class

The research conducted by Team ISE has lead to the conclusion that the Huffman tables are the best targets for selective encryption of standard baseline JPEG images. Because we do not want to increase the size of the file after encryption, Team ISE has decided, as recommended by Professor John Black, to utilize the AES (Advanced Encryption Standard) encryption method.

# 4.1.9.1. JPEG Selective Encryption Algorithm

- 1. Write a single byte of information to the output file stream to indicate the type of this ISE encrypted file. For a JPEG ISE file the byte written will be a 1. Write a second byte indicating the version number of the ISE software used.
- 2. Read from the input file stream one byte at a time and write the information to the output file stream until a two byte frame marker value of ffe0 through ffef, fffe, or ffc0 through ffcf (hexadecimal) is found. These JPEG markers

indicate the beginning of application data, comment data, or Huffman data respectively. If the end of file is reached, proceed to step 15.

- 3. If a Huffman Marker is found, proceed to step 6.
- 4. If an application or comment marker is found, read through the input file stream without writing the information back to the output file stream, checking for a JPEG marker (any two-byte value beginning with ff).
- 5. If a Huffman marker is found, proceed to step 6, otherwise, return to step 2.
- 6. Write the unencrypted Huffman marker to the output file stream.
- 7. Put the next 16 bytes of data, or plain text, into a buffer, checking each for a non-Huffman marker. A non-Huffman marker is any other two-byte, hexadecimal value below ffc0 or above ffcf.
- 8. If a non-Huffman marker is found, proceed to step 12.
- 9. If no non-Huffman marker is found, encrypt the 16-byte block using Rijndael's blockEncrypt() method to produce the cipher text.
- 10. Write the cipher text to the output file stream.
- 11. Return to step 7.
- 12. Encrypt the last 16-byte block containing the non-Huffman marker using Rijndael's blockEncrypt() method to produce the cipher text.
- 13. Write the cipher text to the output file stream.
- 14. Return to step 2.
- 15. Exit the program successfully.

# 4.1.9.2. JPEG Selective Decryption Algorithm

- 1. Read a single byte of information from the input file. The byte will be a 1 if this is a valid ISE JPEG file. Read a second byte that indicates the version number of the ISE software used to create the file. Check version number to ensure compatibility. If not compatible, cancel operation and display error message.
- 2. Read off of the input file stream one byte at a time and write the information to the output file stream until a Huffman marker is found or the end of file has been reached. This marker indicates the beginning of the encrypted data.
- 3. If the end of file is reached, exit the program.
- 4. If a Huffman marker is found, write the two bytes to the output file stream.
- 5. Put the next 16 bytes of data from the input file stream, or cipher text, into a buffer.
- 6. Decrypt this 16-byte block using Rijndael's blockDecrypt() method to produce the plain text.
- 7. Scan the plain text for a non-Huffman marker.
- 8. Write the plain text to the output file stream.
- 9. If the plain text did not contain a non-Huffman marker, return to step 5.
- 10. If the cipher text contains a non-Huffman marker, return to step 2.

# 4.2. The JPEG Manipulator Design

The purpose of the JPEG Manipulator is to provide Team ISE with a tool for testing the data manipulation of JPEG images. The ideal application will supply an easy-to-use, graphical interface that grants the user the ability to simultaneously view both the original image and manipulated image, as well as view and update the original JPEG's data. The team has chosen Microsoft's Visual C# (pronounced cee-sharp) programming language to accomplish these tasks.

Developing the Manipulator in C# will allow the use of the .NET (pronounced dot-net) framework tools, satisfying all of functionality requirements outlined for the application<sup>1</sup>. The .NET framework will reduce the amount of components developed by the team, since .NET offers a wide variety of functionality and tools. The Manipulator should make use of the managed code features .NET offers, to effectively reduce the cost of future program maintenance.

To implement all of the functionality requirements<sup>1</sup>, the Manipulator requires that the team develop an extensive catalog of methods. As mentioned in section 3.3 of this document, these methods break down into six major sub-modules:

- 1. Standard Windows Form Application Methods.
- 2. Manipulator Graphical Interface Methods.
- 3. Manipulator Common Methods.
- 4. Methods to Convert from Binary to ASCII.
- 5. Methods to Convert from ASCII to Binary.
- 6. Methods to Encrypt and Decrypt.

This section of the design document outlines all of the functionality that will be created by Team ISE for the Manipulator. A complete list of the methods needed for the Manipulator is included in this section of the document. Function prototypes, pre-conditions, postconditions, parameter descriptions, return value information and function descriptions for each of the Manipulators methods is included here.

# 4.2.1. Standard Windows Form Application Methods

There are a number of functions that ISE Manipulator application is required to call to begin execution of the main program. In addition, the System.Windows.Form class requires constructor and dispose methods. The Visual Studio Windows Form Designer requires a constructor method, defined as InitializeComponent(). Lastly, we will add an additional method to be invoked by the Form's constructor to initialize all of the variables used by the Manipulator.

This section of the design document defines each of the functions necessary to satisfy the requirements of a standard Windows application. Each of these function prototypes, preconditions, post-conditions, parameters, return values and descriptions are provided below:

<sup>&</sup>lt;sup>1</sup> See Team ISE Requirements Specification for full listing of the ISE project requirements.

# 4.2.1.1. [STAThread] static void Main()

Pre-conditions: None.

Post-conditions:

The Windows Form has been invoked.

Parameters: None.

Return values:

Function returns void.

Description:

This function is the main entry point for a Windows based .NET application. This function calls the Application.Run(System.Windows.Form) method to invoke the main form of the application.

### 4.2.1.2. public frmMain()

Pre-conditions: None.

Post-conditions:

The frmMain Form of the application has been constructed.

Parameters: None.

Return values:

Form constructor, no return type.

Description:

This is the constructor for the frmMain Form of the application. This function will call the InitializeComponent() method and the ISEConstructor() to initialize the application.

## 4.2.1.3. private void InitializeComponent()

Pre-conditions: None.

Post-conditions:

All of the variables created by the Visual Studio .NET Form Designer have been initialized.

Parameters: None.

Return values:

Function returns void.

Description:

This function is required to be called by the Form's constructor. It initializes all of the variables and values set with the form designer at the beginning of the program execution.

### 4.2.1.4. private void ISEConstructor()

Pre-conditions: None. Post-conditions: ISE variables and initialization routines have been executed. Parameters: None. Return values:

Function returns void.

Description:

This function is used to execute all ISE initialization logic. This includes initialization routines for variables and setting defaults.

### 4.2.1.5. protected override void Dispose( bool disposing )

Pre-conditions: None.

Post-conditions:

All of the memory and resources used in the frmMain have been freed. Parameters:

TRUE to release both managed and unmanaged resources and FALSE to release only unmanaged resources.

Return values:

Function returns void.

Description:

This function is called when the application is when the current instance of the Form is destroyed. It is not required, but implementation of this method is recommended for .NET objects that require large amounts of data, to ensure that all memory allocated for the Form is freed immediately when the Form is destroyed.

#### 4.2.2. ISE Manipulator Graphical Interface Methods

There are a number of functions that ISE Manipulator application is required to implement to facilitate the use of the graphical interface objects. In .NET, usually these methods are created in the Form of "events" that can occur on the parent control of any particular Windows object, as the result of a control being invoked be the user of the application. For example, when a user click's on the left mouse button over a Button control, the Button generates an interrupt event within the parent of the Button object. For these interrupts to be processed by the parent control, each desired event for any particular object that should be implemented, must be implemented within the scope of the parent control. If an event is not implemented on the parent control and it occurs during execution, this event will be ignored. All of the event methods required for the JPEG Manipulator's graphical interface are outlined in this section of the design document.

# 4.2.2.1. private void menuOpen\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The menuOpen menu object has generated a Click event. Post-conditions:

A new original JPEG image has been loaded and displayed within the picOriginal and the picOrignalSmall PictureBox controls.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

#### Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuOpen menu object. The purpose of this menu object is to allow the user to open a new original JPEG image file within the application. This function will simply call the LoadNewPicture() function described in section 4.2.3.2 of this document.

#### 4.2.2.2. private void menuExit\_Click(object sender,

### System.EventArgs e)

**Pre-conditions:** 

The menuExit menu object has generated a Click event.

Post-conditions:

The application is terminated and exited successfully. Parameters:

The **sender** parameter is a pointer to the function calling this function.

The e parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuExit menu object. The purpose of this menu object is to allow the user to exit the application when they have finished. This function should check to see if there is any unsaved data before exiting and if so, should ask the user if they want to save the current information. Then, this function will call the Application.Exit() method to successfully exit the Windows application.

#### 4.2.2.3. private void menuAbout\_Click(object sender,

#### System.EventArgs e)

Pre-conditions:

The menuAbout menu object has generated a Click event.

Post-conditions:

The frmAbout Form has been displayed for the user to view.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuAbout menu object. The purpose of this menu object is to allow the user to view the about window to find out details about the system. This function creates a new instance of the frmAbout form and then displays it for the user.

# 4.2.2.4. private void menuNewProject\_Click(object sender, System.EventArgs e)

# Pre-conditions:

The menuNewProject menu object has generated a Click event.

Post-conditions:

A new project file has been created by the application.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuNewProject menu object. The purpose of this menu object is to allow the user to create a new project file that will allow them to store picture, note data and manipulated data of original images. This function should check to see if there is any unsaved data before creating a new project and if so, should ask the user if they want to save the current information. This function should simply call the CreateNewProject() method outlined in section 4.2.3.11 of this document.

# 4.2.2.5. private void menuOpenProject\_Click(object sender,

# System.EventArgs e)

Pre-conditions:

The menuOpenProject menu object has generated a Click event.

Post-conditions:

A previously created project file has been opened by the application and all values previously saved within the project have been reloaded into the application interface.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuOpenProject menu object. The purpose of this menu object is to allow the user to open a previously created project file. This function should check to see if there is any unsaved data before creating a new project and if so, should ask the user if they want to save the current information. The values stored in the project file will be reloaded into the application interface. This function should simply call the LoadNewProject() method outlined in section 4.2.3.9 of this document.

# 4.2.2.6. private void menuSaveProject\_Click(object sender,

# System.EventArgs e)

Pre-conditions:

The menuSaveProject menu object has generated a Click event.

Post-conditions:

This function saves the current values loaded in the Manipulator, project notes and any manipulate data values and stores them in an SEP file.

### Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuOpenProject menu object. The purpose of this menu object is to allow the user to save the current project file, including the original picture, manipulated picture and any notes included in the project. This function will simply call the SaveNewProject() function described later in this document.

# 4.2.2.7. private void txtChangedFile\_TextChanged(object sender, System.EventArgs e)

Pre-conditions:

The txtChangedFile TextBox object has generated a TextChanged event. Post-conditions:

A warning is displayed if the changed text reflects a file path that already exists. Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a TextChanged event generated by the txtChangedFile TextBox object. The purpose of this TextBox is to allow the user to specify the name and path of the file that will be created, if the user chooses to create a manipulated image. This function checks to see if the file name and path already exist, and if so, calls the ShowWarning() function (described later in this document) to display a warning to the users.

# 4.2.2.8. private void txtQuantizer1\_Click(object sender,

# System.EventArgs e)

Pre-conditions:

The txtQuantizer1 TextBox object has generated a Click event.

Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtQuantizerOriginal1 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtQuantizer1 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the first Quantizer table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtQuantizer1 TextBox (before it has been changed) into the txtQuantizerOriginal1 TextBox.

# 4.2.2.9. private void txtQuantizer2\_Click(object sender,

### System.EventArgs e)

Pre-conditions:

The txtQuantizer2 TextBox object has generated a Click event.

Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtQuantizerOriginal2 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtQuantizer2 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the second Quantizer table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtQuantizer2 TextBox (before it has been changed) into the txtQuantizerOriginal2 TextBox.

# 4.2.2.10. private void txtQuantizer3\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The txtQuantizer3 TextBox object has generated a Click event.

Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtQuantizerOriginal3 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtQuantizer3 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the third Quantizer table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtQuantizer3 TextBox (before it has been changed) into the txtQuantizerOriginal3 TextBox.

# 4.2.2.11. private void txtQuantizer4\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The txtQuantizer4 TextBox object has generated a Click event. Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtQuantizerOriginal4 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtQuantizer4 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the fourth Quantizer table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtQuantizer4 TextBox (before it has been changed) into the txtQuantizerOriginal4 TextBox.

# 4.2.2.12. private void txtHuffman1\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The txtHuffman1 TextBox object has generated a Click event. Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal1 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman1 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the first Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman1 TextBox (before it has been changed) into the txtHuffmanOriginal1 TextBox.

# 4.2.2.13. private void txtHuffman2\_Click(object sender, System.EventArgs e)

Pre-conditions:

The txtHuffman2 TextBox object has generated a Click event. Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal2 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The  $\mathbf{e}$  parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman2 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the second Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman2 TextBox (before it has been changed) into the txtHuffmanOriginal2 TextBox.

# 4.2.2.14. private void txtHuffman3\_Click(object sender,

## System.EventArgs e)

Pre-conditions:

The txtHuffman3 TextBox object has generated a Click event.

Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal3 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman3 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the third Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman3 TextBox (before it has been changed) into the txtHuffmanOriginal3 TextBox.

# 4.2.2.15. private void txtHuffman4\_Click(object sender,

# System.EventArgs e)

Pre-conditions:

The txtHuffman4 TextBox object has generated a Click event.

Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal4 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman4 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the fourth Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman4 TextBox (before it has been changed) into the txtHuffmanOriginal4 TextBox.

# 4.2.2.16. private void txtHuffman5\_Click(object sender,

# System.EventArgs e)

Pre-conditions:

The txtHuffman5 TextBox object has generated a Click event.

Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal5 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman5 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the fifth Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman5 TextBox (before it has been changed) into the txtHuffmanOriginal5 TextBox.

# 4.2.2.17. private void txtHuffman6\_Click(object sender,

### System.EventArgs e)

**Pre-conditions:** 

The txtHuffman6 TextBox object has generated a Click event.

Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal6 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman6 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the sixth Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman6 TextBox (before it has been changed) into the txtHuffmanOriginal6 TextBox.

### 4.2.2.18. private void txtHuffman7\_Click(object sender,

#### System.EventArgs e)

**Pre-conditions:** 

The txtHuffman7 TextBox object has generated a Click event. Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal7 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman7 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the seventh Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman7 TextBox (before it has been changed) into the txtHuffmanOriginal7 TextBox.

### 4.2.2.19. private void txtHuffman8\_Click(object sender,

### System.EventArgs e)

**Pre-conditions:** 

The txtHuffman8 TextBox object has generated a Click event. Post-conditions:

If this is the first time the data has been altered, the data is copied into the txtHuffmanOriginal8 TextBox.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data. Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the txtHuffman8 TextBox object. The purpose of this TextBox is to allow the user to manipulate the values in the eighth Huffman table contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the txtHuffman8 TextBox (before it has been changed) into the txtHuffmanOriginal8 TextBox.

# 4.2.2.20. private void btnRestoreQuantizer1\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The btnRestoreQuantizer1 Button object has generated a Click event. Post-conditions:

The information stored within the txtQuantizerOriginal1 (the original picture data) is copied back into the txtQuantizer1 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreQuantizer1 Button object. The purpose of this Button is to allow the user to restore the original data for this Quantizer table to the txtQuantizer1 TextBox.

# 4.2.2.21. private void btnRestoreQuantizer2\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The btnRestoreQuantizer2 Button object has generated a Click event. Post-conditions:

The information stored within the txtQuantizerOriginal2 (the original picture data) is copied back into the txtQuantizer2 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreQuantizer2 Button object. The purpose of this Button is to allow the user to restore the original data for this Quantizer table to the txtQuantizer2 TextBox.

# 4.2.2.22. private void btnRestoreQuantizer3\_Click(object sender, System.EventArgs e)

#### Pre-conditions:

The btnRestoreQuantizer3 Button object has generated a Click event. Post-conditions:

The information stored within the txtQuantizerOriginal3 (the original picture data) is copied back into the txtQuantizer3 TextBox object.

#### Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreQuantizer3 Button object. The purpose of this Button is to allow the user to restore the original data for this Quantizer table to the txtQuantizer3 TextBox.

# 4.2.2.23. private void btnRestoreQuantizer4\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnRestoreQuantizer4 Button object has generated a Click event. Post-conditions:

The information stored within the txtQuantizerOriginal4 (the original picture data) is copied back into the txtQuantizer4 TextBox object.

#### Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the

btnRestoreQuantizer4 Button object. The purpose of this Button is to allow the user to restore the original data for this Quantizer table to the txtQuantizer4 TextBox.

# 4.2.2.24. private void btnRestoreHuffman1\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnRestoreHuffman1 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal1 (the original picture data) is copied back into the txtHuffman1 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman1 Button object. The purpose of this Button is to allow the user to restore the original data for this Huffman table to the txtHuffman1 TextBox.

# 4.2.2.25. private void btnRestoreHuffman2\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnRestoreHuffman2 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal2 (the original picture data) is copied back into the txtHuffman2 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman2 Button object. The purpose of this Button is to allow the user to restore the original data for this Huffman table to the txtHuffman2 TextBox.

# 4.2.2.26. private void btnRestoreHuffman3\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The btnRestoreHuffman3 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal3 (the original picture data) is copied back into the txtHuffman3 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman3 Button object. The purpose of this Button is to allow the user to restore the original data for this Huffman table to the txtHuffman3 TextBox.

# 4.2.2.27. private void btnRestoreHuffman4\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnRestoreHuffman4 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal4 (the original picture data) is copied back into the txtHuffman4 TextBox object.

Parameters:

The sender parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman4 Button object. The purpose of this Button is to allow the user to restore the original data for this Huffman table to the txtHuffman4 TextBox.

# 4.2.2.28. private void btnRestoreHuffman5\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The btnRestoreHuffman5 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal5 (the original picture data) is copied back into the txtHuffman5 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman5 Button object. The purpose of this Button is to allow the user to restore the original data for this Huffman table to the txtHuffman5 TextBox.

# 4.2.2.29. private void btnRestoreHuffman6\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The btnRestoreHuffman6 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal6 (the original picture data) is copied back into the txtHuffman6 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman6 Button object. The purpose of this Button is to allow the user to restore the original data for this Huffman table to the txtHuffman6 TextBox.

# 4.2.2.30. private void btnRestoreHuffman7\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnRestoreHuffman7 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal7 (the original picture data) is copied back into the txtHuffman7 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman7 Button object. The purpose of this Button is to allow the user to restore the original data for this Huffman table to the txtHuffman7 TextBox.

# 4.2.2.31. private void btnRestoreHuffman8\_Click(object sender, System.EventArgs e)

**Pre-conditions:** 

The btnRestoreHuffman8 Button object has generated a Click event. Post-conditions:

The information stored within the txtHuffmanOriginal8 (the original picture data) is copied back into the txtHuffman8 TextBox object.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnRestoreHuffman8 Button object. The purpose of this button is to allow the user to restore the original data for this Huffman table to the txtHuffman8 TextBox.

# 4.2.2.32. private void btnUpdate\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnUpdate Menu Button object has generated a Click event. Post-conditions:

A changed picture has been updated within the application. Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnUpdate Menu Button object. The purpose of this Button object is to allow the user to create a new manipulated image for the user to see.

# 4.2.2.33. private void btnNew\_Click(object sender,

### System.EventArgs e)

**Pre-conditions:** 

The btnNew Menu Button object has generated a Click event.

Post-conditions:

This function clears out all data for pictures.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnNew Menu Button object. The purpose of this Button object is to allow the user to create a new project file that will allow them to store picture and note data about different images.

# 4.2.2.34. private void btnLoad\_Click(object sender,

## System.EventArgs e)

Pre-conditions:

The btnLoad Menu Button object has generated a Click event. Post-conditions:

A previously created project file has been loaded by the application. Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnLoad Menu Button object. The purpose of this Button object is to allow the user to open a previously created project file. The values stored in the project file will be reloaded into the application interface. This function will simply call the LoadNewProject() function described later in this document.

# 4.2.2.35. private void btnSave\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnSave Menu Button object has generated a Click event.

Post-conditions:

This function saves the current values loaded in the Manipulator and any project notes, if included.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnSave Menu Button object. The purpose of this Button object is to allow the user to save a project file and all current information in the application. The values stored in the project file will be reloaded into the application interface. This function will simply call the SaveNewProject() function described later in this document.

## 4.2.2.36. private void btnLoadPicture\_Click(object sender,

### System.EventArgs e)

Pre-conditions:

The btnLoadPicture Menu Button object has generated a Click event. Post-conditions:

An image file has been loaded by the application.

Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnLoadPicture Menu Button object. The purpose of this Button object is to allow the user to open an image file. The values stored in the project file will be reloaded into the application interface. This function will simply call the LoadNewProject() function described later in this document.

# 4.2.2.37. private void btnUpdatePicture\_Click(object sender, System.EventArgs e)

Pre-conditions:

The btnUpdatePicture Menu Button object has generated a Click event. Post-conditions:

A changed picture has been updated within the application. Parameters:

The **sender** parameter is a pointer to the function calling this function. The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the btnUpdatePicture Menu Button object. The purpose of this Button object is to allow the user to create a manipulated image based upon the data changed by user.

# 4.2.2.38. private void menuCut\_Click(object sender,

# System.EventArgs e)

Pre-conditions:

The menuCut menu object has generated a Click event.

Post-conditions:

Selected text has been cut from the text box and copied to the system clipboard. Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuCut menu object. The purpose of this menu object is to allow the user to cut selected text from any TextBox field within the Manipulator. The cut text is copied to the system clipboard for future retrieval.

# 4.2.2.39. private void menuCopy\_Click(object sender,

## System.EventArgs e)

Pre-conditions:

The menuCopy menu object has generated a Click event. Post-conditions:

Selected text has been copied to the system clipboard.

Parameters:

The sender parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

#### Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuCopy menu object. The purpose of this menu object is to allow the user to copy selected text from any TextBox field within the Manipulator. The text is copied to the system clipboard for future retrieval.

### 4.2.2.40. private void menuPaste\_Click(object sender,

### System.EventArgs e)

**Pre-conditions:** 

The menuPaste menu object has generated a Click event.

Post-conditions:

Most recent text on the system clipboard has been pasted to the selected TextBox within the Manipulator.

Parameters:

The **sender** parameter is a pointer to the function calling this function.

The **e** parameter is for the base class to pass event data.

Return values:

Function returns void.

Description:

This function is used to resolve a Click event generated by the menuPaste menu object. The purpose of this menu object is to allow the user to copy the most recent text from the clipboard to a selected Manipulator TextBox.

#### **4.2.3. ISE Manipulator Common Methods**

This section of the document describes the ISE Manipulator common methods. These are a collection of methods mostly called by the interface events in the Manipulator, but should be called by any method requiring any of this functionality. The prototypes and definitions of each of these methods are outlined in this section of the document.

# **4.2.3.1. private void** LoadPicture(string OriginalFilePath, string ChangedFilePath)

Pre-conditions: None.

Post-conditions:

An original JPEG image has been loaded into the picOriginal and picOriginalSmall PictureBox data members and a manipulated JPEG image has been loaded into the picChanged and picChangedSmall data members. Also, all of the data contained in the original file should be loaded into the interface to display for the user.

Parameters:

The OriginalFilePath parameter is a file path of the to the image to be loaded into the picOriginal and picOriginalSmall. The ChangedFilePath parameter is a file path of the to the image to be loaded into the picChanged and picChangedSmall.

# Return values:

Function returns void.

Description:

This method should be called if the Manipulator needs to be completely reload. This method should be used by any other function that needs to reload both images and the data into the interface. This method should check to make sure that any previous image has been closed within the picOriginal, picOriginalSmall, picChanged and picChangedSmall PictureBox controls before trying to load the new images. This function should do some error checking to make sure that these files actually exist before trying to load them. If one (or both) of the parameters does not contain a valid file name and path, then it should be ignored and an error message should be displayed in the txtError. If an image exists, yet it is too far damaged to load into the PictureBox controls, then an error message should be displayed for the user to see. If any errors occur during load time, the error should be displayed in the txtError TextBox for the user to see.

To perform this functionality, this function should call ClearInterfaceData(), to clear the interface. It should call UpdateChangedPicture() to load the picChanged picture. If a valid file doesn't exist in the ChangedFilePath parameter, then it should just load the file in the OriginalFilePath parameter. If the OriginalFilePath parameter doesn't contain a valid file, this function should call one of the ShowWarning() methods to let the user know that the OriginalFilePath is an invalid file and in that case, no data should be loaded to the interface. This function should set the txtOrginalFile data member. It should also open the original file in the picOriginal and picOriginalSmall PictureBox data members. Lastly, this function should call LoadPictureData() for the original file to load all of the data into the TextBox fields of the Manipulator.

### 4.2.3.2. private void UpdateChangedPicture(string FileName)

**Pre-conditions:** 

The data of an image has been previously loaded into the Manipulator. Post-conditions:

A new image based on the FileName parameter has been loaded into the picChanged and the picChangedSmall data fields.

Parameters:

The FileName parameter is the name and path of a JPEG file to be loaded. Return values:

Function returns void.

Description:

This function is used to update picChanged and picChangedSmall data members, by loading a pre-existing image. If the FileName parameter is not a valid JPEG image, then an error message should be displayed by calling the ShowWarning() method. Lastly, this method should do some error checking to make sure this function executes properly. If an error is encountered, then the ShowWarning() method should be called to display the error to the user and the txtError TextBox control should be updated with this error information.

#### **4.2.3.3. private bool ShowWarning(string message, string caption)** Pre-conditions:

None

Post-conditions:

A warning message box is displayed for the user to see and decide how to proceed. This box will be shown until the user clicks either the Ok or Cancel Button control on this message box, at which point, this method will exit.

### Parameters:

The message parameter is explanation of the warning message.

The caption parameter is Window title of warning message box.

Return values:

Function returns True if the user has clicked Ok and False if the user has clicked Cancel.

Description:

The purpose of this method is to be used by any method that wants to display a warning message to the user. In addition, this method should return a True or False value, depending on the response given by the user receiving this message. This method should call the standard MessageBox control to show the message.

# 4.2.3.4. private bool ShowWarning(string message)

Pre-conditions: None.

Post-conditions:

A warning message box is displayed for the user to see and decide how to proceed. This box will be shown until the user clicks either the Ok or Cancel Button control on this message box, at which point, this method will exit.

Parameters:

The message parameter is explanation of the warning message.

Return values:

Function returns True if the user has clicked Ok and False if the user has clicked Cancel.

Description:

This function is a simpler version of the other ShowWarning method. This function will create a default title for the warning message box. Then, this function will call the other ShowWarning(string message, string caption) method with the message parameter and the default title created.

# 4.2.3.5. private void ClearInterfaceData()

Pre-conditions: None.

Post-conditions:

All of the TextBox controls for all of the data fields within the Manipulator will be reinitialized to empty strings.

Parameters: None.

Return values:

Function returns void.

Description:

This purpose of this method is to be called by any other method that needs to clear out all of the data fields within the user interface. Specifically, this method should set all of the strings to empty in every TextBox control found in the data sub-tabs of the Console tab on the Manipulator frmMain Form. It should also clear out all of the PictureBox controls within all of the Tab controls of the application.

## 4.2.3.6. private void WriteFile(ref byte[] ByteDataToWrite)

Pre-conditions: None.

Post-conditions:

A new file with the data contained in the ByteDataToWrite array has been created.

Parameters:

The ByteDataToWrite parameter is byte array of data to be written to file. Return values:

Function returns void.

Description:

The Purpose of this function is to allow the caller to create a new file based upon the data in the byte array passed in. This file created should be the binary value of the byte array and nothing more. If the byte array is null then an empty file should be created. The name of this file will be based upon file name in the txtChangedFile TextBox control. Lastly, this method should do some error checking to make sure this function executes properly. If an error is encountered, then the ShowWarning() method should be called to display the error to the user and the txtError TextBox control should be updated with this error information.

### 4.2.3.7. private void ClearData()

Pre-conditions: None.

Post-conditions:

All of the data members used to store information about the file structure of the current JPEG image are reinitialized to zero.

Parameters: None.

Return values:

Function returns void.

Description:

The purpose of this method is to allow the caller to reinitialize all of the data members that store information about the structure of the previous JPEG image loaded. This function should set the following data members to zero: NumberOfLines, RestartInterval, FrameSize, ExpandImage, RestartMod8, SizeOfHuffman (all 8 array members), SizeOfQuantizer (all 4 array members), SizeOfAppData (all 10 array members), SizeOfScanHeader, SizeOfProgression and SizeOfComments. Also, the FileOrder Queue should be cleared.

### 4.2.3.8. private void LoadNewProject()

Pre-conditions: None.

Post-conditions:

A previously existing SEP project file has been reloaded into the Manipulator. Parameters: None.

Return values:

Function returns void.

Description:

The purpose of the function is to allow the caller to load a pre-existing SEP project file. This function should prompt the user to save the current project, if there is one currently loaded. Then this function should call the ClearInterfaceData() method and then should open the file and read all data, to reload all of the corresponding fields in the interface. This method should load the project notes stored in the SEP file into the txtNotes TextBox interface control. This method should also reload all of the PictureBox controls from the image file information stored in the SEP file. This method should do some error checking to make sure all of the images load and that this method executes properly. If there is an error, the ShowWarning() method should be called and the txtError TextBox control should be updated with this error information.

# 4.2.3.9. private void SaveNewProject()

Pre-conditions: None.

Post-conditions:

All of the current values loaded in the Manipulator, any project notes and current image file names have been saved in a SEP project file name based upon the file name string in the txtProjectPath TextBox control.

Parameters: None.

Return values:

Function returns void.

Description:

The purpose of this method is to allow the caller to save an SEP project file based upon the current values loaded in the interface of the Manipulator. The data saved should include both the file name and paths of the images currently loaded within the Manipulator and all of the data in the TextBox controls on the sub-tabs located under the Console tab, including the txtNotes control for the project notes. The project name should be the file name and path stored in the txtProjectPath TextBox control. If a file with this name already exists, the user should be asked if it is okay to overwrite the pre-existing project file. Lastly, this method should do some error checking to make sure this function executes properly. If an error is encountered, then the ShowWarning() method should be called to display the error to the user and the txtError TextBox control should be updated with this error information.

### 4.2.3.10. private void CreateNewProject()

Pre-conditions: None.

Post-conditions:

The current project within the Manipulator is closed and a new SEP project file is created. All of the data loaded in the Manipulator should stay the same, except the txtNotes TextBox for the project notes should be cleared out for the new project file.

Parameters: None.

Return values:

Function returns void.

Description:

The purpose of this method is to allow the caller to create a new SEP project for the picture and data currently loaded in the Manipulator. If there is a project file currently open, then the user should be prompted to save before this project is closed. The txtNotes TextBox should be cleared out, as these notes belong to the last project. Then the user should be prompted to create a new project name for the new SEP project and all of the current data within the Manipulator should be saved to this new project. Lastly, this method should do some error checking to make sure this function executes properly. If an error is encountered, then the ShowWarning() method should be called to display the error to the user and the txtError TextBox control should be updated with this error information.

### **4.2.4. ISE Methods to Convert from Binary to ASCII**

The Manipulator methods found within this section of the document are related to converting binary data to the ASCII characters displayed in the interface for the user to view. These functions, along with the methods in section 4.2.5 of this document, represent the lowest level of functionality that the application is required to perform. The prototypes and definitions of each of these methods are outlined in this section of the document.

# **4.2.4.1. private void** SetCharValues(int OneByte, ref char HighBits, ref char LowBits)

Pre-conditions: None.

Post-conditions:

The LowBits parameter is set to an ASCII character between 0 to F, based upon the value of bits at positions 0 through 3 of the bit-index of the OneByte parameter passed in. The HighBits parameter is set to an ASCII character of 0 to F, based upon the value of bits at positions 4 through 7 of the bit-index of the OneByte parameter passed in.

#### Parameters:

The OneByte parameter is an integer value between 0 and 255 (8-bits), representing the value of one byte.

The HighBits parameter is a reference to a char where the char value resulting from the 4 most significant bits of the OneByte parameter can be stored. The LowBits parameter is a reference to a char where the char value resulting from the 4 least significant bits of the OneByte parameter can be stored.

Return values:

Function returns void.

Description:

The purpose of this method is to allow the caller to easily convert an 8-bit binary value to two ASCII characters representing the hexadecimal value of these 8-bits. To perform this functionality, this method should split the OneByte parameter into integer values, each with 4 bits in them. Then, this function should call the Convert() method that takes an integer and returns a char for each of these two 4-bit values to get the hexadecimal representation of each. Then, each char should be returned in the two reference parameters.

#### **4.2.4.2. private char** Convert(int Value)

Pre-conditions: None.

Post-conditions:

A character based on the hexadecimal value of the integer parameter passed in should be returned.

Parameters:

The Value parameter is an integer value between 0 and 15 (4-bits).

Return values:

Function returns a char based upon the hexadecimal value of the parameter. Description:

The purpose of this function allows the caller to convert the 4-bit value of the parameter to an ASCII character representing its hexadecimal value. This function will return the character 'X' if the value of the parameter is not between the value of 0 and 15 and an error message box, txtError, will be displayed to the user.

### 4.2.4.3. private void LoadPictureData(string FilePath)

Pre-conditions: None.

Post-conditions:

All of the data for the JPEG image based upon the FilePath parameter is loaded into all of the appropriate interface TextBox controls for the user to view. Parameters:

The FilePath parameter is the file name and path to a JPEG image. Return values:

Function returns void.

Description:

The purpose of this method is to load the binary file data for a JPEG image into the all of the appropriate TextBox data fields within the Manipulator interface. This function opens the JPEG file in binary mode and reads all the data from it. Every byte read from the file is converted to its hexadecimal representation and is stored in the OriginalDataStream data member. Then, to load all of the data in the OriginalDataStream string in to the interface, the LoadInterfaceData() method is called. Lastly, this method should do some error checking to make sure this function executes properly. If an error is encountered, then the ShowWarning() method should be called to display the error to the user and the txtError TextBox control should be updated with this error information.

### 4.2.4.4. private void LoadInterfaceData(ref StringBuilder HexChars)

Pre-conditions: None.

Post-conditions:

All of the character data contained in the HexChars parameter is broken apart and stored in the appropriate TextBox data fields in the Manipulator.

Parameters:

The HexChars parameter contains the file data for a JPEG image converted to ASCII characters representing the hexadecimal value of each byte found in the original JPEG file.

Return values:

Function returns void.

Description:

The purpose of this method is to take an string of ASCII characters that represent a JPEG file, break the file down into its various frames and then input all of this data to its corresponding TextBox data field in the interface. As such, this function is one of the largest functions in the Manipulator and performs many tasks during its execution. This method should read through the data in the HexChars parameter passed in. Every time a file marker is found, it should be enqueued into the FileOrder Queue data member. Then, the data found behind this particular marker should be loaded into its corresponding data field TextBox control in the interface of the Manipulator. Since we have to account for every possible marker found within the JPEG standard<sup>2</sup>, this function should be implemented with a number of switch statements to satisfy all possibilities. Also, as this function encounters the different frames within the file, all of the appropriate file structure data members of the JPEG Manipulator should be set. Lastly, this method should do lots of error checking to make sure this function executes properly. Items to check for errors are possible errors in the structure or format of the file and to make sure no exceptions occur when loading the interface. If an error is encountered, then the ShowWarning() method should be called to display the error to the user and the txtError TextBox control should be updated with this error information.

#### 4.2.5. ISE Methods to Convert from ASCII to Binary

The Manipulator methods found within this section of the document are related to converting data from ASCII characters to Binary format, so that a new image can be created based upon the values currently loaded in the Manipulator's interface. These functions, along with the methods in section 4.2.4 of this document, represent the lowest level of functionality that the application is required to perform. The prototypes and definitions of each of these methods are outlined in this section of the document.

<sup>&</sup>lt;sup>2</sup> For full information about the JPEG standard, refer to the "JPEG Still Image Data Compression Standard" book referenced in the related readings in section 8 of this document.

## 4.2.5.1. private byte SetByteValue(char HighBits, char LowBits)

Pre-conditions: None.

Post-conditions:

The LowBits and HighBits parameters are converted to integers and then combined to form the byte value that is returned by this function.

Parameters:

The HighBits parameter is an ASCII character that represents a value of 0 to 15, in the form of 0 to F, for the 4 most significant bits of the byte that will be returned.

The LowBits parameter is an ASCII character that represents a value of 0 to 15, in the form of 0 to F, for the 4 least significant bits of the byte that will be returned. Return values:

Function returns a byte value based upon the parameters passed in. Description:

The purpose of this method is to allow the caller to easily convert two ASCII characters, between 0 to F, to their binary values and then combine them to form a one-byte value. This function should call the Convert() method that takes a char and returns a byte for each of these two parameters to get the integer value of each. Then, it should combine both of these integer values to form one full byte value. Finally, this byte value should be returned when the function exits.

# 4.2.5.2. private int Convert(char Hex)

Pre-conditions: None.

Post-conditions:

An integer representing the binary value of the hexadecimal ASCII character parameter passed will be returned.

Parameters:

The Hex parameter is an ASCII character between 0 and F.

Return values:

Function returns an int based upon the hexadecimal value of the char parameter. Description:

The purpose of this function allows the caller to convert an ASCII character between 0 and F to its corresponding integer value of 0 to 15. This function will return a -1 if the char parameter passed in is not between the value of 0 and F and an error message will be displayed for the user.

# 4.2.5.3. private bool CreateChangedPicture(ref byte [] File)

Pre-conditions: None.

Post-conditions:

All of the character data contained in each of the data TextBox controls for the JPEG file is recombined and input, in order, into the File parameter passed. Parameters:

The File parameter is storage space for the new file byte array. All the data for the new JPEG image will be based on the conversion of the ASCII characters that are currently loaded in all of the data fields of the Manipulator's interface. Return values:

Function returns void.

Description:

The purpose of this method is to take all of the data currently loaded in the Manipulator's interface and recombine these values into one large byte array. This byte array will contain all of the binary data in the exact form the as the current ASCII chars loaded in the data fields of the Manipulator. As such, this function is one of the largest functions in the Manipulator and performs many tasks during its execution. This function should start dequeuing and re-enqueuing the markers stored in the FileOrder Queue. For each file marker found in this queue, the data in the corresponding interface data TextBox should be processed. This function should read the data from the particular TextBox, convert this data to binary and then input the resulting data into the File byte array parameter passed into this function. Lastly, this method should do lots error checking to make sure this function executes properly. If an error is encountered, then the ShowWarning() method should be updated with this error information.

### 4.2.5.4. private void CreateISEImage()

Pre-conditions: None.

Post-conditions:

All of the data for the new JPEG image being created is written to the file name contained in the txtChangedFile TextBox field.

Parameters: None.

Return values:

Function returns void.

Description:

The purpose of this method is to create a new manipulated image based upon all of the data currently loaded within the Manipulator. To perform this functionality, this function should call the CreateChangedPicture() method to create a file string to store the new file data. Then, this function should call the WriteFile() method to write all of this data to the new file. Then, to update the Manipulated picture files, this function should call the UpdateChangedPicture() method. Lastly, this method should do some error checking to make sure this function executes properly. If an error is encountered, then the ShowWarning() method should be called to display the error to the user and the txtError TextBox control should be updated with this error information.

### 4.2.6. Data Members of the JPEG Manipulator

This section of the design document defines all data members the JPEG Manipulator application will use to perform its functions. Many of these data members are Windows form components used for the graphical user interface, along with a smattering of primitive data types. The main form, frmMain, of the Manipulator application is inherited from the System.Windows.Forms class. It is a Windows form control that contains all of the following data members, along with all of the previously described methods.

# 4.2.6.1. public class frmMain: System.Windows.Forms.Form

This class contained within the JPEG Manipulator namespace is the definition of the main form of the Manipulator application. This form contains all of the data members for the application. Through this form, the entire JPEG Manipulator application is executed. This class is broken down into all of the data members and methods found in section 4.2 of this document.

# 4.2.6.2. Menu Controls

The Manipulator will employ a menu control to fulfill its functionality requirement<sup>3</sup> of behaving like a standard Windows application. The declaration of this data member is as follows:

private System.Windows.Forms.MainMenu menuFrmMain;

Aside from the Menu control, there are a number of MenuItem controls connected to this Menu. All of the declarations for these menu items are as follows:

private System.Windows.Forms.MenuItem menuFile; private System.Windows.Forms.MenuItem menuOpen; private System.Windows.Forms.MenuItem menuExit; private System.Windows.Forms.MenuItem menuOpenProject; private System.Windows.Forms.MenuItem menuSaveProject; private System.Windows.Forms.MenuItem menuNewProject; private System.Windows.Forms.MenuItem menuHelpMain; private System.Windows.Forms.MenuItem menuHelpMain; private System.Windows.Forms.MenuItem menuHelpMain; private System.Windows.Forms.MenuItem menuHelp; private System.Windows.Forms.MenuItem menuHelp; private System.Windows.Forms.MenuItem menuEdit; private System.Windows.Forms.MenuItem menuEdit; private System.Windows.Forms.MenuItem menuCut; private System.Windows.Forms.MenuItem menuCut;

# 4.2.6.2. Picture Box Controls

To serve the purpose of displaying images within the Manipulator, a series of Picture Box controls will be employed for image viewing. The declaration of these data members is as follows:

private System.Windows.Forms.PictureBox picOriginal;

<sup>&</sup>lt;sup>3</sup> For a full listing of all Team ISE product requirements, please see the Final Requirements Specification referenced in the Related Readings in section 8 of this document.

private System.Windows.Forms.PictureBox picChanged; private System.Windows.Forms.PictureBox picOriginalSmall; private System.Windows.Forms.PictureBox picChangedSmall;

# 4.2.6.3. Save/Open File Dialog Controls

For the purpose of browsing for files to open or save, the Manipulator will employ standard dialog box controls. The declaration of these data members is as follows:

private System.Windows.Forms.SaveFileDialog saveFileDialog1; private System.Windows.Forms.OpenFileDialog openFileDialog1; private System.Windows.Forms.OpenFileDialog openFileDialog;

# 4.2.6.4. Component Control

In order to be properly classified as a valid Windows Form derived from the Form class, each Windows form requires the IContainer to be one of its data members. This allows the Form to be used as a component. The declaration of this data member is as follows:

private System.ComponentModel.IContainer components;

# 4.2.6.5. ToolTip Control

To further ease the use of the Manipulator, all controls viewable to the user in the interface will have some tool tip information as part of their data, so that users will know what purpose all of the controls in the application serve. The declaration of this data member is as follows:

private System.Windows.Forms.ToolTip toolTips;

# 4.2.6.6. Tab Controls

To break up all of the information and Windows controls that will be displayed for the Manipulator user, the application is designed to use a series of tab controls that will hold the various categories of data found within a JPEG image. Certain tabs, namely the tabProject, tabEncrypt and tabDecrypt will not contain data directly from the image, but the data stored here will relate to the specific image. The main tab control used for the Console, Original Picture and Manipulated Picture tab pages will be declared on the frmMain, which is declared as follows:

private System.Windows.Forms.TabControl tabMain;

In addition, the Console tab will have another set of tab controls that contain all of the various JPEG data presentation controls, and will be declared as follows:

private System.Windows.Forms.TabControl tabSubConsole;
All of the tab pages will be placed on one of these previous two tab controls. All of the tab pages that will be placed on tabMain will be declared as follows:

private System.Windows.Forms.TabPage tabConsol; private System.Windows.Forms.TabPage tabOriginal; private System.Windows.Forms.TabPage tabChanged;

All of the TabPage controls that will be placed on tabSubConsole will be declared as follows:

private System.Windows.Forms.TabPage tabProject; private System.Windows.Forms.TabPage tabFile; private System.Windows.Forms.TabPage tabQuantizer; private System.Windows.Forms.TabPage tabEncodedData; private System.Windows.Forms.TabPage tabHuffman1; private System.Windows.Forms.TabPage tabHuffman2; private System.Windows.Forms.TabPage tabApplicationData; private System.Windows.Forms.TabPage tabApplicationData; private System.Windows.Forms.TabPage tabEncrypt; private System.Windows.Forms.TabPage tabEncrypt; private System.Windows.Forms.TabPage tabEncrypt;

#### 4.2.6.7. TextBox Controls

This application will use a large number of TextBox controls to store all of the potential data contained within a JPEG image. The declarations for all of these TextBox controls used within the Manipulator application are outlined in this section of the document. The TextBox controls that will be found on the tabFile TabPage control will be declared as:

private System.Windows.Forms.TextBox txtChangedFile; private System.Windows.Forms.TextBox txtOriginalFile; private System.Windows.Forms.TextBox txtComments; private System.Windows.Forms.TextBox txtFileSize;

The TextBox controls that will be found on the tabHuffman1 and tabHuffman2 TabPage controls will be declared as:

private System.Windows.Forms.TextBox txtHuffman1; private System.Windows.Forms.TextBox txtHuffman2; private System.Windows.Forms.TextBox txtHuffman3; private System.Windows.Forms.TextBox txtHuffman4; private System.Windows.Forms.TextBox txtHuffman5; private System.Windows.Forms.TextBox txtHuffman6; private System.Windows.Forms.TextBox txtHuffman7; private System.Windows.Forms.TextBox txtHuffman7; private System.Windows.Forms.TextBox txtHuffman8;

private System.Windows.Forms.TextBox txtHuffmanOriginal1;

private System.Windows.Forms.TextBox txtHuffmanOriginal2; private System.Windows.Forms.TextBox txtHuffmanOriginal3; private System.Windows.Forms.TextBox txtHuffmanOriginal4; private System.Windows.Forms.TextBox txtHuffmanOriginal5; private System.Windows.Forms.TextBox txtHuffmanOriginal6; private System.Windows.Forms.TextBox txtHuffmanOriginal7; private System.Windows.Forms.TextBox txtHuffmanOriginal7;

The TextBox controls that will be found on the tabEncodedData TabPage control will be declared as:

private System.Windows.Forms.TextBox txtEncodedData; private System.Windows.Forms.TextBox txtScanHeader; private System.Windows.Forms.TextBox txtOriginalEncodedData; private System.Windows.Forms.TextBox txtOriginalHeader;

The TextBox controls that will be found on the tabApplicationData TabPage control will be declared as:

private System.Windows.Forms.TextBox txtApplicationData1; private System.Windows.Forms.TextBox txtApplicationData2; private System.Windows.Forms.TextBox txtApplicationData3; private System.Windows.Forms.TextBox txtApplicationData4; private System.Windows.Forms.TextBox txtApplicationData5; private System.Windows.Forms.TextBox txtApplicationData6; private System.Windows.Forms.TextBox txtApplicationData7; private System.Windows.Forms.TextBox txtApplicationData7; private System.Windows.Forms.TextBox txtApplicationData8; private System.Windows.Forms.TextBox txtApplicationData7; private System.Windows.Forms.TextBox txtApplicationData8; private System.Windows.Forms.TextBox txtApplicationData10;

The TextBox controls that will be found on the tabQuantizer TabPage control will be declared as:

private System.Windows.Forms.TextBox txtQuantizer1; private System.Windows.Forms.TextBox txtQuantizer2; private System.Windows.Forms.TextBox txtQuantizer3; private System.Windows.Forms.TextBox txtQuantizer4;

private System.Windows.Forms.TextBox txtQuantizerOriginal1; private System.Windows.Forms.TextBox txtQuantizerOriginal2; private System.Windows.Forms.TextBox txtQuantizerOriginal3; private System.Windows.Forms.TextBox txtQuantizerOriginal4;

The TextBox controls that will be found on the tabMisc TabPage control will be declared as:

private System.Windows.Forms.TextBox txtNumberLines; private System.Windows.Forms.TextBox txtRestartMod8; private System.Windows.Forms.TextBox txtHierarchial; private System.Windows.Forms.TextBox txtExpand; private System.Windows.Forms.TextBox txtRestart; private System.Windows.Forms.TextBox txtError;

The TextBox controls that will be found on the tabProject TabPage control will be declared as:

private System.Windows.Forms.TextBox txtProjectPath; private System.Windows.Forms.TextBox txtNotes;

#### 4.2.6.8. Label Controls

To allow the user to understand what all of the Windows controls used within the Manipulator's interface do, a Label should be made for most of the TextBox controls, along with a few other controls. The declaration for each of these Label controls is outline in this section of the document. The Label controls that will be found on the tabFile TabPage control will be declared as:

private System.Windows.Forms.Label lblOriginalFile; private System.Windows.Forms.Label lblChangedFile; private System.Windows.Forms.Label lblFileSize; private System.Windows.Forms.Label lblComments;

The Label controls that will be found on the tabProject TabPage control will be declared as:

private System.Windows.Forms.Label lblFilePath; private System.Windows.Forms.Label lblNotes;

The Label controls that will be found on the tabHuffman1 and tabHuffman2 TabPage controls will be declared as:

private System.Windows.Forms.Label lblHuffman1; private System.Windows.Forms.Label lblHuffman2; private System.Windows.Forms.Label lblHuffman3; private System.Windows.Forms.Label lblHuffman5; private System.Windows.Forms.Label lblHuffman6; private System.Windows.Forms.Label lblHuffman7; private System.Windows.Forms.Label lblHuffman8;

private System.Windows.Forms.Label lblHuffmanMarker1; private System.Windows.Forms.Label lblHuffmanMarker2; private System.Windows.Forms.Label lblHuffmanMarker3; private System.Windows.Forms.Label lblHuffmanMarker4; private System.Windows.Forms.Label lblHuffmanMarker5; private System.Windows.Forms.Label lblHuffmanMarker6; private System.Windows.Forms.Label lblHuffmanMarker7; private System.Windows.Forms.Label lblHuffmanMarker8;

private System.Windows.Forms.Label lblHuffmanOriginalMarker1; private System.Windows.Forms.Label lblHuffmanOriginalMarker2; private System.Windows.Forms.Label lblHuffmanOriginalMarker3; private System.Windows.Forms.Label lblHuffmanOriginalMarker4; private System.Windows.Forms.Label lblHuffmanOriginalMarker5; private System.Windows.Forms.Label lblHuffmanOriginalMarker6; private System.Windows.Forms.Label lblHuffmanOriginalMarker6; private System.Windows.Forms.Label lblHuffmanOriginalMarker6;

private System.Windows.Forms.Label lblHuffmanOriginal1; private System.Windows.Forms.Label lblHuffmanOriginal2; private System.Windows.Forms.Label lblHuffmanOriginal3; private System.Windows.Forms.Label lblHuffmanOriginal4; private System.Windows.Forms.Label lblHuffmanOriginal5; private System.Windows.Forms.Label lblHuffmanOriginal6; private System.Windows.Forms.Label lblHuffmanOriginal7; private System.Windows.Forms.Label lblHuffmanOriginal7;

The Label controls that will be found on the tabEncodedData TabPage control will be declared as:

private System.Windows.Forms.Label lblScanHeader; private System.Windows.Forms.Label lblEncodedData; private System.Windows.Forms.Label lblOriginalHeader; private System.Windows.Forms.Label lblOriginalEncodedData;

The Label controls that will be found on the tabApplicationData TabPage control will be declared as:

private System.Windows.Forms.Label lblApplicationData1; private System.Windows.Forms.Label lblApplicationData2; private System.Windows.Forms.Label lblApplicationData3; private System.Windows.Forms.Label lblApplicationData4; private System.Windows.Forms.Label lblApplicationData5; private System.Windows.Forms.Label lblApplicationData6; private System.Windows.Forms.Label lblApplicationData7; private System.Windows.Forms.Label lblApplicationData8; private System.Windows.Forms.Label lblApplicationData7; private System.Windows.Forms.Label lblApplicationData8; private System.Windows.Forms.Label lblApplicationData8; private System.Windows.Forms.Label lblApplicationData8; private System.Windows.Forms.Label lblApplicationData10; private System.Windows.Forms.Label lblApplicationMarker1; private System.Windows.Forms.Label lblApplicationMarker2; private System.Windows.Forms.Label lblApplicationMarker3; private System.Windows.Forms.Label lblApplicationMarker4; private System.Windows.Forms.Label lblApplicationMarker5; private System.Windows.Forms.Label lblApplicationMarker6; private System.Windows.Forms.Label lblApplicationMarker7; private System.Windows.Forms.Label lblApplicationMarker7; private System.Windows.Forms.Label lblApplicationMarker8; private System.Windows.Forms.Label lblApplicationMarker8; private System.Windows.Forms.Label lblApplicationMarker8; private System.Windows.Forms.Label lblApplicationMarker9;

The Label controls that will be found on the tabQuantizer TabPage control will be declared as:

private System.Windows.Forms.Label lblQuantizer1; private System.Windows.Forms.Label lblQuantizer2; private System.Windows.Forms.Label lblQuantizer3; private System.Windows.Forms.Label lblQuantizer4;

private System.Windows.Forms.Label lblQuantizerOriginal1; private System.Windows.Forms.Label lblQuantizerOriginal2; private System.Windows.Forms.Label lblQuantizerOriginal3; private System.Windows.Forms.Label lblQuantizerOriginal4;

private System.Windows.Forms.Label lblQuantizerOriginalMarker1; private System.Windows.Forms.Label lblQuantizerOriginalMarker2; private System.Windows.Forms.Label lblQuantizerOriginalMarker3; private System.Windows.Forms.Label lblQuantizerOriginalMarker4;

private System.Windows.Forms.Label lblQuantizerMarker1; private System.Windows.Forms.Label lblQuantizerMarker2; private System.Windows.Forms.Label lblQuantizerMarker3; private System.Windows.Forms.Label lblQuantizerMarker4;

The Label controls that will be found on the tabMisc TabPage control will be declared as:

private System.Windows.Forms.Label lblNumberLines; private System.Windows.Forms.Label lblRestartMarker; private System.Windows.Forms.Label lblRestart; private System.Windows.Forms.Label lblNumberLinesMarker; private System.Windows.Forms.Label lblError; private System.Windows.Forms.Label lblRestartMod8; private System.Windows.Forms.Label lblHierarchialMarker; private System.Windows.Forms.Label lblHierarchialMarker; private System.Windows.Forms.Label lblHierarchial; private System.Windows.Forms.Label lblHierarchial; private System.Windows.Forms.Label lblExpand;

#### 4.2.6.9. Button Controls

The Manipulator interface will also provide a series of standard Windows Button controls for the user to click on to begin execution of certain functionality. All of the declarations for these Button controls are outlined in this section of the document. The Button controls that will be found on the tabQuantizer TabPage control will be declared as:

private System.Windows.Forms.Button btnRestoreQuantizer1; private System.Windows.Forms.Button btnRestoreQuantizer2; private System.Windows.Forms.Button btnRestoreQuantizer3; private System.Windows.Forms.Button btnRestoreQuantizer4;

The Button controls that will be found on the tabHuffman1 and the tabHuffman2 TabPage controls will be declared as:

private System.Windows.Forms.Button btnRestoreHuffman1; private System.Windows.Forms.Button btnRestoreHuffman2; private System.Windows.Forms.Button btnRestoreHuffman3; private System.Windows.Forms.Button btnRestoreHuffman5; private System.Windows.Forms.Button btnRestoreHuffman6; private System.Windows.Forms.Button btnRestoreHuffman7; private System.Windows.Forms.Button btnRestoreHuffman7; private System.Windows.Forms.Button btnRestoreHuffman7;

The Button controls that will be found on the tabProject TabPage control will be declared as:

private System.Windows.Forms.Button btnLoad; private System.Windows.Forms.Button btnNew; private System.Windows.Forms.Button btnSave; private System.Windows.Forms.Button btnLoadPicture; private System.Windows.Forms.Button btnSavePicture; private System.Windows.Forms.Button btnUpdatePicture;

#### 4.2.6.10. Additional Forms

In addition to the frmMain form, the Manipulator will have a couple of other small forms for providing a help Window, an about Window, and a loading form for the user to view. All of the declarations for the additional forms contained within the Manipulator are as follows:

private System.Windows.Forms.Form MainAbout; private System.Windows.Forms.Form MainHelp; private System.Windows.Forms.Form frmLoad;

#### 4.2.6.11. Image Type Members

To store the images that will be loaded into each of the PictureBox controls, the Manipulator will contain four data members to store each of these image's data. The declarations of each of these members are as follows:

private System.Drawing.Image JPEG; private System.Drawing.Image ISE; private System.Drawing.Image JPEGsmall; private System.Drawing.Image ISEsmall;

#### 4.2.6.12. Miscellaneous Members

Finally, a large portion of the data members mentioned within this document and/or already contained within the Manipulator application will be a slew of miscellaneous data members to store any additional data needed for the application. Since they do not fall under any other particular sections, the declarations of all of these data members are outlined here. The data member declarations are as follows:

private Queue FileOrder;

private string ChangedFileName;

private StringBuilder OriginalDataStream;

private int NumberOfLines; private int RestartInterval; private int FileSize; private int ExpandImage; private int RestartMod8;

private const int MAX\_HUFFMAN = 8; private const int MAX\_QUANTIZER = 4; private const int MAX\_APPDATA = 10;

private int [] SizeOfHuffman = new int [MAX\_HUFFMAN]; private int [] SizeOfQuantizer = new int [MAX\_QUANTIZER]; private int [] SizeOfAppData = new int [MAX\_APPDATA];

private int SizeOfScanHeader; private int SizeOfProgression; private int SizeOfComments;

private int FrameSize; private int Count; private int Value; private int High; private int Low;

private FileStream OriginalFile; private FileStream NewFile;

private const int MAX\_FILE\_SIZE = 10485760; // 10 meg private const int AVE\_FILE\_SIZE = 5242880; // 5 meg

private byte [] NewData;

private bool LoadingInterface; private bool LoadingProject;

These sections sum up all of the data members needed to complete the JPEG Manipulator application. All of the data members required for this application are listed here. The final product should be implemented with little-to-no difference from the data members listed above.

#### 4.3. Team ISE Web Site Design

To support the required functionality of the ISE web site, there are a series of web pages that need to be implemented. The web site will be implemented in HTML 4.01 Transitional to ease any future maintenance required by the sponsor. The following is a description of the design of the Team ISE web site.

#### 4.3.1. The ISE Web Site Index Page

The ISE index page, located at http://128.138.75.184, will be the default start page of the ISE web site. To conform to various web server standards, this index will be named index.html. This page contains an introduction to the website.

#### 4.3.2. The ISE Menu Bar

The ISE menu bar was generated using Xara Menu maker. The menu bar's top level consists of links to the other web pages. The "Documents", "Downloads", and "Links" buttons contain submenus which allow the user to directly connect to respective documents, downloads, and links without visiting the actual pages. The various buttons in the menu are:

- 1. The "Home" button links to index.html page.
- 2. The "Project Proposal" button will open the final project proposal document, in PDF form.
- 3. The "Documentation" button will display the DocumentIndex.html.
- 4. The "Project Sponsor" button will display the Sponsor.html.
- 5. The "Team Info" button will display the Team\_ISE.html page.
- 6. The "Downloads" button will display the Download.html page.
- 7. The "Links" button will display the Links.html page.
- 8. The "Contact button will display the Contact.html page.

#### 4.3.3. The ISE Project Proposal Document

The project proposal document will be shown in PDF format by clicking on the "Project Proposal" button on the menu. This will cause the document to be displayed in a new browser window. This document will be named ProjectProposal.pdf and will not contain links to other places within the ISE web site.

#### 4.3.4. The ISE Documentation Page

The ISE documentation page will contain all of the final documents created by the team during the course of this project. This page will have several links contained within it. However, at the time of creating this document the team cannot be sure about the final number of links that will be created for this page. The names of each of the buttons on this page will correspond to the documents that they link to. This page will be named DocumentIndex.html.

#### 4.3.5. The ISE Project Sponsor Page

The project sponsor page will provide a short description of the sponsor, Tom Lookabaugh, the work he is currently involved in and a link to his web page. Information on this page can be displayed as either images or text. This page will be named Sponsor.html.

#### 4.3.6. The Team ISE Info Page

The team information page will provide a short description of the Team ISE members, a picture of the team and links to various web pages. Information on this page can be displayed as either images or text. This page will be named Team\_ISE.html.

#### 4.3.7. The ISE Download Page

The ISE download page will contain the final production code and Manipulator application installer, along with a few other minor items, such as the .NET framework that is required before installing the Manipulator. This page will have several links contained within it. However, at the time of creating this document the team cannot be sure about the final number of links that will be created for this page. In addition to the download items, the page will contain some screenshots and product information. The names of each of the buttons on this page will correspond to the particular action chosen by the user. Information on this page can be displayed as either images or text. This page will be named Download.html.

#### 4.3.8. The ISE Links Page

The ISE links page will contain links to various related web pages. This page will have several links contained within it. However, at the time of creating this document the team cannot be sure about the final number of links that will be created for this page. Each link will conform to the button links on the menu page and will open in a new instance of the browser if the link redirects the user to a different web site. This page will be named Links.html.

### **5. FILE DESCRIPTIONS**

Since both the ISE production code and the JPEG Manipulator will be using input files and producing output files, we have dedicated this section of the document to defining how these files should be represented. These files break down into four categories: JPEG Standard Image Files, JPEG ISE Encrypted Files, Test Suite Manipulated Images and Test Suite Project Files.

#### 5.1. JPEG Standard Image Files

Both the Encryptor and the Manipulator will require standard JPEG image files as input and the Decryptor should produce standard JPEG images as output. A valid JPEG file, as defined within this document, is one that conforms to the ISO JPEG Baseline Still Image Compression Standard<sup>4</sup>. Both the Encryptor and Manipulator require standard JPEG images but do not discriminate against file names or extensions. Behavior and output of the Encryptor and the Manipulator, when processing files that do not conform to the ISO JPEG standard, will be considered undefined.

#### **5.2. JPEG ISE Encrypted Files**

The Encryptor and the Manipulator will produce selectively encrypted JPEG image files and both the Decryptor and the Manipulator will take selectively encrypted JPEG files as input for processing. A valid JPEG ISE encrypted file should maintain the structure exactly as its corresponding decrypted JPEG image file with three exceptions. First, there should be a one-byte file descriptor prefixed on the original image file, which describes the type of ISE file being processed. Second, every data frame within the file should be processed in accordance to the algorithm outlined in section 4.1.9 of this document. Third, the file extension should be changed to .ise.

#### 5.3. Test Suite Manipulated Images

When using the Manipulator to alter images by changing data within the different frames of the file, the Manipulator will produce images based upon the structure of the file currently loaded. Although most of the time these files will conform to the format of a standard JPEG image, the Manipulator should not restrict the user from creating the file in anyway desired. The user should be allowed to input any data desired and the Manipulator should not discriminate against any modifications in any frame of the original image, as long as the data is an ASCII character between 0 and F. Please note that this means that files created by the Manipulator in testing mode may or may not be able to be loaded with a standard JPEG image viewer. Of course, files processed in decryption mode by the Manipulator should conform to the JPEG standard image files outlined in section 5.1 of this document.

#### **5.4. Test Suite Project Files**

Since the Manipulator will supply the user with the ability to input comments about a particular project and save all of the currently loaded project information, the Manipulator will create files other than images or encrypted files. The Manipulator will also create

<sup>&</sup>lt;sup>4</sup> For full information about the JPEG standard, refer to the "JPEG Still Image Data Compression Standard" book referenced in the related readings in section 8 of this document.

project files, with the extension .sep, that will contain all of the project data for future use. The user will have the ability to create these files from the Manipulator's save project option.

Two considerations will determine the format of these files: The exact format of the JPEG file and whether or not the fields have been manipulated. This second condition is necessary to avoid writing redundant data and to minimize the size of the .sep file. The format of the file will be an ASCII file with the following data:

- 1. Project notes data followed by a new line.
- 2. Original JPEG file name and path followed by a new line.
- 3. The manipulated JPEG file name and path followed by a new line.
- 4. For the quantizer tables, write the number that have been modified followed by a new line.
- 5. If the number is greater than zero, write the quantizer table(s) number followed by a new line.
- 6. Write the modified table values to the file, followed by a new line.
- 7. For the Huffman tables, write the number that have been modified followed by a new line.
- 8. If the number is greater than zero, write the Huffman table number followed by a new line.
- 9. Write the modified table values to the file, followed by a new line.
- 10. If the encoded data has been modified, write the modified data to the file, followed by a new line.

### 6. SUMMARY

This project design document has outlined the necessary functionality required to complete the ISE class production code, the JPEG Mainpulator test suite and the Team ISE web site. The user interface for each of these modules, found in section 2, has been provided as a design guideline for the ISE final products. The high-level modular decomposition, found in section 3, gives a general overview of the high-level design for each of the different modules in the project. The design section, located in section 4, provides an in-depth, low-level description of each module to supply a guideline for the upcoming development process. In addition, section 5 defines how input and output files should be formatted and the standards those files should conform to. In writing the document, the team has tried to be as specific as possible about the design of each of the project modules. Team ISE's hope is that this design document will provide enough information to successfully complete the project with little-to-no future change in the design of any module.

### 7. GLOSSARY

#### AES

AES is an abbreviation for Advanced Encryption Standard. AES is an encryption system that utilizes block ciphering. http://csrc.nist.gov/CryptoToolkit/aes/

#### ANSI C++

ANSI is an abbreviation for the American National Standards Institute. C++, pronounced "cee-plus-plus," is a programming language that was created Bjarne Stroupstrup at AT&T Bell Laboratories in 1983. ANSI C++ is the current standard C++ programming language as defined by the American National Standards Institute.

#### **Baseline Compression**

The Baseline Compression of a JPEG image is a subset of the sequential compression algorithm as it is define by the ISO standard for JPEG images.

#### **C**#

C#, pronounced "cee-sharp," is a programming language that was created by Microsoft Incorporated in 1999. C# is an object-oriented programming language that enables programmers to quickly build a wide range of applications based on the latest Microsoft .NET technologies.

#### **Cipher Text**

A cipher text is the resulting data of some plain text data that has undergone a process of encryption.

#### Compression

A technique designed to reduce the amount of memory needed to store data. Typically, these algorithms utilize patterns in a file to reduce the size.

#### Cryptography

The practice and study of encoding data such that the original data can only be decoded by trusted individuals, for the purpose of data secrecy.

#### Cryptosystem

A Cryptosystem is a system for encrypting and decrypting data.

#### Decryption

A procedure used in cryptography to convert cipher text into plain text.

#### Encryption

A procedure used in cryptography to convert plain text into cipher text.

#### GUI

GUI is an abbreviation for Graphical User Interface.

#### IJG

IJG is an abbreviation for the Independent JPEG Group. IJG is an informal group that writes and distributes a widely used free C++ library that provides JPEG image compression utilities. http://www.ijg.org/

#### ISO

ISO is an abbreviation for the International Organization for Standardization. ISO is the world's largest developer of standards, particularly the development of technical standards.

#### **JPEG**

JPEG is an abbreviation for the Joint Photographic Experts Group. A JPEG image is an image that has undergone a compression technique designed specifically to compress image file data.

#### Key

A key is some data known only to trusted individuals.

#### **Military Secrecy**

A level of secrecy where all of the original data is hidden.

#### MP3

MP3 is an abbreviation for the MPEG-1 Audio Layer-3. MP3 is a standard for compressing raw audio data.

#### MPEG

MPEG is an abbreviation for the Moving Picture Experts Group. MPEG is a standard for compressing raw digital video and audio data.

#### **Plain Text**

Plain text data is the original, unencrypted data.

#### Rijndael

The Rijndael, pronounced "rain doll," is the original name of a type of block cipher encryption system. Rijndael is now known as the AES encryption system. http://www.esat.kuleuven.ac.be/~rijmen/rijndael/

#### **Post-condition**

In reference to a method or function, the post-condition is a condition that the system should be in, if the function of method executes properly.

#### **Pre-condition**

In reference to a method or function, the pre-condition is a condition that has occurred before this method or function is called.

#### **Selective Encryption**

A cryptosystem which employs cunning methods to encrypt small, yet vital portions of data to reduce the amount of data encrypted while still rendering the file useless. Selective Encryption typically uses the knowledge of a file format and specifically how the data contained in the file relates to the files purpose and uses this knowledge to decide which portion of the file is encrypted.

#### **Visual Studio .NET**

Visual Studio .NET is Microsoft Corporation's latest integrated development environment for creating a wide variety of different types of software applications in multiple programming languages.

#### ZIP

ZIP is a standard file format for compressing a file without discriminating against the original file's type.

### **8. RELATED READINGS**

#### [Chang and Li 96]

Chang, H. and Li, X. On the Application of Image Decomposition to Image Compression and Encryption. 1996.

Describes image degradation based on compression and encryption.

#### [Chang and Li 2000]

Chang, H. and Li, X. Partial Encryption of Compressed Images and Videos. 2000.

Describes a partial encryption scheme used on compressed multimedia files.

#### [Droogenbroek and Benedett 2002]

Droogenbroek, M. and Benedett, R. *Techniques for Selective Encryption of Uncompressed and Compressed Images*. 2002.

#### [Kailasanathan and Naini 2003]

Kailasanathan, C. and Naini, R. *Compression Performance of JPEG Encryption Scheme*. 2003.

Describes compression performance of JPEG encryption.

#### [Daigaku and Griffith and Jarchow and Kadhim and Pouzeshi]

Daigaku, S., Griffith, G., Jarchow, J., Kadhim, J. and Pouzeshi A. *Requirement Specification*. 2003.

Describes the requirement for Team ISE and for the ISE project.

#### [Daigaku and Griffith and Jarchow and Kadhim and Pouzeshi]

Daigaku, S., Griffith, G., Jarchow, J., Kadhim, J. and Pouzeshi A. *System Architecture*. 2003.

Describes the high-level system architecture for the ISE project.

#### [Li and Knipe and Cheng 97]

Li, X., Knipe, J. and Cheng, H. *Image Compression and Encryption Using Tree Structures*. 1997.

Describes compression methods that utilize tree structures.

#### [Lookabaugh and Sicker and Keaton and Guoand and Vedula 2003]

Lookabaugh, T., Sicker, D., Keaton, D., Guoand, W. and Vedula, I. Security Analysis of Selectively Encrypted MPEG-e Streams. 2003.

Description of the methods and results of applying selective encryption to MPEG-2 streams.

#### [Miano 99]

Miano, J. *Compressed Image File Formats*. Addison Wesley Longman, Inc., Reading, Massachusetts, 1999.

Provides a description of the JPEG file format.

#### [Norcen and Uhl 2003]

Norcen, R. and Uhl, A. Selective Encryption of the JPEG2000 Bitstream. 2003.

Describes a selective encryption scheme on JPEG2000 files.

#### [Pennebaker and Mitchell 93]

Pennebaker, W. and Mitchell J. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, New York, 1993,

Provides a thorough description of the JPEG file format and its components.

#### [Podesser and Schmidt and Uhl 2002]

Podesser, M., Schmidt, H. and Uhl, A. Selective Bitplane Encryption for Secure Transmission of Image Data in Mobile Environments. 2002.

Describes Bitplane Encryption.

#### [Seo and Kim and Yoo and Dey and Agrawal 2003]

Seo, Y., Kim, D., Yoo, J., Dey, S., Agrawal, A. Wavelet Domain Imag Encryption by Subband Selection and Data Bit Selection. 2003.

Describes Wavelet Domain and Data Bit encryption methods.

# Design Presentation

# Team 1SE Image Selective Encryption



# Team ISE Image Selective Encryption

**Joe Jarchow Joseph Kadhim Geoffrey Griffith** Shinya Daigaku **Andrew Pouzeshi** 

### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

Presentation Overview

### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

**Presentation Overview** 

### **Problem:**

- Multimedia files are very large
- Encryption is expensive
  - Processing time
  - File size
- No widely accepted solutions
  - Encrypt entire file
  - No encryption

### **Problem Statement**

## **Affected User Scenarios:**

- Images on websites
- File sharing
- Cable TV

### **Problem Statement**

## Solution:

Selective Encryption

**Definition from MPEG paper:** 

Selective encryption applies encryption to a subset of a file with the expectation that the entire file will be rendered useless to anyone who cannot decrypt that subset.

### **Problem Statement**

### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

**Presentation Overview** 

## **Selective Encryption Requirements:**

- Perceivable degradation of file
- Encryption of less than 10%
- Minimize required computation
- Minimize increase in file size
- Cryptanalytic approach



**Encryption of Compressed File Types:** 

- Independent of time (JPEG)
  - Must affect image related target
  - Can use a block or stream cipher
- Synchronous (MPEG)
  - Target could affect the image
  - Target could affect time components
  - Might require stream cipher

### **Initial Research**



- Published international standards
  Partitioned into standard components
  - Descriptive
  - Mathematical

### **Initial Research**

### **JPEG Standard:**

Standard implementation of JPEG compression

http://www.ijg.org

### **Initial Research**

### JPEG Structure:

- Markers, headers and data
- Example:

ff e0 00 10 4a 46 49 46 00 01 01 01 00 48 00 48 00 00

## **Initial Research**

### Marker:

- Indicates which component
- Example marker:

ff e0 (indicates Application Data)

### **Initial Research**

### Header:

- Indicates size of parameters to follow
- Example header:

## 00 10 -- (16 bytes of data will follow)

### **Initial Research**

### Data:

- The information itself
- Example data:

4a 46 49 46 00 01 01 01 00 48 00 48 00 00 (16 bytes of information indicating what application created the file.)

### **Initial Research**

**Encrypting** *During* Compression:

- Would not produce standard file
- Requires reimplementation

# **Encrypting** *After* Compression:

- Layered approach
- Creates intermediate file
  - Allows different extension
- Algorithm can be easily reviewed
- Applicable to non-synchronous files

## **Initial Research**

## **General Development Approach:**

- Study Compression Standard
- Study earlier approaches
- Create a testing toolkit
- Evaluate each target:
  - Percentage of file
  - Perceivable damage
- Design selective encryption algorithm
- Cryptanalytic approach

**Initial Research**
# **Cryptanalytic Approach:**

- White hat
- Black hat
- Review by crypto community
- Correction of algorithm

### **Initial Research**

Joe J

### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

**Presentation Overview** 

**Criteria For Bad Targets:** 

- Optional markers
- Not used in Baseline JPEG images
- Does not affect visibility of the image
- Easily guessed or forged by a hacker

#### **Target Selection Process**

### **Determining Initial Bad Targets:**

- Resources:
  - JPEG Still Image Data Compression Standard
  - Compressed Image File Formats
  - ISO DIS 80918-1 Requirements and Guidelines
  - ISO DIS 80918-2 Compliance Testing
  - <u>http://www.funducode.com/freec/</u>

fileformats/format3/format3b.htm

**Target Selection Process** 

## APP - Application

No affect to visibility

**COM - Comments** 

- No affect to visibility
- **DAC Define Arithmetic Conditioning Tables** 
  - Not part of Baseline Compression
- **DHP Define Hierarchical Progression** 
  - Not part of Baseline Compression
- DNL Define Number of Lines
  - Easily forged (set size)

**Target Selection** 

- DRI Define Restart Interval
  - Easily forged (set size)
- EOI End of Image
  - Easily forged (always last marker)
- EXP Expand
  - Not part of Baseline Compression
- JPG Reserved for Future Extensions
  - Not used in Baseline Compression

### Target Selection

- RES Reserved
  - Not used in Baseline Compression
- RST Restart
  - Not part of Baseline Compression
- TEM Temporary
  - Not used in Baseline Compression
- SOS Start of Scan
  - Easily reconstructed
- Markers themselves are predictable



Target Selection

# **Remaining Targets for Selective Encryption:**

- Encoded Data Stream
- Quantizer Tables
- Huffman Tables





### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations



**Presentation Overview** 



- Target Analysis Toolkit
  - Convert
  - Analyze
  - Manipulator





# Convert:

- C++ program
- Convert Binary to Hexadecimal
- File information for a single JPEG image

**Statistical Analysis** 

This is an ASCII representation (in hexadecimal) of the binary values found in the file : Dust.jpg

Markers Found:=========

ff d8 -- Start of Image

ff e0 - Application Data - 00 10 - (16 bytes) - 4a 46 49 46 00 01 01 01 00 48 00 48 00 00

ff db -- Define Quantization Table -- 00 43 -- (67 bytes) -- 00 06 04 05 06 05 04 06 06 05 06 07 07 06 08 0a 10 0a 0a 09 09 0a 14 0e 0f 0c 10 17 14 18 18 17 14 16 16 1a 1d 25 1f 1a 1b 23 1c 16 16 20 2c 20 23 26 27 29 2a 29 19 1f 2d 30 2d 28 30 25 28 29 28

ff c0 -- Huffman Table -- Baseline DCT -- 00 11 -- (17 bytes) -- 08 01 cb 02 4a 03 01 22 00 02 11 01 03 11 01

ff c4 -- Huffman Table -- 00 1f -- (31 bytes) -- 00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09 0a 0b

#### **Statistical Analysis**

## Analyzer:

- File information for multiple JPEG's
  - Average file size
  - Average number of Huffman tables
  - Average size of Huffman tables
  - Average number of Quantizer tables
  - Average size of Quantizer tables
  - Average size of the encoded stream
  - Average number of markers
  - Number of files processed





## Analyzer (cont):

- Percent of the file dedicated to:
  - Huffman tables
  - Quantizer tables
  - Encoded Stream

## **Statistical Analysis**

## **Test Cases for JPEG Analysis:**

- Over 2500 JPEG images selected
  - Internet web sites
  - Digital photographs
  - Manmade images
- Size ranges:
  - •10-19KB, 100 KB, 1 MB, and larger
- Resolution Ranges:
  ·320x240, 640x480, and 800x640 pixels

**Statistical Analysis** 



**Statistical Analysis** 

#### **Encoded Data Stream:**

- SOI (Start of Image) marker
- Compressed data stream
- Takes up a large portion of the file
- Averaged 90% of the file!

## **Statistical Analysis**

## **Quantizer Tables:**

- DQT (Define Quantization Table) markers
- Defines Resolution
  - Luminance
  - Chrominance
- Averaged 0.88% of the file
- Unpredictable affects on image
- Might not visually damage the image!
- Can be replaced with another Quantizer!





### **Huffman Tables:**

- DHT (Define Huffman Table) markers
- Used to encode/decode the image data
- Averaged 1.84% of the file
- Considerable damage to image
- Mathematically derived from the image
- This makes the Huffman Tables a perfect target for Selective Encryption

**Statistical Analysis** 

### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

Geoff

**Presentation Overview** 

## **Requirements:**

- Testing tool
- Graphical user interface
- Displays each component
- Easy manipulation of JPEG files
- See changes side by side



### Modules:

- Standard Windows methods
- Graphical User Interface
- Common methods
- Convert binary to ASCII
- Convert ASCII to binary
- Encrypt and Decrypt methods



### Manipulator Design



## **Standard Windows Methods:**

- Required functions like main()
- Initialization functions
- Constructors and Destructors



## **Graphical User Interface:**

- Methods called during user interaction
- Event handlers
  - menus
  - buttons
  - text boxes



# **Common Methods:**

- Create/Load/Save
  - project(s)
  - picture(s)
- Show warning(s)
- Clear interface data
- Updated manipulated picture



Manipulator Design



Convert Binary to ASCII: Convert ASCII to binary

- Methods to load images to interface
- Create images from interface

**Encrypt and Decrypt methods** 

Calls production code methods



**Manipulator Design** 

### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

**Presentation Overview** 



## **JPEG Selective Encryption:**

- Remove application data
- Remove comment data
- Leave initial Huffman marker
- Encrypt:
  - Huffman data (except initial marker)
  - Next non-Huffman marker and header



Manipulator Design



## **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations





**Presentation Overview** 

### **Requirements:**

- Secure
- No increase in file size
- Recommendation from Prof. John Black





# AES (Rijndael):

- NIST selection of AES standard
- Block Cipher
- Rijmen and Daemen
- Open source optimized implementation
- Variable block length (128, 192, 256)
- Only whole byte operations

# **Encryption Algorithm Selection**



### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

**Presentation Overview** 



# **Encryption Algorithm:**

- Write file-type-byte to ".ise" file
  - '1' for JPEG
- Read from input file
- Write unencrypted to ".ise" file

**Selective Encryption Algorithms** 





**Selective Encryption Algorithms** 

Shinya

# **Encryption Algorithm (cont):**

- Read/Write until marker [ffc0 ffcf]
  - Indicates Huffman specification
    - ffc0 -- baseline frame
    - ffc4 -- Huffman table

## **Selective Encryption Algorithms**





**Selective Encryption Algorithms** 

Shinya




# **Encryption Algorithm (cont):**

- Write until non-Huffman marker
  - Below ffc0
  - Above ffcf









**Encryption Algorithm (cont):** 

- Read/Write unencrypted
  - Until end of file (ffd9)
  - Unless another Huffman marker

- Efficiency
  - 97% evaluated by only a few if statements



# **Decryption Algorithm:**

- Read file-type-byte from ".ise" file
  - '1' for JPEG
- Read/Write until marker [ffc0 ffcf]
  - Indicates start of encrypted data





### **Selective Encryption Algorithms**

Shinya



**Selective Encryption Algorithms** 

Shinya

# **Decryption Algorithm (cont):**

- Write decrypted text to output file
- Read/Write unencrypted
  - Until end of file
  - Unless another Huffman marker



### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

**Presentation Overview** 



**Object Oriented Outline:** 

- Data Abstraction
  - ISE constructors
  - Virtual encrypt/decrypt methods
  - Data members and gets/sets
    - File names
    - Key
  - Make file name methods



### **ISE Production Code Design**

- Information hiding
  - Data members
    - protected
  - Get/Set methods
    - File names
    - Key
    - File type



### **ISE Production Code Design**

Inheritance

# JPEG\_ISE Class

- Encrypt
- Decrypt

# **ISE** Class

- Constructor
- Gets/Sets
- Data Members



# **ISE Production Code Design**



- Polymorphism
  - Constructors
    - ise()
    - ise (key, input\_file\_name, ise\_file\_name)
      - encrypting
    - ise(key, ise\_file\_name, output\_file\_name)
      - decrypting

# **ISE Production Code Design**

- Polymorphism
  - Encryption
    - encrypt\_file()
    - encrypt\_file(key, input\_file\_name, ise\_file\_name)
  - Decryption
    - decrypt\_file()
    - decrypt\_file(key, ise\_file\_name, output\_file\_name)

### **ISE Production Code Design**

### **API Usage:**

#### **Encryption Scenario:**

char[] myKey = "ISE\_IS\_THE\_BEST"; char[] myInputFile = "myImage.jpg"; char[] myISEFile = "myImage.ise"; jpeg\_ise\* myISE; myISE = new jpeg\_ise(myKey,myInputFile,MyISEFile); myISE->encrypt\_file(); ceree myISE;

**ISE Production Code Design** 



# API Usage (cont):

### **Decryption Scenario:**

char[] myKey = "ISE\_IS\_THE\_BEST"; char[] myISEFile = "myImage.ise"; char[] myOutputFile = "myImageDecrypt.jpg"; jpeg\_ise\* myISE; myISE = new jpeg\_ise(); myISE->set\_key(myKey); myISE->set\_ise\_file(myISEFile); myISE->set\_output\_file(myOutputFile); myISE->decrypt\_file(); delete myISE;

**ISE Production Code Design** 

### **OO Benefits:**

- Objects easily extendable to other formats
- Clean, reliable code
- Apply what we've learned





### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations



**Presentation Overview** 

#### **Requirements:**

- Easy to maintain
- Distribute products/documentation
- Create on existing computer in lab

• http://128.138.75.184

Web Site Design



# Home Page





This website represents a team of University of Colorado students working under the sponsorship of Professor Tom Lookabaugh in the department of Computer Science to develop a series of selective encryption schemes applicable to various multi-media targets.

#### Web Site Design



### **Documentation**



Web Site Design

# Downloads



Web Site Design

#### Links



Web Site Design

### **Presentation Overview:**

- Statement of problem
- Initial research into compressed files
- Target Selection Process
- JPEG Statistical Analysis
- JPEG Manipulator Design
- JPEG Manipulator Demonstration
- Encryption Algorithm Selection
- JPEG Selective Encryption Algorithms
- ISE Production Code Design
- ISE Web Site Design
- Future Considerations

**Presentation Overview** 

Joe J

**Future Considerations:** 

- Black hat attacks
  - Huffman table
    - Replacement
    - Reconstruction
      - Based on Quantizer
      - Based on Application
  - Quantizer table
- Publish web site for community
- Corrections

**Future Considerations** 

Joe J

# Questions

# Questions



A

# Test Plan

# Team ISE Image Selective Encryption

ISE Test Plan March, 2004

# Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

100

101

0100

#### **Project Proposal**

Traffic constantly flows between computers connected to the Internet. Large volumes of information may take a long time traveling from destination to destination. Such a reduction in speed makes it desirable to compress the file as much as possible in order to send the smallest amount of data required. Thus, compression of data has allowed for the high-speed data transfers that have made Internet communication and business more feasible.

In addition to sending the smallest amount of information possible, users also attempt to maintain a certain level of security upon their information. Due to the fact that common encryption methods generally manipulate an entire file, most encryption algorithms tend to make the transfer of information more costly in terms of time and bandwidth. Thus, users pay a price for security relative to their desired level of security. One possible solution would be a system of encryption that works cooperatively with the standard compression schemes. *Selective Encryption* of only a small percentage of the file's bits will facilitate this solution. Because most encryption schemes will make the file larger, selective encryption seeks only to encrypt portions of the file that will make it unusable. In other words, if a user does not have the proper decryption device, the file should not be usable. Selective encryption will minimize the necessary increase in file size due to encryption while maintaining a maximum level of uselessness, or damage, to the product.

Team ISE (Image Selective Encryption) will deliver a package for selectively encrypting JPEG (Joint Photographic Experts Group) still image files. The package will provide the tools necessary to encrypt the critical information of a JPEG file in cooperation with existing standard compression tools. This package will handle JPEG files in such a way that only a small percentage of the total file will be encrypted. Selective Encryption security will not extend to the level of complete encryption, but rather to a level that would deter all but brute force attacks, allowing users to securely protect private JPEG images.

A JPEG image could be encrypted with any of the sufficiently secure encryption algorithms available to the open source community, but this can result in an increase in file size or can require a large amount of processing time. However, by selecting small but vital portions of a file and encrypting only those few bytes can render an image unusable. The initial statistical analysis done by the team will consist of specifically breaking down the standard JPEG compression scheme into its usable parts and evaluate which of the parts, if encrypted, will cause a potential user to pay for rights to the image or force subscription to the provider service.

An additional aspect of the encryption analysis will be the determination of the specific targets in the file for encryption. For example in an MPEG file there are headers that contain a small portion of the overall number of bits but which are extremely vital to the reproduction of the movie by the user. So, if certain headers were to be encrypted the percentage of the file being manipulated would be less than ten percent of the total number of bits in the file. Although only a small portion will be encrypted, the resulting damage experienced by an unauthorized user would be sufficient to cause the user to pay for the decryption package. However, there are other targets that, while they can be encrypted and will do sufficient damage, can be guessed by an

attacker. The attacker could, with some degree of effort, render the file useful without use of the decryption software. For example, if the frame rate of an MPEG file was encrypted, an attacker could try all three of most common frame rates and one of these is certain to produce the correct rate for the particular video. In the case of JPEG Selective Encryption, Team ISE will have to balance the targets for encryption against ease of simple attacks.

A permanent web site will be constructed by the team to make the software package available to anyone interested in the Team's project. As it is vital to the world of cryptography to let the community view the approach, the first form of the working prototype will be made available on the web site. From this, feedback can be received not only from the team itself, but also from the cryptography community at large.

So, following the guidelines of the ongoing MPEG research (also being guided by the sponsor), the team will study the JPEG process and earlier attempts at encryption. With the sponsor's assistance, Team ISE will devise a workable approach to handling individual JPEG images following the concept of selective encryption.

1. INTRODUCTION	1
2. TEST ENVIRONMENT	3
3. TESTS	4
3.1. Production Code Test	4
3.1.1. JPEG_ISE Constructor with Key Only	4
3.1.2. JPEG_ISE Constructor with All Parameters	5
3.1.3. Set_Key Function with Valid Key	5
3.1.4. Set_Key Function with Invalid Key	6
3.1.5. Set_Input_File_Name Function with Valid Input File	7
3.1.6. Set_Input_File_Name Function with NULL	7
3.1.7. Set_Input_File_Name Function with Non-Valid File	8
3.1.8. Set_Output_File_Name Function with Valid Output File	9
3.1.9. Set_Output_File_Name Function with NULL	9
3.1.10. Set_Output_File_Name Function with Non-Valid File	10
3.1.11. Get_Input_File_Name Function When input_file_name != NULL	11
3.1.12. Get_Input_File_Name Function When input_file_name == NULL	11
3.1.13. Get_Output_File_Name Function When input_file_name != NULL	12
3.1.14. Get_Output_File_Name Function When input_file_name == NULL	12
3.1.15. Encrypt_File Function Normal Use	13
3.1.16. Encrypt_File Function with Invalid Input File	13
3.1.17. Encrypt_File Function with Output ISE File Name Not Set	14
3.1.18. Decrypt_File Function Normal Use	14
3.1.19. Decrypt_File Function with Non-Jpeg-Ise Input File	15
3.1.20. Decrypt_File Function with Invalid Input File	15
3.1.21. Decrypt_File Function with Output File Name Not Set	16
3.1.22. Decrypt_File Function with Incorrect Key	16
3.2. Manipulator Test	17
3.2.1. Menu Options	17
3.2.2. Button Control Tests	
3.2.3. General Tests	32
3.3 Web Site Test	35
3.3.1 The Menu Frame Page	35
3.3.2 The Main Frame Pages	39
4. SUMMARY	45
5. RELATED READINGS	46

#### **1. INTRODUCTION**

Team ISE is sponsored by Assistant Professor of Computer Science, Tom Lookabaugh, at the University of Colorado: http://itd.colorado.edu/lookabaugh/. Tom Lookabaugh is currently involved in selective encryption research on standard MPEG (Moving Picture Experts Group) files and is interested in researching the application of Selective Encryption for other multimedia formats.

The goal of selective encryption is to minimize the amount of encryption applied to a file while maximizing the damage done to the image being viewed by a user not in possession of the authorized decryption package. Complete encryption is not a requirement of the process, nor is rendering the file useless to the level of complete military secrecy. It is acceptable for an attacker to be able to view portions of the file; however, the file should be distorted enough that an attacker would not wish to use the encrypted file, but would rather purchase or subscribe to the decryption method for access to the original files.

Multimedia files prove to be good subjects for selective encryption, as these files tend to be very large and employ compression algorithms that concentrate critical information in small portions of their bit stream. If the critical data in certain multimedia standards is encrypted properly, the remaining information becomes useless to those without the appropriate decryptor. There are many types of compression algorithms that fit this description, such as MPEG 1, 2 and 4 video, G.723 and G.729 video, AAC audio, MP3 audio, JPEG and JPEG2000 image formats. Applying a Selective Encryption security solution to selected multimedia formats will greatly increase the protection level of important information.

The focus of the ISE project is to research and develop an algorithm for selectively encrypting the JPEG *baseline* compression image standard. The product of the research and development will be a package that will encrypt a file so that the amount of the file being encrypted is relatively small (on the order of 1-2% of the total file). The product will be delivered in a package that will include an encryptor and a decryptor for JPEG files and a testing suite. A web site will be constructed to facilitate the delivery of the product and documentation about the process. The encryptor and decryptor will encrypt and decrypt selected targets contained within JPEG files. The ISE project will employ the AES (Advanced Encryption Standard) for our Selective Encryption algorithm. This package will be made available in a purely open source form on our final web site.

In addition to the package containing the decryptor and encryptor, Team ISE will also provide a test suite available to prospective users. The test suite will be used to aid in the research, development and testing of the team's final product. The test suite will provide the functions necessary to complete this project. First, it will allow the user to preview a standard JPEG image. Second, the test suite will break down the various portions of a JPEG image and provide the ability to manipulate the data in all of the portions. Third, after altering the data in any particular file, the test suite will provide the capability to preview the encryption attempt without the benefit of compatible decryption. Forth, the suite will have the ability to decrypt an encrypted file. The decryption options will allow the user try to defeat the encryption methods.

Any selective encryption scheme could be developed using a package that implemented these features, however, the delivered test suite will only employ the AES encryption scheme chosen by the team. The test suite will be available to download from the team web site.

The final web site will be deployed on a web server provided by the Sponsor. The machine facilitating the web server will use the Linux Red Hat 9.0 operating system platform. The team will acquire a fixed IP address from the proper University of Colorado authorities and will develop a simple web site capable of delivering information to viewers about the benefits and application of Selective Encryption technology. The site will provide users the option to download and use the final software package. The site will also provide links to important information and will remain in place as long as the sponsor deems necessary.

The final software package will accomplish the complex task of selectively encrypting a JPEG baseline standard image while providing a simple user interface. Team ISE has identified three specific types of users: high-end art users, typical Internet image users, and small, low-end image users. The research and software will be tailored to these users' needs. Figure 1.1 is a flow chart showing the general logic design of the team's final product.



Figure 1.1: Conceptual Overview of ISE Software

This document describes the test plan for the various components of the ISE project, and is used to verify that the project meets the requirements set forth in the ISE *Requirements Document*. It describes the environments, both hardware and software, necessary to test the production code, Manipulator, and web site. It then proceeds to give a detailed description of the tests themselves.

#### **2. TEST ENVIRONMENT**

This section of the text plan document outlines that Environment used to test the ISE Production Code, the ISE Manipulator, and the ISE web site. The ISE Production Code tests should be conducted in the following environment:

#### Software:

- Any Version of Redhat Linux 9.0 and higher.
- Windows 9x/ME/NT/200x/XP and higher.
- Mac OS X or higher.

#### Hardware:

- Generic Color Monitor.
- Mouse as part of the User Interface.
- Keyboard as part of the User Interface.
- Support for a 32-bit processor assembly instructions for AES optimizations.

The ISE Manipulator tests should be conducted in the following environment:

#### Software:

- Windows 9x/ME/NT/200x/XP and higher.
- Microsoft .NET Framework Version 1.1 or Higher.

#### Hardware:

- Generic Color Monitor.
- Mouse as part of the User Interface.
- Keyboard as part of the User Interface.

The ISE Web Site tests should be conducted in the following environment:

#### Software:

- Microsoft Internet Explorer 6.0 or higher.
- Netscape Navigator 6.0 or higher.
- Mozilla 1.5.1 or higher.
- Safari 1.0 or higher.
- Support for HTML version 4.01 transitional.

#### Hardware:

- Generic Color Monitor.
- Mouse as part of the User Interface.
- Keyboard as part of the User Interface.

Unless explicitly invoking any instance of the ISE products as part of a test procedure, these tests assume that an instance of the product is running.

#### **3. TESTS**

The tests are organized into three separate sections which deal with the different components of the ISE project. The sections are:

- **1. ISE Production Code**
- 2. ISE Manipulator
- 3. ISE Web Site

Each test in the Test Plan has seven components:

Purpose	The reason for the test.
Procedure	The steps to follow to conduct the test.
Expected Result	The results necessary to pass the test.
Comments	Any comments the tester might have.
Date	Date the test was conducted.
Tester	Name of the person conducting the test.
Outcome	Outcome of the test (Pass or Fail).

#### **3.1. Production Code Test**

This section of the test plan is to outline out all of the testing requirements and desired results for the ISE Production Code. To test all of the functionality provided by this class, we've have designed a set of tests to cover all of the class methods. These tests were conducted as outlined in the following sections.

#### 3.1.1. JPEG\_ISE Constructor with Key Only

Purpose:	The purpose of this test is to determine if a jpeg_ise object can be created with only an encryption/ decryption key.
Procedure:	<ol> <li>Create a pointer to a character array in a C++ program containing the desired key information.</li> <li>Call the jpeg_ise(key) constructor with this key as the only parameter.</li> </ol>
Expected Result:	A new object of type jpeg_ise will be created. The key will be set using the information passed in the parameter. A default value of NULL will be set for both the input and output file names.
Comments:	In order to use this object for encryption or decryption, the user must call the set_input_file_name() and set_output_file_name() functions to set the desired jpeg and ise files.

Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

#### 3.1.2. JPEG\_ISE Constructor with All Parameters

Purpose:	The purpose of this test is to determine if a jpeg_ise object can be created with an encryption/ decryption key as well as the input and/or output file name.
Procedure:	<ol> <li>Create three pointers to character arrays in a C++ program, the first containing the desired key information, the second containing the input file name and the output file name.</li> <li>Call the jpeg_ise() constructor with all three pointers as it's arguments.</li> </ol>
Expected Result:	A new object of type jpeg_ise will be created. The key will be set using the information passed in the first parameter. The second and third parameters will be used to set the input and output file names.
Comments:	For the input and output file names, one parameter should be a jpeg file name and the other should be an ise file name, in either order. The user can verify that the input and output files were set correctly using the get_input_file_name() and get_output_file_name() functions.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

#### 3.1.3. Set\_Key Function with Valid Key

Purpose:	The purpose of this test is to determine if a key can be
	created with a valid character string.
Procedure:	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing one or more characters indicating the desired key information.</li> <li>Call the set_key() function with this key as the only parameter.</li> </ol>
------------------	---
Expected Result:	The encryption/decryption key will be created for the object using the information in the character array. The function should return 0 to indicate that the key was successfully created for the object.
Comments:	The key information in the calling program should not be damaged or modified in any way by this function.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

# 3.1.4. Set\_Key Function with Invalid Key

Purpose:	The purpose of this test is to determine if the set_key() function exits gracefully given NULL for the key information.
Procedure:	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing NULL, which is invalid for jpeg_ise key information.</li> <li>Call the set_key() function with this key as the only parameter.</li> </ol>
Expected Result:	It should return 1 indicating an invalid key.
Comments:	If the object did not contain a valid key previous to this function call, the function will need to be called again with a valid key for the object to be used for encryption or decryption.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku

Outcome:	Pass		
3.1.5. Set_Input_File_Nam	3.1.5. Set_Input_File_Name Function with Valid Input File		
Purpose:	The purpose of this test is to determine if an input file name can be created with a valid character string.		
Procedure:	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing the desired input file name with a .jpeg, .jpg, or .ise extension.</li> <li>Call the set_input_file_name() function with this pointer as the only parameter</li> </ol>		
Expected Result:	The input file name will be created for the object using the information in the character array. The function should return 0 to indicate that the input file name was successfully created for the object.		
Comments:	The input file name information in the calling program should not be damaged or modified in any way by this function. The input file must exist and be of either ise or jpeg type.		
Date:	March 6, 2004		
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku		
Outcome:	Pass		

# 3.1.6. Set\_Input\_File\_Name Function with NULL

Purpose:	The purpose of this test is to determine if the set_input_file_name() function exits gracefully given NULL for the file name.
Procedure:	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing NULL for the input file name.</li> <li>Call the set_input_file_name() function with this pointer as the only parameter.</li> </ol>

Expected Result:	The function should exit without setting the jpeg_ise object's input file name. It should return 1 indicating an invalid file name.
Comments:	If the object did not contain a valid input file name previous to this function call, the function will need to be called again with a valid file name for the object to be used for encryption or decryption.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

# 3.1.7. Set\_Input\_File\_Name Function with Non-Valid File

Purpose:	The purpose of this test is to determine if the set_input_file_name() function exits gracefully given a non-valid file for the file name, i.e. the file is of neither jpeg nor ise type.
Procedure	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing a non-valid file for the input file name.</li> <li>Examples of non-valid files are bitmaps, text files, or any other non-jpeg or non-ise file types.</li> <li>Call the set_input_file_name() function with this pointer as the only parameter.</li> </ol>
Expected Result:	The function should exit without setting the jpeg_ise object's input file name. It should return 1 indicating an invalid file name.
Comments:	If the object did not contain a valid input file name previous to this function call, the function will need to be called again with a valid file name for the object to be used for encryption or decryption.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## 3.1.8. Set\_Output\_File\_Name Function with Valid Output File

Purpose:	The purpose of this test is to determine if an output file name can be created with a valid character string.
Procedure:	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing the desired output file name with a .jpeg, .jpg, or .ise extension.</li> <li>Call the set_output_file_name() function with this pointer as the only parameter.</li> </ol>
Expected Result:	The output file name will be created for the object using the information in the character array. The function should return 0 to indicate that the output file name was successfully created for the object.
Comments:	The output file name information in the calling program should not be damaged or modified in any way by this function. The output file must exist and be of either ise or jpeg type.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## 3.1.9. Set\_Output\_File\_Name Function with NULL

Purpose:	The purpose of this test is to determine if the set_output_file_name() function exits gracefully given NULL for the file name.
Procedure:	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing NULL for the output file name.</li> <li>Call the set_output_file_name() function with this pointer as the only parameter.</li> </ol>
Expected Result:	The function should exit without setting the jpeg_ise object's output file name. It should return 1 indicating an invalid file name.

Comments:	If the object did not contain a valid output file name previous to this function call, a default name will be created during encryption or decryption based on the input file name.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

# 3.1.10. Set\_Output\_File\_Name Function with Non-Valid File

Purpose:	The purpose of this test is to determine if the set_output_file_name() function exits gracefully given a non-valid file for the file name, i.e. the file is of neither jpeg nor ise type.
Procedure:	<ol> <li>Create a jpeg_ise object.</li> <li>Create a pointer to a character array in a C++ program containing a non-valid file for the output file name.</li> <li>Examples of non-valid files are bitmaps, text files, or any other non-jpeg or non-ise file types.</li> <li>Call the set_output_file_name() function with this pointer as the only parameter.</li> </ol>
Expected Result:	The function should exit without setting the jpeg_ise object's output file name. It should return 1 indicating an invalid file name.
Comments:	If the object did not contain a valid output file name previous to this function call, a default name will be created during encryption or decryption based on the input file name.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

Purpose:	The purpose of this test is to determine if the get_input_file_name() function returns the proper string indicating the name of the input file.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid input file.</li> <li>Call the get_input_file_name() function with no parameters.</li> </ol>
Expected Result:	The function should return a pointer to a character string containing the same name as the input file used when creating the object.
Comments:	If the input file is properly set for the jpeg_ise object, then a valid pointer to the char array will be returned.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## 3.1.11. Get\_Input\_File\_Name Function When input\_file\_name != NULL

# 3.1.12. Get\_Input\_File\_Name Function When input\_file\_name == NULL

Purpose:	The purpose of this test is to determine if the get_input_file_name() function returns NULL when the input_file_name is equal to NULL.
Procedure:	<ol> <li>Create a jpeg_ise with key only.</li> <li>Call the get_input_file_name() function with no parameters.</li> </ol>
Expected Result:	The function should return NULL.
Comments:	If the input file is not explicitly set by the user for the jpeg_ise object, then the default NULL will be returned.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

Purpose:	The purpose of this test is to determine if the get_output_file_name() function returns the proper string indicating the name of the output file.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid output file.</li> <li>Call the get_output_file_name() function with no parameters.</li> </ol>
Expected Result:	The function should return a pointer to a character string containing the same name as the input file used when creating the object.
Comments:	If the output file is properly set for the jpeg_ise object, then a valid pointer to the char array will be returned.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## 3.1.13. Get\_Output\_File\_Name Function When input\_file\_name != NULL

# 3.1.14. Get\_Output\_File\_Name Function When input\_file\_name == NULL

Purpose:	The purpose of this test is to determine if the get_output_file_name() function returns NULL when the input_file_name is equal to NULL.
Procedure:	<ol> <li>Create a jpeg_ise object with key only.</li> <li>Call the get_output_file_name() function with no parameters.</li> </ol>
Expected Result:	The function should return NULL.
Comments:	If the output file is not explicitly set by the user for the jpeg_ise object, then the default NULL will be returned.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## **3.1.15.** Encrypt\_File Function Normal Use

Purpose:	The purpose of this test is to determine if the encrypt_file() function selectively encrypts a jpeg image.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid key, input jpeg file, and output ise file.</li> <li>Call the encrypt_file() function with no parameters.</li> </ol>
Expected Result:	The function should return 0 to indicate success. The original jpeg image should be undamaged and the ISE file should contain the encrypted jpeg.
Comments:	The function should return 0 to indicate success. The original jpeg image should be undamaged and the ise file should contain the encrypted jpeg.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

# **3.1.16.** Encrypt\_File Function with Invalid Input File

Purpose:	The purpose of this test is to determine if the encrypt_file() function exits gracefully given an input file that does not exist.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid key and output ise file name and a jpeg file name that does not exist.</li> <li>Call the encrypt_file() function with no parameters.</li> </ol>
Expected Result:	The function should return 1 to indicate that the input jpeg file could not be opened. The function should then exit without encrypting any data.
Comments:	The output ise file should be empty due to the fact that no encryption was performed.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## 3.1.17. Encrypt\_File Function with Output ISE File Name Not Set

Purpose:	The purpose of this test is to determine if the encrypt_file() function calls the make_ise_file_name() function to make a default output ise file.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid key and input jpeg file. Leave the output file name to be the default NULL.</li> <li>Call the encrypt_file() function with no parameters.</li> </ol>
Expected Result:	The function should call make_ise_file_name() to create an ise file name based on the input jpeg file name. Encryption should proceed and return 0 indicating a success.
Comments:	The output ise file should be created and named based on the input jpeg file. This file will contain the encrypted jpeg file information. If the ise file could not be created for any reason, this function will return 2.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

# 3.1.18. Decrypt\_File Function Normal Use

Purpose:	The purpose of this test is to determine if the decrypt_file() function selectively decrypts an ise image.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid key, input ise file, and output jpeg file.</li> <li>Call the decrypt_file() function with no parameters.</li> </ol>
Expected Result:	The function should return 0 to indicate success. The original ise image should be undamaged and the new jpeg file should contain the exact same information as the original jpeg.
Comments:	To test if the image decrypted properly, the user can try to look at the image. Also, to make sure that there is no difference between the original and decrypted jpeg images, the user could run the Unix "diff" command on the two files.

Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## 3.1.19. Decrypt\_File Function with Non-JPEG-ISE Input File

Purpose:	The purpose of this test is to determine if the decrypt_file() function exits gracefully given an input ise file that is not an encrypted jpeg image, i.e. the ise file contains a decrypted mp3 or zip file.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid key and output jpeg file name and an ise file name that contains an encrypted mp3 or zip file.</li> <li>Call the decrypt_file() function with no parameters.</li> </ol>
Expected Result:	The function should return 5 to indicate that the input file is not jpeg-ise. The function should then exit without decrypting any data.
Comments:	Due to the fact that the only ise files that exist are all from jpegs, the tester will have to change the first byte in the ise to mimic a different ise file type.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## **3.1.20.** Decrypt\_File Function with Invalid Input File

Purpose:	The purpose of this test is to determine if the decrypt_file() function exits gracefully given an input ise file that does not exist.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid key and output jpeg file name and an ise file name that does not exist.</li> <li>Call the decrypt_file() function with no parameters.</li> </ol>

Expected Result:	The function should return 2 to indicate that the input ise file could not be opened. The function should then exit without decrypting any data.
Comments:	The output jpeg file should be empty due to the fact that no decryption was performed.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

# 3.1.21. Decrypt\_File Function with Output File Name Not Set

Purpose:	The purpose of this test is to determine if the decrypt_file() function calls the make_output_file_name() function to make a default output file.
Procedure:	<ol> <li>Create a jpeg_ise object with a valid key and input ise file. Leave the output file name to be the default NULL.</li> <li>Call the decrypt_file() function with no parameters.</li> </ol>
Expected Result:	The function should call make_output_file_name() to create an output file name based on the input ise file name. Decryption should proceed and return 0 indicating a success.
Comments:	The output jpeg file should be created and named based on the input jpeg file. This file will contain the decrypted jpeg file information. If the ise file could not be created for any reason, this function will return 2.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

## 3.1.22. Decrypt\_File Function with Incorrect Key

Purpose	The purpose of this test is to make sure that the
	decrypt_file() function does not produce a properly
	decrypted jpeg image when given an incorrect key.

Procedure:	<ol> <li>Encrypt a jpeg image and create an ise file with a valid key</li> <li>Call the set_key() function with a new valid key.</li> <li>Call decrypt_file with the ise file and the new key.</li> </ol>
Expected Result:	The function should return 0 to indicate that the file was decrypted. The new jpeg image produced should not be a valid jpeg image.
Comments:	To test that the image did not decrypted properly, the user can try to look at the new image. Also, the user could run the Unix "diff" command on the original image and the new decrypted image to see that there are differences.
Date:	March 6, 2004
Tester:	Joe Jarchow / Joseph Kadhim / Shinya Daigaku
Outcome:	Pass

### **3.2. Manipulator Test**

This section of the test plan is to list out all of the testing requirements and desired results for the ISE JPEG Manipulator. To test this massive amount of functionality, this testing breaks down into three main pieces:

- 1. Menu Options
- 2. Button Controls
- 3. General Tests

The "Menu Options" section will test all of the different menu options available in the Manipulator, like the "Save Project" or "Open Picture" options that are available. The "Button Controls" section will test all of the different button control found within the Manipulator, like "Save Project" or "Update Picture" buttons available on the Project sub-tab on the Console tab. Finally, the "General Tests" section of this document will test all the rest of the miscellaneous functionality, like if the SEP project file is set up correctly or to test if the TextBox controls are working correctly.

#### 3.2.1. Menu Options

This section of the test plan is to list out the menu functions that need to be tested. Included in this section is each of the tests, a short description of the test and the expected results.

#### **3.2.1.1. File Menu Tests**

This section of the test plan is to test all of the File Menu options. Each of the File Menu options has a test under this section.

#### 3.2.1.1.1. New Project Menu Option Test

Purpose:	To test the "New Project" menu option to make sure that a new project is created when this option is selected.
Procedure:	<ol> <li>Prior to choosing the "New Project" option, open a new picture in the Manipulator.</li> <li>Then click the "New Project" menu option under the File menu.</li> </ol>

- **Expected Result:** All of the old information in the Manipulator should be cleared out for a new project to be created and they should be prompted for a new project file name and path.
- **Comments:** This is not required to make a new project, for instance, you could just load in a picture and then click the "Save Project" option and the current information loaded into the Manipulator will be saved. This option is intended to allow the user to quickly clear out the Manipulator and start a new project.
- **Date:** March 6, 2004
- **Tester:** Geoffrey Griffith
- Outcome: Pass

#### 3.2.1.1.2. Open Project Menu Option Test

- **Purpose:** To test the "Open Project" menu option to make sure that a previously saved project is loaded into the Manipulator when this option is selected.
- Procedure:
  1. Prior to choosing the "Open Project" option, open a new picture in the Manipulator.
  2. Change a few values in some of the text controls.
  3. Then click the "Open Project" menu option under the File menu.
  4. Choose a valid SEP project file to be loaded by using the dialog box.

Expected Result:	When the "Open Project" option is selected, the user should be prompted to first save any previous information. Then, all of the old information loaded in the Manipulator should be cleared out for a project being loaded and then all previous project information should be reloaded properly.
Comments:	This test should probably be completed in conjunction with the next test, which is the "Save Project" option.
Date:	March 6, 2004
Tester:	Geoffrey Griffith

Outcome: Pass

#### 3.2.1.1.3. Save Project Menu Option Test

<b>Purpose:</b>	To test the "Save Project" menu option to make sure that a
	project is saved properly in the SEP file to be stored for future
	use.

Procedure:	1. Prior to choosing the "Save Project" option, open a new
	picture in the Manipulator.
	2. Change a few values in some of the text controls.
	3. Then click the "Save Project" menu option under the File
	menu.
	4. Choose a valid name and file path for the SEP project file to
	be created.

- **Expected Result:** When the "Save Project" option is selected, the user should be prompted to choose a location and file name for the SEP project file. Then, all of the current information loaded in the Manipulator should be saved in the project file being created.
- **Comments:** This test should probably be completed in conjunction with the next test, which is the "Open Project" menu option.
- **Date:** March 6, 2004
- **Tester:** Geoffrey Griffith
- Outcome: Pass

#### 3.2.1.1.4. Open Picture Menu Option Test

Purpose:	To test the "Open Picture" menu option to make sure that a picture and its data are properly loaded into the Manipulator.
Procedure:	<ol> <li>Have a valid JPEG image and an invalid JPEG image available.</li> <li>Try opening both, one at a time, in the Manipulator</li> </ol>
Expected Result:	The valid JPEG should be loaded into the Manipulator with all the values loaded into the interface, under the proper headings. The invalid image load attempt should generate an error message about the file structure.
Comments:	The Manipulator should not discriminate against file name, but rather the file structure. Even if the file is a valid JPEG but labeled as .BMP or some other format, the file should still load properly.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

#### **3.2.1.1.5. Update Picture Menu Option Test**

- **Purpose:** To test the "Update Picture" menu option to make sure that a picture is generated from the values that are currently loaded in the Manipulator interface (whether they are user updated or not).
- Procedure:
  1. Load a picture into the Manipulator.
  2. Before changing any values, try making a replica of the picture by choosing the "Update Picture" menu option.
  3. Then try changing some values in the Manipulator and choose the "Update Picture" option again.
  4. Using the converted program, convert all 3 files (the original and the 2 new images) and verify that both the files were created with information provided in the Manipulator.
- **Expected Result:** The first picture created should have the exact same values as original converted picture. The second picture should only have values that are different from the original where they were changed/updated in the Manipulator.

Comments:	Only change a few values at first to changed to make sure they work properly for all the fields.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

### 3.2.1.1.6. Exit Menu Option Test

Purpose:	To test the "Exit" menu option to make sure that a the user can properly exit the program.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Change a few values, but don't do anything else.</li> <li>Be sure NOT to save before hitting the "Exit" option.</li> <li>Choose the "Exit" menu option.</li> </ol>
Expected Result:	Before the application is closed, the user should be prompted to save the current information. Then, after the user has provided input, the application should be closed.
Comments:	None.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

## 3.2.1.2 Edit Menu Tests

This section of the test plan is to test all of the Edit Menu options. Each of the Edit Menu options has a test under this section.

### 3.2.1.2.1. Copy Menu Option Test

Purpose:	To test the "Copy" menu option to make sure that when text is selected, we copy it to the system clipboard.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Highlight some data values.</li> </ol>

3. Click the "Copy" menu option.4. In some other program, like notepad or word, try pasting the text in.

- **Expected Result:** The highlighted text in the Manipulator should be pasted to the new document. Also, the text in the Manipulator should remain unchanged.
- Comments: None.

Date:March 6, 2004Tester:Geoffrey GriffithOutcome:Pass

### 3.2.1.2.2. Cut Menu Option Test

Purpose:	To test the "Cut" menu option to make sure that the user can cut text out of a given field and paste it back into another.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Highlight some data values.</li> <li>Click the "Cut" menu option.</li> <li>In some other program, like notepad or word, try pasting the text in.</li> </ol>

- **Expected Result:** The highlighted text in the Manipulator should be pasted to the new document. Also, the text in the Manipulator should be removed from the text control.
- Comments: None.
- **Date:** March 6, 2004
- **Tester:** Geoffrey Griffith
- Outcome: Pass

#### 3.2.1.2.3. Paste Menu Option Test

**Purpose:** To test the "Paste" menu option to make sure that the user can paste text into the different text controls in the Manipulator.

Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Highlight some data values.</li> <li>Click the "Copy" menu option.</li> <li>Highlight some other data values.</li> <li>Click the "Paste" menu option.</li> </ol>
Expected Result:	The highlighted text in the Manipulator should be pasted to the selected text.
Comments:	If for some reason the "Copy" menu won't work, use <ctrl+c> button, which is guaranteed to work.</ctrl+c>
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

#### 3.2.1.3. View Menu Tests

This section of the test plan is to test all of the View Menu options. Each of the View Menu options has a test under this section.

## 3.2.1.3.1. Stretch Large Original Menu Option Test

Purpose:	To test the "Stretch Large Original" menu option to make sure that the user can both stretch and view normally the large original picture on the Original Picture tab.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Click on the Original Picture tab.</li> <li>Click the "Stretch Large Original" menu option several times.</li> </ol>
Expected Result:	The Large Original image should toggle between stretch mode and normal mode. Also, when the image is in stretch mode, there should be a check mark next to the menu option.
Comments:	None.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

## 3.2.1.3.2. Stretch Large Changed Menu Option Test

Purpose:	To test the "Stretch Large Changed" menu option to make sure that the user can both stretch and view normally the large changed picture on the Changed Picture tab.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Click on the Changed Picture tab.</li> <li>Click the "Stretch Large Changed" menu option several times.</li> </ol>
Expected Result:	The Large Changed image should toggle between stretch mode and normal mode. Also, when the image is in stretch mode, there should be a check mark next to the menu option.
Comments:	None.
Date:	March 6, 2004
Tester:	Geoffrey Griffith

**Outcome:** Pass

## 3.2.1.3.3. Stretch Small Original Menu Option Test

Purpose:	To test the "Stretch Small Original" menu option to make sure that the user can both stretch and view normally the small original picture on the Console tab.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Click on the Console Picture tab.</li> <li>Click the "Stretch Small Original" menu option several times.</li> </ol>
Expected Rest	<b>ult:</b> The Small Original image should toggle between stretch mode and normal mode. Also, when the image is in stretch mode,

- there should be a check mark next to the menu option.
- **Comments:** None.
- March 6, 2004 Date:
- Geoffrey Griffith **Tester:**
- **Outcome:** Pass

## 3.2.1.3.4. Stretch Small Changed Menu Option Test

Purpose:	To test the "Stretch Small Changed" menu option to make sure that the user can both stretch and view normally the small changed picture on the Console tab.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Click on the Console Picture tab.</li> <li>Click the "Stretch Small Changed" menu option several times.</li> </ol>
Expected Result:	The Small Changed image should toggle between stretch mode and normal mode. Also, when the image is in stretch mode, there should be a check mark next to the menu option.
Comments:	None.
Date:	March 6, 2004
Tester:	Geoffrey Griffith

Outcome: Pass

## **3.2.1.3.5. Stretch All Menu Option Test**

Purpose:	To test the "Stretch All" menu option to make sure that the user can both stretch and view normally all of the images in one click.
Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Click the "Stretch All" menu option several times.</li> <li>Each time you click the "Stretch All" option be sure to check all of the pictures to make sure they updated correctly.</li> </ol>
Expected Result:	The Small Changed image should toggle between stretch mode and normal mode. Also, when the image is in stretch mode, there should be a check mark next to the menu option.
Comments:	None.
Date:	March 6, 2004

- **Tester:** Geoffrey Griffith
- Outcome: Pass

#### 3.2.1.4. About Menu Tests

This section of the test plan is to test all of the About Menu options. Each of the About Menu options has a test under this section.

### 3.2.1.4.1. About Menu Option Test

Purpose:	To test the "About" menu option to make sure that the user can view the project information and the people associated with the project.
Procedure:	1. Click the "About" menu option.
Expected Result:	A new window should open up with the appropriate project information. This window should close when is it clicked on.
Comments:	Try several times in a row to make sure it works right.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

### **3.2.2. Button Control Tests**

This section of the test plan is to list out the menu functions that need to be tested. Included in this section is each of the tests, a short description of the test and the expected results.

#### 3.2.2.1. Project Sub-Tab Button Tests

This section of the test plan is to test all of the buttons on the Project sub-tab. Each of the buttons on the Project sub-tab has a test under this section.

### 3.2.2.1.1. New Project Button Test

Purpose:	To test the "New Project" button to make sure that a new project is created when this option is selected.
Procedure:	<ol> <li>Prior to clicking the "New Project" button, open a new picture in the Manipulator.</li> <li>Then click the "New Project" button under the Project sub- tab on the Console tab.</li> </ol>

Expected Result:	All of the old information in the Manipulator should be cleared out for a new project to be created and they should be prompted for a new project file name and path.
Comments:	This is not required to make a new project, for instance, you could just load in a picture and then click the "Save Project" button and the current information loaded into the Manipulator will be saved. This option is intended to allow the user to quickly clear out the Manipulator and start a new project.
Date:	March 6, 2004
Tester:	Geoffrey Griffith

**Outcome:** Pass

### 3.2.2.1.2. Load Project Button Test

Purpose:	To test the "Load Project" menu option to make sure that a previously saved project is loaded into the Manipulator when this option is selected.
Procedure:	<ol> <li>Prior to clicking the "Load Project" button, open a new picture in the Manipulator.</li> <li>Change a few values in some of the text controls.</li> <li>Then click the "Load Project" button located under the Project sub-tab under the Console tab.</li> <li>Choose a valid SEP project file to be loaded by using the dialog box.</li> </ol>
Expected Result:	When the "Load Project" button is clicked, the user should be prompted to first save any previous information. Then, all of

- Expected Result: When the "Load Project" button is clicked, the user should be prompted to first save any previous information. Then, all of the old information loaded in the Manipulator should be cleared out for a project being loaded and then all previous project information should be reloaded properly.
- **Comments:** This test should probably be completed in conjunction with the next test, which is the "Save Project" button.
- **Date:** March 6, 2004
- **Tester:** Geoffrey Griffith
- Outcome: Pass

## **3.2.2.1.3.** Save Project Button Test

Purpose:	To test the "Save Project" button to make sure that a project is saved properly in the SEP file to be stored for future use.
Procedure:	<ol> <li>Prior to clicking the "Save Project" button, open a new picture in the Manipulator.</li> <li>Change a few values in some of the text controls.</li> <li>Then click the "Save Project" button under the Project subtab under the Console.</li> <li>Choose a valid name and file path for the SEP project file to be created.</li> </ol>
Expected Result:	When the "Save Project" button is clicked, the user should be prompted to choose a location and file name for the SEP project file. Then, all of the current information loaded in the Manipulator should be saved in the project file being created.
Comments:	This test should probably be completed in conjunction with the next test, which is the "Open Project" button.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

## **3.2.2.1.4. Load Picture Button Test**

Purpose:	To test the "Load Picture" button to make sure that a picture and its data are properly loaded into the Manipulator.
Procedure:	<ol> <li>Have a valid JPEG image and an invalid JPEG image available.</li> <li>Try opening both, one at a time, in the Manipulator by clicking on the "Load Picture" button.</li> </ol>
Expected Result:	The valid JPEG should be loaded into the Manipulator with all the values loaded into the interface, under the proper headings. The invalid image load attempt should generate an error message about the file structure.
Comments:	The Manipulator should not discriminate against file name, but rather the file structure. Even if the file is a valid JPEG but

labeled as .BMP or some other format, the file should still load properly.

Date:	March 6, 2004
-------	---------------

**Tester:** Geoffrey Griffith

Outcome: Pass

#### 3.2.2.1.5. Save Picture Button Test

**Purpose:** To test the "Save Picture" button to make sure that a picture load in the changed picture image boxes are properly saved to file as a JPEG image.

Procedure:
1. Open a valid JPEG image in the Manipulator.
2. Alter a few values and create an image that is not the same, but still viewable as a JPEG image.
3. Click the "Save Picture" button located on the Project subtab of the Console tab.

- **Expected Result:** The viewable JPEG loaded into the Manipulator changed image boxes should be saved to file. The values saved should be the values that are currently loaded into the text controls (except the encoded stream) of the Manipulator.
- **Comments:** This image should be tested by trying to open the created JPEG image in a standard image viewer (or multiple viewers for that matter). This image should be viewable as normal.
- **Date:** March 6, 2004

**Tester:** Geoffrey Griffith

Outcome: Pass

### **3.2.2.1.6. Update Picture Button Test**

**Purpose:** To test the "Update Picture" button to make sure that a picture is generated from the values that are currently loaded in the Manipulator interface (whether they are user updated or not).

Procedure:	<ol> <li>Load a picture into the Manipulator.</li> <li>Before changing any values, try making a replica of the picture by clicking the "Update Picture" button.</li> <li>Then try changing some values in the Manipulator and click the "Update Picture" button again.</li> <li>Using the converted program, convert all 3 files (the original and the 2 new images) and verify that both the files were created with information provided in the Manipulator.</li> </ol>
Expected Result:	The first picture created should have the exact same values as original converted picture. The second picture should only have values that are different from the original where they were changed/updated in the Manipulator.
Comments:	Only change a few values at first to changed to make sure they work properly for all the fields. You may want to use the converter to convert the images produced hexadecimal to evaluate the data contained in the image.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

## 3.2.2.2. Huffman and Quantizer Sub-Tab Button Tests

This section of the test plan is to test all of the buttons on the Project sub-tab. Each of the buttons on the Project sub-tab has a test under this section.

#### **3.2.2.1.** Clear Button Tests

Purpose:	To test the all "Clear" buttons on their corresponding text controls on both of the Huffman sub-tabs and the Quantizer sub-tab.
Procedure:	<ol> <li>Prior to clicking the "clear" button, open a new picture in the Manipulator.</li> <li>Try altering text in each for the Huffman tables and Quantizer tables.</li> <li>Then, for each of the different Quantizer and Huffman table fields, click the corresponding clear button.</li> </ol>

- **Expected Result:** The corresponding table field should be cleared out. Also, check to make sure that click one clear doesn't affect any of the other text fields.
- **Comments:** Most images won't have 4 Quantizer and/or 8 Huffman tables, so to test the unused fields, simply type some data into the field and then hit the "Clear" button. Also, if the field hasn't been previously altered, then it original data should be moved to the corresponding original data field.
- **Date:** March 6, 2004
- **Tester:** Geoffrey Griffith
- Outcome: Pass

#### 3.2.2.2. Random Button Tests

- Purpose: To test the all "Random" buttons on their corresponding text controls on both of the Huffman sub-tabs and the Quantizer sub-tab.
  Procedure: 1. Prior to clicking the "Random" button, open a new picture in the Manipulator. 2. Try clicking the "Random" button to add a random byte onto the end of the tables.
  Expected Result: The corresponding table field should have a random byte
- **Expected Result:** The corresponding table field should have a random byte appended to the end of it. Also, make sure that if the field has not been altered previously, that the information be moved to the corresponding original data TextBox control.
- **Comments:** Most images won't have 4 Quantizer and/or 8 Huffman tables, so to test the unused fields, simply hit the "Random" button, a byte will still be added to the end of an empty table. Also, if the field hasn't been previously altered, then it original data should be moved to the corresponding original data field.
- **Date:** March 6, 2004
- **Tester:** Geoffrey Griffith
- Outcome: Pass

#### **3.2.2.3. Restore Button Tests**

Purpose:	To test the all "Restore" buttons on their corresponding text controls on both of the Huffman sub-tabs and the Quantizer sub-tab.
Procedure:	<ol> <li>Prior to clicking the "Restore" button, open a new picture in the Manipulator.</li> <li>Change some data values in the Manipulator to get the data input into the corresponding original text field.</li> <li>Try clicking the "Restore" button to restore the originally loaded data into the corresponding tables.</li> </ol>
Expected Result:	The corresponding table field should be restored to the original value loaded from the image file.
Comments:	Most images won't have 4 Quantizer and/or 8 Huffman tables, so to test the unused fields, simply hit the "Restore" button, the original table will be restored to the corresponding field.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

#### **3.2.3. General Tests**

This section of the test plan is to test all of the other functions not covered by nay other section here. Each of the test in this section reflects some piece of the manipulator that has not previously been tested by any other section in the document.

#### **3.2.3.1.** TextBox Control Test

This section of the test plan is to test all of the TextBox controls found within the Manipulator. Each of the tests are described in their following section.

#### **3.2.3.1.1.** Changeable TextBox Control Tests

Purpose:	To test the all "TextBox" controls in the Manipulator to make sure they are working properly.
Procedure:	1. Open a new picture in the Manipulator.

2. For each TextBox control that is not "grayed out," change some data values. If no data currently exists in the field, then just try adding some text into the control.

**Expected Result:** The data should be entered into the proper TextBox control, if the control is not "grayed out." If you encounter a control where this doesn't work, please write down the name of each one.

<b>Comments:</b>	None.	

**Date:** March 6, 2004

**Tester:** Geoffrey Griffith

Outcome: Pass

### 3.2.3.1.2. Non-Changeable TextBox Control Tests

- Purpose:To test the all non-changeable "TextBox" controls in the<br/>Manipulator to make sure they are working properly.
- Procedure:
  1. Open a new picture in the Manipulator.
  2. For each TextBox control that is "grayed out," try to change some data values. If no data currently exists in the field, then just try adding some text into the control.
- **Expected Result:** The data should NOT be entered into the proper TextBox control. If you encounter a control where this doesn't work, please write down the name of each one.
- **Comments:** The non-changeable fields aren't as important as the changeable ones, but they should still all be checked. This will ensure that the original data won't be destroyed, so that the user can restore it if needed.

**Date:** March 6, 2004

### **Tester:** Geoffrey Griffith

### Outcome: Pass

3.2.3.1.3. Generating	<b>New JPEG</b>	<b>Image Tests</b>
-----------------------	-----------------	--------------------

- **Purpose:** To test to make sure that the new image being created includes all of the values currently stored in the Manipulator and only those values.
- Procedure:
  1. Open a new picture in the Manipulator.
  2. Try changing a bunch of different values for the picture.
  3. Generate the new picture.
  4. Use the converter to convert the original image and the newly generated image to an ASCII file and compare all of the data values.
- **Expected Result:** The only data that should be changed in the newly generated image file from the original file is the data that was updated. Also, this updated data should be reflected in the new file as well.
- **Comments:** The Converter should be sufficient to do this, but you may also want to run the newly generated picture through an image viewer (if the new image itself is viewable). You should use the Design document to evaluate whether or not the format of this file is correct.
- **Date:** March 6, 2004
- **Tester:** Geoffrey Griffith
- Outcome: Pass

### 3.2.3.1.4. Generating New SEP Project File Tests

Purpose: To test to make sure that the new SEP file being created includes all of the values currently stored in the Manipulator and only those values.
Procedure: 1. Open a new picture in the Manipulator.
2. Try changing a bunch of different values for the picture.
3. Generated the new JPEG picture.
4. Add some project notes into the Project Notes text field.
5. Then save the SEP project file.
6. Then open the SEP file in some text processor, like NotePad or Word. You should be able to see all of the values saved in this file.

**Expected Result:** All of the changed file information should be stored in this file. Also the path and file name of both the original JPEG image and the changed JPEG image should be shown here as well. You should use the Design document to evaluate whether or not the format of this file is correct.

Comments:	None.
Date:	March 6, 2004
Tester:	Geoffrey Griffith
Outcome:	Pass

### 3.3. Web Site Test

This section of the test plan document outlines the series of tests created to test all the functionality of the ISE Website. The web site test will be broken into the following categories:

### 1. The Menu Frame Page

### 2. The Main Frame Pages

The tests and results for both of these categories are compiled in the following sections of this document.

### **3.3.1.** The Menu Frame Page

This section of the Test plan outlines tests done on the page Button.html, which appears in the Menu frame.

### **3.3.1.1.** The Home Button

Purpose:	This test is to verify that the Home button links to the correct page.
Procedure:	Click the Home button on the Menu.
Expected Result:	The page Home.html should open in the Main Frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi

Outcome:	Pass
Outcome.	1 455

# 3.3.1.2. The Project Proposal Button

Purpose:	This test is to verify that the Project Proposal button links to the correct page.
Procedure:	Click the Project Proposal button on the Menu.
Expected Result:	The document ProjectProposal.pdf should open in the Main frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## **3.3.1.3.** The Documentation Button

Purpose:	This test is to verify that the Documentation button links to the correct page.
Procedure:	Click the Documentation button on the Menu.
Expected Result:	The page DocumentIndex.html should open in the Main Frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## 3.3.1.4. The Project Sponsor Button

Purpose:	This test is to verify that the Project Sponsor button links to
	the correct page.

Procedure:	Click the Project Sponsor button on the Menu.
Expected Result:	The page Sponsor.html should open in the Main Frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## 3.3.1.5. The Team Info Button

Purpose:	This test is to verify that the Team Info button links to the correct page.
Procedure:	Click the Team Info button on the Menu.
Expected Result:	The page Team_ISE.html should open in the Main Frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## 3.3.1.6. The Download Button

Purpose:	This test is to verify that the Download button links to the correct page.
Procedure:	Click the Download button on the Menu.
Expected Result:	The page Download.html should open in the Main Frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi

<b>Outcome:</b>	Pass
0	

## 3.3.1.7. The Links Button

Purpose:	This test is to verify that the Links button links to the correct page.
Procedure:	Click the Links button on the Menu.
Expected Result:	The page Links.html should open in the Main Frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## 3.3.1.8. The Message Board Button

Purpose:	This test is to verify that the Message Board button links to the correct page.
Procedure:	Click the Message Board button on the Menu.
Expected Result:	The page index.php should open in the Main Frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## **3.3.2.** The Main Frame Pages

This section outlines the tests done on the various pages displayed in the Main frame.

## 3.3.2.1. DocumentIndex.html Requirements Button

Purpose:	This test is to verify that the Requirements button on the DocumentIndex.html page functions correctly.
Procedure:	Click the Requirements button on the page.
Expected Result:	A .pdf reader should open ISEFinalRequirements.pdf file in the Main frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

### 3.3.2.2. DocumentIndex.html Prototype Plan Button

Purpose:	This test is to verify that the Prototype Plan button on the DocumentIndex.html page functions correctly.
Procedure:	Click the Prototype Plan button on the page.
Expected Result:	A .pdf reader should open ISEPrototypePlan.pdf file in the Main frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

### 3.3.2.3. DocumentIndex.html Sys Arch Design Button

Purpose:	This test is to verify that the Sys Arch Design button on the
	DocumentIndex.html page functions correctly.

Procedure:	Click the Sys Arch Design button on the page.	
Expected Result:	A .pdf reader should open ISESystemArchitectureDesign.pdf file in the Main frame.	
Comments:	None.	
Date:	March 6, 2004	
Tester:	Andrew Pouzeshi	
Outcome:	Pass	

## 3.3.2.4. DocumentIndex.html Design Document Button

Purpose:	This test is to verify that the Design Document button on the DocumentIndex.html page functions correctly.
Procedure:	Click the Design Document button on the page.
Expected Result:	A .pdf reader should open DesignSpecFinal.pdf file in the Main frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## 3.3.2.5. Sponsor.html Project Sponsor Button

Purpose:	This test is to verify that the Project Sponsor button on the Sponsor.html page functions correctly.
Procedure:	Click the Project Sponsor button on the page.
Expected Result:	The page located at http://www.cs.colorado.edu/people/tom_lookabaugh.html should be displayed in the Main frame.
Comments:	None.

Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## 3.3.2.6. Download.html Production Code Button

Purpose:	This test is to verify that the Production Code button on the Download.html page functions correctly.
Procedure:	Click the Production Code button on the page.
Expected Result:	The browser should prompt a window asking the user where they would like to download the zip file code.zip.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

## 3.3.2.7. Download.html Manipulator Button

Purpose:	This test is to verify that the Manipulator button on the Download.html page functions correctly.
Procedure:	Click the Manipulator button on the page.
Expected Result:	The browser should prompt a window asking the user where they would like to download the zip file manipulator.zip.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass
------------------------	--
3.3.2.8. Download.html	.NET Framework Button
Purpose:	This test is to verify that the .NET Framework button on the Download.html page functions correctly.
Procedure:	Click the .NET Framework button on the page.
Expected Result:	The browser should prompt a window asking the user where they would like to download the file dotnetfx.exe.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

### 3.3.2.9. Download.html Alpha Test Button

Purpose:	This test is to verify that the .NET Framework button on the Download.html page functions correctly.
Procedure:	Click the .NET Framework button on the page.
Expected Result:	The browser should prompt a window asking the user where they would like to download the file dotnetfx.exe.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

### 3.3.2.10. Links.html www.ijg.org/ Button

Purpose:	This test is to verify that the www.ijg.org/ button on the
	Links.html page functions correctly.

Procedure:	Click the www.ijg.org/ button on the page.
Expected Result:	The page located at http://www.ijg.org should be displayed in the Main frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

### 3.3.2.11. Links.html rijndael algo Button

Purpose:	This test is to verify that the rijndael algo button on the Links.html page functions correctly.
Procedure:	Click the rijndael button on the page.
Expected Result:	The page located at http://www.esat.kuleuven.ac.be/~rijmen/rijndael/ should be displayed in the Main frame.
Comments:	None.
Date:	March 6, 2004
Tester:	Andrew Pouzeshi
Outcome:	Pass

### 3.3.2.12. Links.html Project Sponsor Button

Purpose:	This test is to verify that the Project Sponsor button on the Links.html page functions correctly.
Procedure:	Click the Project Sponsor button on the page.
Expected Result:	The page located at http://www.cs.colorado.edu/people/tom_lookabaugh.html should be displayed in the Main frame.
Comments:	None.

**Date:** March 6, 2004

Tester: Andrew Pouzeshi

Outcome: Pass

### 4. SUMMARY

This document gives a detailed test plan for the ISE Production Code, Manipulator, and the ISE web site. These tests should be sufficient to prove that the Production Code, Manipulator, and web site are functioning correctly. All of the results uncovered by the team members during the alpha testing process have been listed here as well.

### **5. RELATED READINGS**

#### [Chang and Li 96]

Chang, H. and Li, X. On the Application of Image Decomposition to Image Compression and Encryption. 1996.

Describes image degradation based on compression and encryption.

#### [Chang and Li 2000]

Chang, H. and Li, X. Partial Encryption of Compressed Images and Videos. 2000.

Describes a partial encryption scheme used on compressed multimedia files.

#### [Droogenbroek and Benedett 2002]

Droogenbroek, M. and Benedett, R. *Techniques for Selective Encryption of Uncompressed and Compressed Images*. 2002.

#### [Kailasanathan and Naini 2003]

Kailasanathan, C. and Naini, R. *Compression Performance of JPEG Encryption Scheme*. 2003.

Describes compression performance of JPEG encryption.

#### [Daigaku and Griffith and Jarchow and Kadhim and Pouzeshi]

Daigaku, S., Griffith, G., Jarchow, J., Kadhim, J. and Pouzeshi A. *Requirement Specification*. 2003.

Describes the requirement for Team ISE and for the ISE project.

#### [Daigaku and Griffith and Jarchow and Kadhim and Pouzeshi]

Daigaku, S., Griffith, G., Jarchow, J., Kadhim, J. and Pouzeshi A. *System Architecture*. 2003.

Describes the high-level system architecture for the ISE project.

#### [Li and Knipe and Cheng 97]

Li, X., Knipe, J. and Cheng, H. *Image Compression and Encryption Using Tree Structures*. 1997.

Describes compression methods that utilize tree structures.

#### [Lookabaugh and Sicker and Keaton and Guoand and Vedula 2003]

Lookabaugh, T., Sicker, D., Keaton, D., Guoand, W. and Vedula, I. Security Analysis of Selectively Encrypted MPEG-e Streams. 2003.

Description of the methods and results of applying selective encryption to MPEG-2 streams.

#### [Miano 99]

Miano, J. *Compressed Image File Formats*. Addison Wesley Longman, Inc., Reading, Massachusetts, 1999.

Provides a description of the JPEG file format.

#### [Norcen and Uhl 2003]

Norcen, R. and Uhl, A. Selective Encryption of the JPEG2000 Bitstream. 2003.

Describes a selective encryption scheme on JPEG2000 files.

#### [Pennebaker and Mitchell 93]

Pennebaker, W. and Mitchell J. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, New York, 1993,

Provides a thorough description of the JPEG file format and its components.

#### [Podesser and Schmidt and Uhl 2002]

Podesser, M., Schmidt, H. and Uhl, A. Selective Bitplane Encryption for Secure Transmission of Image Data in Mobile Environments. 2002.

Describes Bitplane Encryption.

#### [Seo and Kim and Yoo and Dey and Agrawal 2003]

Seo, Y., Kim, D., Yoo, J., Dey, S., Agrawal, A. Wavelet Domain Imag Encryption by Subband Selection and Data Bit Selection. 2003.

Describes Wavelet Domain and Data Bit encryption methods.

# ISE Class Man Pages

# Team ISE Image Selective Encryption

## Table of Contents

ise man page	1
jpeg ise man page	2
encrypt file man page	3
decrypt file man page	4
set key man page	5
set input file name man page	5
set output file name man page	7
get input file name man page8	}
get output file name man page	)

ise::ise(3)

NAME ise::ise()

#### SYNOPSIS

#include <ise.h>
ise::ise(char\* key, char\* input\_file\_name, char\* output\_file\_name);

#### DESCRIPTION

ise::ise()Only classes that extend the ise class use this constructor.An ise object is constructed with the data necessary to encrypt or decrypt a file.This constructor only requires that the key be provided. The input\_file\_name and output\_file\_name arguments are optional and will be set to a default value of NULL.For more information about the ise class, please visit http://128.138.75.184.

PRE-CONDITIONS key must be a pointer to a character string.

POST-CONDITIONS An ise object is created containing the specified data members.

#### PARAMETERS

key is a pointer to the encryption/decryption key. input\_file\_name is the name and path of the input file to be encrypted or decrypted. output\_file\_name is the file name and path for the output file generated by encryption or decryption.

EXAMPLE USEAGE Only classes that extend the ise class use this constructor.

SEE ALSO
jpeg\_ise::jpeg\_ise()

TEAM ISE

ISE Production Code Functions jpeg\_ise::jpeg\_ise(3)

NAME jpeg\_ise::jpeg\_ise()

SYNOPSIS #include <ise.h> jpeg\_ise::jpeg\_ise(char\* key, char\* input\_file\_name, char\* output\_file\_name)

#### DESCRIPTION

jpeg\_ise::jpeg\_ise() An ise object is constructed with the data necessary to encrypt or decrypt a file. This overloaded constructor only requires that key be provided. The input\_file\_name and output\_file\_name arguments are optional and will be set to a default value of NULL. For more information on the jpeg\_ise and ise classes, please visit http://128.138.75.184.

PRE-CONDITIONS key must be a pointer to a character string.

POST-CONDITIONS A jpeg\_ise object is created containing the specified data members.

#### PARAMETERS

key is a pointer to the encryption key. input\_file\_name is the name and path of the input file to be encrypted or decrypted. output\_file\_name is the file name and path for the output file generated by encryption or decryption.

EXAMPLE USEAGE // constructor with only the key jpeg\_ise MyJpegIseObj (MyKey);

// constructor with all arguments for encryption
jpeg\_ise MyJpegIseObj (MyKey, MyJpeg, MyIse);

// constructor with all arguments for decryption
jpeg\_ise MyJpegIseObj (MyKey, MyIse, MyJpeg);

SEE ALSO ise::ise()

TEAM ISE

ISE Production Code Functions jpeg\_ise::encrypt\_file(0)

NAME jpeg\_ise::encrypt\_file()

SYNOPSIS #include <ise.h> int jpeg\_ise::encrypt\_file()

#### DESCRIPTION

jpeg\_ise::encrypt\_file() The encrypt\_file() method will take a standard baseline compression jpeg file and selectively encrypt the Huffman Table frames found within the file. If the output file already exists, the existing file will be overwritten, otherwise, a new encrypted file will be created for the selectively encrypted jpeg image.

#### PRE-CONDITIONS

The input\_file\_name and key ise data memberds must be set using either the overloaded constructor or the set\_input\_file\_name(char\* name) and set\_key(char\* key) functions prior to calling this method.

This function requires that the input and output file pointers be at the head of the file.

#### POST-CONDITIONS

An encrypted file will be created with the name and path specified by the output\_file\_name data member.

If this data member is NULL, then a default file name will be created based upon the input\_file\_name data member.

#### RETURN VALUES

An integer is returned indicating a success or failure. 0 will indicate a success.

1 will indicate could not open input file name.

2 will indicate could not create ise file name.

3 will indicate could not open ise file.

#### EXAMPLE USEAGE

// MyJpegIseObj must be of type jpeg\_ise
// encrypt jpeg file to ise file
MyJpegIseObj.encrypt\_file();

SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name(), ise::set\_output\_file\_name(), ise::set\_key(), jpeg\_ise::decrypt\_file()

TEAM ISE

ISE Production Code Functions jpeg\_ise::decrypt\_file(0) NAME jpeg\_ise::decrypt\_file()

SYNOPSIS #include <ise.h> int jpeg\_ise::decrypt\_file()

#### DESCRIPTION

jpeg\_ise::decrypt\_file()

The decrypt\_file method will take a jpeg\_ise file and selectively decrypt the Huffman Table frames found within the file.

If the output file already exists, the existing file will be overwritten. Otherwise, a new file will be created for the selectively decrypted jpeg image.

#### PRE-CONDITIONS

The input\_file\_name and key ise data members must be set using either the jpeg\_ise() overloaded constructor or the set\_input\_file\_name(char\* name) and set\_key(char\* key) functions prior to calling this method.

This code requires that the input and output file pointers be at the head of the file.

#### POST-CONDITIONS

A decrypted file will be created with the name and path specified by the output\_file\_name data member.

If this data member is NULL, then a default file name will be created based upon the input\_file\_name data member.

#### RETURN VALUES

An integer is returned indicating a success or failure.
0 will indicate a success.
1 will indicate input file is not a jpeg ise file.
2 will indicate could not open ise file.
3 will indicate could not create output jpeg file.
4 will indicate could not open output jpeg file.

#### EXAMPLE USEAGE

// MyJpegIseObj must be of type jpeg\_ise
// decrypt ise file to jpeg file
MyJpegIseObj.decrypt\_file();

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name(), ise::set\_output\_file\_name(), ise::set\_key(), jpeg\_ise::encrypt\_file()

TEAM ISE

ISE Production Code Functions ise::set\_key(1)

NAME ise::set\_key()

SYNOPSIS #include <ise.h> int ise::set\_key(char\* name)

DESCRIPTION ise::set\_key() The method will use the specified name to create a valid key to be used by the ise encryption or decryption methods.

PRE-CONDITIONS name must be a pointer to a character string.

POST-CONDITIONS The key data member will be set using the new string specified. Any previous information in the key will be lost.

PARAMETERS name is a pointer to a character string containing the key information for either encryption or decryption.

RETURN VALUES An integer is returned indicating a success or failure. 0 will indicate a success. 1 will indicate an invalid key.

EXAMPLE USEAGE // create key for encryption/decryption char MyKey[] = "EnterKeyHere";

// MyIseObj must be of type ise or an inheriting class
// set the key
MyIseObj.set\_key(MyKey);

SEE ALSO
ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name(), ise::set\_output\_file\_name()

TEAM ISE

ISE Production Code Functions ise::set\_input\_file\_name(1)

NAME ise::set\_input\_file\_name()

SYNOPSIS #include <ise.h> int ise::set\_input\_file\_name(char\* name)

DESCRIPTION ise::set\_input\_file\_name() This method is used to set the input\_file\_name data member for an ise object. The method must be called prior to the encryption or decryption methods if the input\_file\_name was not specified in the constructor.

PRE-CONDITIONS name must be a pointer to a valid jpeg or ise file type.

POST-CONDITIONS The input\_file\_name data member will be set using the new string specified. Any previous data in input\_file\_name will be lost.

PARAMETERS name is a pointer to a character string containing the input\_file\_name, specifying the input file to encryption or decryption.

RETURN VALUES An integer is returned indicating a success or failure. 0 will indicate a success. 1 will indicate an invalid input file name.

EXAMPLE USEAGE // MyJpegIseIbj must be of type jpeg\_ise // set a jpeg input file for encryption MyJpegIseObj.set\_input\_file\_name(MyJpeg);

// set an ise input file for decryption
MyJpegIseObj.set\_input\_file\_name(MyISE);

SEE ALSO ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_key(), ise::set\_output\_file\_name()

ise::set\_output\_file\_name(1)

NAME ise::set\_output\_file\_name()

SYNOPSIS #include <ise.h> int ise::set\_output\_file\_name(char\* name)

DESCRIPTION ise::set\_output\_file\_name() This method is used to set the output\_file\_name ise data member.

PRE-CONDITIONS name must be a pointer to a valid jpeg or ise file type.

POST-CONDITIONS The output\_file\_name data member will be set using the new string specified. Any previous data in output\_file\_name will be lost.

PARAMETERS name is a pointer to a character string containing the output\_file\_name, specifying the output file to encryption or decryption.

RETURN VALUES An integer is returned indicating a success or failure. A 0 will indicate a success. A 1 will indicate an invalid output file name.

EXAMPLE USEAGE // MyJpegIseObj must be of type jpeg\_ise // set a jpeg output file for decryption MyJpegIseObj.set\_output\_file\_name(MyJpeg);

// set an ise output file for encryption
MyJpegIseObj.set\_output\_file\_name(MyISE);

SEE ALSO
ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_key(), ise::set\_input\_file\_name()

ise::get\_input\_file\_name(0)

NAME ise::get\_input\_file\_name()

SYNOPSIS #include <ise.h> int ise::get\_input\_file\_name()

DESCRIPTION char\* ise::get\_input\_file\_name() This is the accessor method for the input\_file\_name ise data member.

RETURN VALUES The method will return the input\_file\_name data member as a character string. If the input\_file\_name is not set, the method will return NULL.

EXAMPLE USEAGE // MyIseObj must be of type ise or an inheriting class fileName = MyIseObj.get\_input\_file\_name();

SEE ALSO
ise::get\_output\_file\_name(), ise::get\_ise\_file\_type(), ise::set\_input\_file\_name()

TEAM ISE

NAME ise::get\_output\_file\_name()

SYNOPSIS #include <ise.h> char\* ise::get\_output\_file\_name()

DESCRIPTION ise::get\_output\_file\_name() This is the accessor method for the output\_file\_name ise data member.

RETURN VALUES The method will return the output\_file\_name data member as a character string. If the output\_file\_name is not set, the method will return NULL.

EXAMPLE USEAGE // MyIseObj must be of type ise or an inheriting class. fileName = MyIseObj.get\_output\_file\_name();

SEE ALSO
ise::get\_input\_file\_name(), ise::get\_ise\_file\_type(), ise::set\_input\_file\_name()

TEAM ISE

# ISE Class Reference

# Team ISE Image Selective Encryption

## Table of Contents

ise()
jpeg_ise() 2
encrypt_file() 3
decrypt_file() 4
set_key()
<pre>set_input_file_name()6</pre>
<pre>set_output_file_name()7</pre>
get_key()
get_input_file_name()9
get_output_file_name() 10
make_ise_file_name() 11
make_output_file_name()12
get_ise_file_type() 13
example driver program

#### ise::ise(3)

#### NAME

ise::ise()

#### SYNOPSIS

#include <ise.h>

ise::ise(char\* key, char\* input\_file\_name, char\* output\_file\_name);

#### DESCRIPTION

ise::ise()

Only classes that extend the ise class use this constructor.

An **ise** object is constructed with the data necessary to encrypt or decrypt a file. This constructor only requires that the **key** be provided. The **input\_file\_name** and **output\_file\_name** arguments are optional and will be set to a default value of NULL.

For more information about the ise class, please visit http://128.138.75.184.

#### PRE-CONDITIONS

key must be a pointer to a character string.

#### **POST-CONDITIONS**

An ise object is created containing the specified data members.

#### PARAMETERS

**key** is a pointer to the encryption/decryption key.

**input\_file\_name** is the name and path of the input file to be encrypted or decrypted.

**output\_file\_name** is the file name and path for the output file generated by encryption or decryption.

#### EXAMPLE USEAGE

Only classes that extend the ise class use this constructor.

#### SEE ALSO

jpeg\_ise::jpeg\_ise()

#### NAME

jpeg\_ise::jpeg\_ise()

#### SYNOPSIS

#include <ise.h>

jpeg\_ise::jpeg\_ise(char\* key, char\* input\_file\_name, char\* output\_file\_name)

#### DESCRIPTION

jpeg\_ise::jpeg\_ise()

An ise object is constructed with the data necessary to encrypt or decrypt a file. This overloaded constructor only requires that **key** be provided.

The **input\_file\_name** and **output\_file\_name** arguments are optional and will be set to a default value of NULL.

For more information on the jpeg\_ise and ise classes, please visit http://128.138.75.184.

#### PRE-CONDITIONS

**key** must be a pointer to a character string.

#### POST-CONDITIONS

A jpeg\_ise object is created containing the specified data members.

#### PARAMETERS

**key** is a pointer to the encryption key.

**input\_file\_name** is the name and path of the input file to be encrypted or decrypted.

**output\_file\_name** is the file name and path for the output file generated by encryption or decryption.

#### EXAMPLE USEAGE

// constructor with only the key
jpeg\_ise MyJpegIseObj (MyKey);

// constructor with all arguments for encryption
jpeg\_ise MyJpegIseObj (MyKey, MyJpeg, MyIse);

// constructor with all arguments for decryption
jpeg\_ise MyJpegIseObj (MyKey, MyIse, MyJpeg);

#### SEE ALSO

ise::ise()

TEAM ISELast Change: 15 April 2004ISE Production Code Functions

jpeg\_ise::encrypt\_file(0)

#### NAME

jpeg\_ise::encrypt\_file()

#### SYNOPSIS

#include <ise.h>
int jpeg\_ise::encrypt\_file()

#### DESCRIPTION

jpeg\_ise::encrypt\_file()

The encrypt\_file() method will take a standard baseline compression jpeg file and selectively encrypt the Huffman Table frames found within the file. If the output file already exists, the existing file will be overwritten, otherwise, a new encrypted file will be created for the selectively encrypted jpeg image.

#### PRE-CONDITIONS

The **input\_file\_name** and **key** ise data memberds must be set using either the overloaded constructor or the set\_input\_file\_name(char\* **name**) and set\_key(char\* **key**) functions prior to calling this method.

This function requires that the input and output file pointers be at the head of the file.

#### POST-CONDITIONS

An encrypted file will be created with the name and path specified by the **output\_file\_name** data member.

If this data member is NULL, then a default file name will be created based upon the **input\_file\_name** data member.

#### **RETURN VALUES**

An integer is returned indicating a success or failure.

0 will indicate a success.

1 will indicate could not open input file name.

2 will indicate could not create ise file name.

3 will indicate could not open ise file.

#### EXAMPLE USEAGE

// MyJpegIseObj must be of type jpeg\_ise
// encrypt jpeg file to ise file
MyJpegIseObj.encrypt\_file();

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name(), ise::set\_output\_file\_name(), ise::set\_key(), jpeg\_ise::decrypt\_file()

jpeg\_ise::decrypt\_file(0)

ISE Production Code Functions NAME

jpeg\_ise::decrypt\_file()

#### SYNOPSIS

#include <ise.h>
int jpeg\_ise::decrypt\_file()

#### DESCRIPTION

jpeg\_ise::decrypt\_file()

The decrypt\_file method will take a jpeg\_ise file and selectively decrypt the Huffman Table frames found within the file.

If the output file already exists, the existing file will be overwritten. Otherwise, a new file will be created for the selectively decrypted jpeg image.

#### PRE-CONDITIONS

The **input\_file\_name** and **key** ise data members must be set using either the jpeg\_ise() overloaded constructor or the set\_input\_file\_name(char\* **name**) and set\_key(char\* **key**) functions prior to calling this method.

This code requires that the input and output file pointers be at the head of the file.

#### POST-CONDITIONS

A decrypted file will be created with the name and path specified by the **output\_file\_name** data member.

If this data member is NULL, then a default file name will be created based upon the **input\_file\_name** data member.

#### RETURN VALUES

An integer is returned indicating a success or failure.

0 will indicate a success.

1 will indicate input file is not a jpeg ise file.

2 will indicate could not open ise file.

3 will indicate could not create output jpeg file.

4 will indicate could not open output jpeg file.

#### EXAMPLE USEAGE

// MyJpegIseObj must be of type jpeg\_ise
// decrypt ise file to jpeg file
MyJpegIseObj.decrypt\_file();

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name(), ise::set\_output\_file\_name(), ise::set\_key(), jpeg\_ise::encrypt\_file()

#### ise::set\_key(1)

#### NAME

ise::set\_key()

#### SYNOPSIS

#include <ise.h>
int ise::set\_key(char\* name)

#### DESCRIPTION

ise::set\_key()

The method will use the specified **name** to create a valid key to be used by the ise encryption or decryption methods.

#### **PRE-CONDITIONS**

name must be a pointer to a character string.

#### POST-CONDITIONS

The **key** data member will be set using the new string specified. Any previous information in the **key** will be lost.

#### PARAMETERS

**name** is a pointer to a character string containing the key information for either encryption or decryption.

#### RETURN VALUES

An integer is returned indicating a success or failure. 0 will indicate a success. 1 will indicate an invalid key.

#### EXAMPLE USEAGE

// create key for encryption/decryption
char MyKey[] = "EnterKeyHere";

// MyIseObj must be of type ise or an inheriting class
// set the key
MyIseObj.set\_key(MyKey);

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name(), ise::set\_output\_file\_name()

#### NAME

ise::set\_input\_file\_name()

#### SYNOPSIS

#include <ise.h>
int ise::set\_input\_file\_name(char\* name)

#### DESCRIPTION

ise::set\_input\_file\_name()

This method is used to set the **input\_file\_name** data member for an ise object. The method must be called prior to the encryption or decryption methods if the **input\_file\_name** was not specified in the constructor.

#### PRE-CONDITIONS

name must be a pointer to a valid jpeg or ise file type.

#### **POST-CONDITIONS**

The **input\_file\_name** data member will be set using the new string specified. Any previous data in **input\_file\_name** will be lost.

#### PARAMETERS

**name** is a pointer to a character string containing the **input\_file\_name**, specifying the input file to encryption or decryption.

#### **RETURN VALUES**

An integer is returned indicating a success or failure. 0 will indicate a success. 1 will indicate an invalid input file name.

#### EXAMPLE USEAGE

// MyJpegIseIbj must be of type jpeg\_ise
// set a jpeg input file for encryption
MyJpegIseObj.set\_input\_file\_name(MyJpeg);

// set an ise input file for decryption
MyJpegIseObj.set\_input\_file\_name(MyISE);

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_key(), ise::set\_output\_file\_name()

ise::set\_output\_file\_name(1)

#### NAME

ise::set\_output\_file\_name()

#### SYNOPSIS

#include <ise.h>
int ise::set\_output\_file\_name(char\* name)

#### DESCRIPTION

ise::set\_output\_file\_name()
This method is used to set the output\_file\_name ise data member.

#### **PRE-CONDITIONS**

name must be a pointer to a valid jpeg or ise file type.

#### POST-CONDITIONS

The **output\_file\_name** data member will be set using the new string specified. Any previous data in **output\_file\_name** will be lost.

#### PARAMETERS

**name** is a pointer to a character string containing the **output\_file\_name**, specifying the output file to encryption or decryption.

#### **RETURN VALUES**

An integer is returned indicating a success or failure. A 0 will indicate a success. A 1 will indicate an invalid output file name.

#### EXAMPLE USEAGE

// MyJpegIseObj must be of type jpeg\_ise
// set a jpeg output file for decryption
MyJpegIseObj.set\_output\_file\_name(MyJpeg);

// set an ise output file for encryption
MyJpegIseObj.set\_output\_file\_name(MyISE);

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_key(), ise::set\_input\_file\_name()

ise::get\_key(0)

NAME

ise::get\_key()

#### SYNOPSIS

#include <ise.h>
char \* ise::get\_key()

#### DESCRIPTION

ise::get\_key()

This method will return a char pointer for the **key** string.

This is an ise class protected function, and is only called from within inheriting classes.

#### **RETURN VALUES**

This method will return a char pointer for the key string.

#### EXAMPLE USEAGE

This function is only called by inheriting classes.

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), jpeg\_ise::encrypt\_file(), jpeg\_ise::decrypt\_file(), ise::get\_key()

#### ise::get\_input\_file\_name(0)

#### NAME

ise::get\_input\_file\_name()

#### SYNOPSIS

#include <ise.h>
int ise::get\_input\_file\_name()

#### DESCRIPTION

char\* ise::get\_input\_file\_name()
This is the accessor method for the input\_file\_name ise data member.

#### **RETURN VALUES**

The method will return the **input\_file\_name** data member as a character string. If the **input\_file\_name** is not set, the method will return NULL.

#### EXAMPLE USEAGE

// MyIseObj must be of type ise or an inheriting class fileName = MyIseObj.get\_input\_file\_name();

#### SEE ALSO

ise::get\_output\_file\_name(), ise::get\_ise\_file\_type(), ise::set\_input\_file\_name()

ise::get\_output\_file\_name(0)

#### NAME

ise::get\_output\_file\_name()

#### SYNOPSIS

#include <ise.h>
char\* ise::get\_output\_file\_name()

#### DESCRIPTION

ise::get\_output\_file\_name()
This is the accessor method for the output\_file\_name ise data member.

#### **RETURN VALUES**

The method will return the **output\_file\_name** data member as a character string. If the **output\_file\_name** is not set, the method will return NULL.

#### EXAMPLE USEAGE

// MyIseObj must be of type ise or an inheriting class. fileName = MyIseObj.get\_output\_file\_name();

#### SEE ALSO

ise::get\_input\_file\_name(), ise::get\_ise\_file\_type(), ise::set\_input\_file\_name()

#### NAME

ise::make\_ise\_file\_name()

#### SYNOPSIS

#include <ise.h>
int ise::make\_ise\_file\_name()

#### DESCRIPTION

ise::make\_ise\_file\_name()

The file name and path created will be the same as the string pointed to by the **input\_file\_name** ise data member, except that the extension of the file will be changed to .ise.

If this file already exists, then a 0 will be added on to the end of the file name, just before the extension.

If this file already exists, the function will keep incrementing this number and checking, until the new file name does not previously exist.

This is an ise class private function, and is only called within the ise class.

#### PRE-CONDITIONS

The user of the class has previously set the **input\_file\_name** data member.

#### POST-CONDITIONS

The **output\_file\_name** data member points to a string with a file name and file path, based upon the string pointed to by the **input\_file\_name** data member.

#### RETURN VALUES

An integer is returned indicating a success or failure. 0 will indicate a success. 1 will indicate a failure.

#### EXAMPLE USEAGE

This function is called privately by other ise functions.

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name()

#### NAME

ise::make\_output\_file\_name()

#### SYNOPSIS

#include <ise.h>
int ise::make\_output\_file\_name();

#### DESCRIPTION

ise::make\_output\_file\_name()

The file name and path created will be the same as the string pointed to by the **input\_file\_name** ise data member, except that the extension of the file will be changed to .jpg.

If this file already exists, then a 0 will be added on to the end of the file name, just before the extension.

If this file already exists, the function will keep incrementing this number and checking, until the new file name does not previously exist.

This is an ise class private function and is only called from within the ise class.

#### PRE-CONDITIONS

The user of the class has previously set the **input\_file\_name** data member.

#### **POST-CONDITIONS**

The **output\_file\_name** data member points to a string with a file name and file path, based upon the string pointed to by the **input\_file\_name** data member.

#### RETURN VALUES

An integer is returned indicating a success or failure. 0 will indicate a success. 1 will indicate a failure.

#### EXAMPLE USEAGE

This function is called privately by other ise functions.

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), ise::set\_input\_file\_name()

#### NAME

ise::get\_ise\_file\_type()

#### SYNOPSIS

#include <ise.h>
int ise::get\_ise\_file\_type()

#### DESCRIPTION

ise::get\_ise\_file\_type()

This method will return an integer corresponding to the original file type of an encrypted ise file.

#### **RETURN VALUES**

The function will return an integer indicating the type of the original file from which the specified ise file was created.

0 will indicate an unknown or unimplemented file type.

1 will indicate a jpeg file.

The return values may be extended to accommodate other file types.

#### EXAMPLE USEAGE

// MyIseObj must be of type ise or an inheriting class. fileType = MyIseObj.get\_ise\_file\_type();

#### SEE ALSO

ise::ise(), jpeg\_ise::jpeg\_ise(), jpeg\_ise::encrypt\_file(), jpeg\_ise::decrypt\_file()

///-----/// /// File Name: Main.cpp /// /// File Description: /// /// This file is designed as an example program using the ISE class /// functionality. A user can modify this program by uncommenting the extra /// code and making this file. /// /// Project Name: Selective Encryption for JPEG Images CSCI 4308-4318: Senior Project /// /// August 2003 to May 2004 Department of Computer Science /// University of Colorado at Boulder /// /// /// Project Sponsor: Tom Lookabaugh Assistant Professor of Computer Science /// University of Colorado at Boulder /// /// /// Project Manager: Bruce Sanders University of Colorado at Boulder /// /// /// Team ISE Members: Shinya Daigaku Geoffrey Griffith /// /// Joe Jarchow /// Joseph Kadhim Andrew Pouzeshi /// /// ///-----/// /// This code is open source and may be used with no cost. /// The authors are in no way responsible for any effects /// from the usage of this code. It is provided as is with /// no warranties, protections, promises or any form of /// support. The authors would hope it would only be used /// for good purposes. Thank you. /// ///-----

#include <stdlib.h>
#include <iostream>
#include <fstream>
#include <string>
#include "rijndael-api-fst.h"
#include "ise.h"

using namespace std;

int main(int argc, char \* argv[])
{
 // key used for encryption and decryption
 char MyKey[] = "EnterKeyHere";

```
// original jpeg file to be encrypted
char * MyJpeg = "c:/ralphie.jpg";
```

// name for ise file created during encryption
char \* MyIse = "c:/gumble.ise";

// name for new jpeg after decryption
char \* finalJpeg = "c:/newralphie.jpg";

int err;

```
// call the constructor with only the key
jpeg_ise test(MyKey);
```

// call the constructor with all arguments
//jpeg\_ise test(dumKey, MyJpeg, MyIse);

```
//if (err != 0) cout << "set_key() failed!\n";</pre>
```

```
// set the input jpeg file
err = test.set_input_file_name(MyJpeg);
if (err != 0) cout << "set_input_file_name() failed!\n";</pre>
```

```
// set the output ise file
err = test.set_output_file_name(MyIse);
if(err != 0) cout << "set_output_file_name() failed!\n";</pre>
```

```
// test get input/output file names functions
cout << "Input file name: " << test.get_input_file_name() << "\n";
cout << "Output file name: " << test.get_output_file_name() << "\n";</pre>
```

// encrypt the file
err = test.encrypt\_file();

if (err != 0) cout << "Encryption failed!\n"; else cout << "File encryption successful!\n";

```
// set the output jpeg file
err = test.set_output_file_name(finalJpeg);
if(err != 0) cout << "set_output_file_name() failed!\n";</pre>
```

```
// test get input/output file names functions
cout << "Input file name: " << test.get_input_file_name() << "\n";
cout << "Output file name: " << test.get_output_file_name() << "\n";</pre>
```

```
// decrypt the file
err = test.decrypt_file();
if (err != 0) cout << "Decryption failed!\n";
else cout << "File decryption successful!\n";</pre>
```

```
return 0;
```

## Manipulator Tutorial

# Team ISE Image Selective Encryption
# **Manipulator Tutorial Manual**

# Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

101

# **Table of Contents**

Introduction	1
1. Installing the ISE JPEG Manipulator	1
2. Uninstalling the ISE JPEG Manipulator	2
3. Running the JPEG Manipulator Application	2
4. Loading a JPEG image	4
5. Manipulating JPEG Image Data	5
6. Creating an SEP Project File	7
Closing Remarks	9

### **Introduction:**

The ISE JPEG Manipulator is an application designed to allow the user to examine, manipulate and create images from the data of pre-existing JPEG images. This document is a short tutorial to provide users with step-by-step instruction for some the Manipulator's more common operations. This document uses the "splash.jpg" image included with the Manipulator in the main program directory. You can use this image to follow along with the tutorial. Also, for a complete listing of all of the specific functionality of the ISE JPEG Manipulator, be sure to refer to the Manipulator User Manual.

## **<u>1. Installing the ISE JPEG Manipulator:</u>**

The Manipulator comes prepackaged with a full installation script that performs all the necessary functions to properly install the Manipulator with a minimal degree of user effort. If a previous version of the Manipulator was installed prior to this installation, be sure to completely remove the previous version before preceding with this installation, otherwise this installation will not complete properly. Uninstalling the Manipulator is covered under section 1.4 of the user manual or section 2 of this document.

The Manipulator has several system requirements needed as a minimum to properly install and run the Manipulator. These requirements are as follows:

- 1. Microsoft Windows NT/2000/XP Operating System.
- 2. Microsoft .NET framework version 1.1.
- 3. A monitor, mouse and keyboard for the host computer.
- 4. 100 MB of free Hard Disk space.
- 5. 300 MHz or faster CPU.
- 6. 64 MB of RAM.

If your system meets all minimum system requirements and there is no previous version of the Manipulator installed, you are now ready to begin the installation process. To install the ISE JPEG Manipulator, complete the following steps:

1. If the ISE JPEG Manipulator has not been previously downloaded, be sure to download the installation package to the computer where you wish to install the program. Be sure to save the download some place that can be easily accessed, like your "Desktop." You can download the JPEG Manipulator from:

#### http://128.138.75.184/code/ISEManipulator107.zip

- 2. Once the file has completed downloading, double-click on the file to begin the installation process.
- 3. Follow through on-screen instructions to successfully complete the installation. Team ISE recommends using all of the default settings for installation.

# **<u>2. Uninstalling the ISE JPEG Manipulator:</u>**

The Manipulator installation package also includes an uninstall script to remove all of the application files, if the need arises. The uninstaller will remove all data copied and created during the installation process. Please note that images created by the user or any original JPEG pictures used will not be removed, unless they are saved within the ISE program folder (the folder created in the programs files during installation). To perform the uninstall process, complete the following steps:

- 1. Go to "Start" >> "Settings" >> "Control Panel"
- 2. Once you click on the "Control Panel," you should see the contents of the Control Panel folder. Double-click on the "Add/Remove Programs" icon.
- 3. Within the Add/Remove Programs utility, find the "JPEG Manipulator" entry and click on it to highlight it in blue and then click on the "Remove" button next to it.
- 4. Follow through the on-screen instructions to successfully complete removal of the application.

Please note that you must uninstall any previous versions of the ISE JPEG Manipulator before installing any updated versions.

#### **<u>3. Running the JPEG Manipulator Application:</u>**

Once the Manipulator has successfully completed installation, you should be able to instantiate the application without further delay. If the default installation was chosen, then the program can be invoked by going to:

Start Button >> Programs >> ISE >> JPEG >> JPEG Manipulator

Otherwise, if a different location for the program menu has been chosen, then go to the folder where the program was installed and then go to:

ISE >>> JPEG >>> JPEG Manipulator

Once the JPEG Manipulator icon is clicked, you will start the JPEG Manipulator application and can begin working with JPEG images. Although the application is designed to work specifically with the Baseline compression standard for JPEG images, the application should work with other JPEG formats as well.

Once the application has been opened, you should be at the main application window. Figure 1 is a picture of the main application window:



Figure 1: The Manipulator Main Window.

From this window you can perform almost any function within the application and this is the main window you will do work from. When the application is invoked, the "Console" tab (on the top-right of the window) will be the default tab selected and is where most of the work in the application is done. The "Console" tab has a number of sub-tabs that contain all of the data for the current project and/or images loaded within the application. Also, it is possible to view both the original and the manipulated images in a larger window by clicking on the "Original Picture" or "Manipulated Picture" tabs (respectively), located to the right of the "Console" tab. At the top of the window is a main menu to allow the user to perform common operations during use of the application.

Each of the sub-tabs found on the "Console" tab are related to the data found in a JPEG image or a Project file. The following is a brief description of each sub-tab's purpose:

- 1. "Project" sub-tab: Contains project file and project note data. Also contains buttons for performing common tasks within the manipulator.
- 2. "File Information" sub-tab: Contains original and manipulated picture file names and paths and any file comment data.
- 3. "Headers" sub-tab: Contains the data found in the SOF0 frame, if the image loaded is a Baseline standard compression.

- 4. "Huffman Tables 1 & 2" sub-tabs: Contains the compression data found in any compression frame markers ffc1 to ffcf in the JPEG image.
- 5. "Quantizer Table" sub-tab: Contains the data found in the DQT frames of the JPEG image.
- 6. "Encoded Data" sub-tab: Contains the SOS frame data and the first 20,000 bytes of the encoded data stream.
- 7. "Application Data" sub-tab: Contains the data for all of the APP frames (if included with the JPEG image).
- 8. "Misc" sub-tab: Contains any other frame data not included on any other tab.

The following sections outline the use of only some of the functionality here. Please refer to the Manipulator User Manual for a complete listing of this functionality.

## 4. Loading a JPEG image:

The best way to begin is to Load a JPEG image for editing. There are two ways to do this within the application: on the main menu or on the "Project" sub-tab located on the "Console" tab. The following are detailed, step-by-step instructions for loading images:

- 1. Click on the "File" menu option. This will cause the menu to become visible at the top of the window (shown in figure 2).
- 2. Then click on the "Load Picture" menu option (shown in figure 2). This will cause the "Open JPEG File" dialog box to appear (shown in figure 3).



Figure 2: The "File" Menu (Load Picture Highlighted)

- 3. Then select an image by browsing to it and clicking on the desired image (shown in figure 3). Only images that have a JPEG extension should be visible within this window.
- 4. Once you have selected the desired image, click on the "Open" button located on the bottom left of the "Open JPEG File" dialog box (shown in figure 3). This will cause the image to be loaded into the Manipulator's interface and both the picture and its data should be visible.



Figure 3: The "Open JPEG File" Dialog Box

Note: The image loaded here "splash.jpg" file included in the ISE folder (even though this picture shows the image in a different folder). Use this image to follow along with this tutorial.

## 5. Manipulating JPEG Image Data:

The most common function of the Manipulator (and the purpose for which it was designed) is to allow a user to manipulate data within a JPEG image and create a new image based upon the new data. Since there are many different pieces of a JPEG image, there are a large number of ways to make changes. Keep in mind that changing an image won't necessarily cause it to look different, only that the new image will contain the specified data. The following are detailed, step-by-step instructions for creating these ISE project files:

1. Once a JPEG image has been loaded in the Manipulator, click on the "Huffman Table 1" sub-tab to view the compression table data. This will cause the "Huffman Table 1" tab to be displayed, with any data that is included in the image (shown in figure 4). This data is represented by each individual byte's hexadecimal value. For example, if the table data says "0f," this means that particular byte has a value of 15 in base ten.

Project   File Information   Header   Hullman Tables 1   Hullman Tables 2   Quantizer Tab	tele Encoded Data Application Data Misc
Huffman 1: ffc4 00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00	Huffman 2: ffc4 10 00 02 01 03 03 02 04 03 05 05 04 04 00 00 01 7d 01 02 03 00
Clear Random Of US US US US US US US	Clear Autom 42 b1 c1 15 52 d1 10 24 33 62 72 82 09 0a 16 17 18 19 1a 25 26
Original 1:	Original 2:
Restore	Restore
Hullman 3. II:4 01 00 03 01 01 01 01 01 01 01 01 01 00 00 00 00	Hullman 4: II:04 11 00 02 01 02 04 04 03 04 07 05 04 04 00 01 02 77 00 01 02 03
Clear Random 04 05 06 07 08 09 0a 0b	Clear Random b1 04 05 21 31 06 12 41 51 07 61 71 13 22 32 81 08 14 42 91 a1 b1 c1 09 23 33 52 10 15 62 72 d1 0a 16 24 34 e1 25 11 17 18 191a
Original 3:	Original 4:
Restore	Rettore
,	,

Figure 4: The "Huffman Tables 1" Sub-Tab

2. Then click on any of the text boxes where the individual tables data is displayed. When any of these text boxes are selected, the original data in the table will be stored just below in the grey text box below it. This is to ensure the original data in the image can always be restored in the future and can be done at any time by clicking on the corresponding "Restore" button. Change a few values within the table to begin the process of creating a manipulated image (shown in figure 5).

Project   File Information   Headers   Huffman Tables 1   Huffman Tables 2   Quantizer Tab	ole Encoded Data Application Data Misc
Huffman 1: ffc4 Clear Random Clear Random C	Huffman 2:         ffc4         10 00 02 01 03 03 02 04 03 05 05 04 04 00 00 01 7d 01 02 03 00           Clear         Random         41 105 12 21 31 41 06 13 51 61 07 22 71 14 32 81 91 a1 08 23           42 b1 c1 15 52 d1 f0 24 33 62 72 82 09 0a 16 17 18 19 1a 25 26
Original 1: ffc4 Restore 00 00 01 05 01 01 01 01 01 00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09 0a 0b	Original 2:           Restore
Huffman 3:         ffc4         01 00 03 01 01 01 01 01 01 01 01 01 01 01 00 00	Huffman 4:         ffc4         11 00 02 01 02 04 04 03 04 07 05 04 04 00 01 02 77 00 01 02 03           Clear         Random         11 04 05 21 31 06 12 41 51 07 61 71 13 22 32 81 08 14 42 91 a1           b1 c1 09 23 33 52 f0 15 62 72 d1 0a 16 24 34 c1 25 f1 17 18 19 1a
Original 3:           Restore	Original 4:

Figure 5: Manipulated Compression Table Data

- 3. Now that we've made changes to the original image data, we can actually create a new image. Click on the "File" menu option. This will cause the menu to become visible at the top of the window (shown in figure 6).
- 4. Then click on the "Update Picture" menu option (shown in figure 6). This will cause a new file JPEG file to be generated by the manipulator, based on any of the updated picture data. Once the new image has been generated, the Manipulator will attempt to load it into the "Manipulated Picture" boxes. Depending on the changes made to the image, it may or may not be viewable, but if not, then a default message image will be loaded instead. If the image is viewable, it may be the same or it may be distorted. The picture in figure 7 represents an example of how an image may look due to distortion caused by altering the image data.



Figure 6: The "File" Menu (Update Picture Highlighted)



Figure 7: Example of an Manipulated Image

# 6. Creating an SEP Project File:

Another common function of the Manipulator is to allow the user to save information about a loaded image and any changes made to it in a project file. This project can then be loaded and updated at any time in the future, without losing data on an image that is currently a work in progress. There are two ways to do this within the application: on the main menu or on the "Project" sub-tab located on the "Console" tab. The following are detailed, step-by-step instructions for creating these ISE project files:

- 1. Click on the "File" menu option. This will cause the menu to become visible at the top of the window (shown in figure 8).
- 2. Then click on the "Save Project" menu option (shown in figure 8). This will cause the "Save SEP File" dialog box to appear (shown in figure 9).
- 3. Then create a new project by typing a name in the "File Name" text box or browse to an existing SEP file and click on it (shown in figure 9). Only images that have an SEP extension should be visible within this window.



Figure 8: The "File" Menu (Save Project Highlighted)

Save SEP File		<u>? ×</u>
Save in:	🖙 Local Disk (C:) 💽 🔶 🖆 🖽 -	
My Recent Documents Desktop My Documents My Computer	<ul> <li>cs1300</li> <li>Documents and Settings</li> <li>DXSDK</li> <li>Impressions Games</li> <li>Inetpub</li> <li>Office10</li> <li>Program Files</li> <li>t2</li> <li>Temp</li> <li>WINDOWS</li> <li>WUTemp</li> </ul>	
My Network Places	File name: NewProject	ive
	Save as type: Project files (*.SEP)	ncel

Figure 9: The "Save SEP File" Dialog Box

4. Once you have chosen the project name, click on the "Save" button located on the bottom left of the "Save SEP File" dialog box (shown in figure 9). This will cause all of the current data loaded in the Manipulator to be saved under the existing file name and path. Keep in mind that if you choose an existing file, all of the previous project file data will be overwritten with the new project information.

### **Closing Remarks:**

As you can see, the JPEG Manipulator is designed to be a straightforward, Windows-style application, to make the learning curve extremely easy. This tutorial was designed to give a basic overview of the Manipulator, but for more specific information about any of the functions more in-depth, please see the Manipulator User Manual. Also, for more information about the JPEG file format, please see the reference section of the ISE Manipulator Manual for documents pertain to this topic.

# Manipulator User Reference

# Team 1SE Image Selective Encryption

# **Manipulator Reference Manual**

# Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

0100

101

# **Table of Contents**

Introduction	ii
Chapter 1: Getting Started	1
1.1 ISE Manipulator Minimum System Requirements	1
1.2 ISE Manipulator Recommended System Requirements	1
1.3 Installing the ISE JPEG Manipulator	1
1.4 Uninstalling the ISE JPEG Manipulator	2
1.5 Running the ISE JPEG Manipulator Application	3
Chapter 2: JPEG Manipulator Functionality	4
2.1 Understanding How Data is Represented in the Application	4
2.2 The Manipulator's Main Window	4
2.3 The Main Menu of the Manipulator	5
2.3.1 The File Menu	5
2.3.2 The Edit Menu	6
2.3.3 The View Menu	6
2.3.4 The Help Menu	7
2.4 The Original Picture Tab	7
2.5 The Manipulated Picture Tab	8
2.6 The Console Tab	8
2.7 The Console Tab JPEG Data Sub-Tabs	10
2.7.1 The Project Sub-Tab	10
2.7.2 The File Information Sub-Tab	11
2.7.3 The Header Data Sub-Tab	12
2.7.4 The Huffman Table Sub-Tabs	13
2.7.5 The Quantizer Table Sub-Tab	14
2.7.6 The Encoded Data Sub-Tab	14
2.7.7 The Application Data Sub-Tab	15
2.7.8 The Misc Data Sub-Tab	16
Chapter 3: References and Related Readings	17

#### **Introduction**

The ISE JPEG Manipulator is an application designed to allow the user to examine, manipulate and create images from the data of pre-existing JPEG images. Team ISE, at the University of Colorado at Boulder, developed this software in conjunction with a research on the topic of JPEG Image Selective Encryption. The Manipulator was employed by the team to evaluate data from different JPEG images and provide support in testing different encryption schemes.

The purpose of this manual is to inform the user about the functionality of the JPEG Manipulator. Throughout the course of this manual, there will be a lot of terminology related to JPEG images. Unfortunately, this manual does not explain in-depth the about how JPEG compression formats work and assumes that the user has some prior knowledge of these concepts. If you do not have any previous experience with these types of compression formats, Team ISE recommends reviewing some of the reference material included at the end of this manual.

This software is provided by Team ISE and the University of Colorado AS IS and although it is believed to be in good working order, said members provide no guarantees and/or warranties about the Usage, Quality, Stability and/or Correctness of this software. Neither Team ISE nor the University of Colorado shall be held liable for any damages what-so-ever, arising out of or related to the use of or the inability to use the ISE JPEG Manipulator software. Anyone installing, using, or having previously installed or used the JPEG Manipulator agrees to all of these conditions. Any user should properly back-up all data before using it in conjunction with the ISE JPEG Manipulator software.

## **Chapter 1: Getting Started**

Before the user can begin using the JPEG Manipulator, there are a number of tasks that must be accomplished. This section of the user manual is to explain the system requirements, provide installation instructions and opening the application. Please read over the following information for a description of these details.

#### **<u>1.1 ISE Manipulator Minimum System Requirements:</u>**

The Manipulator has several system requirements needed as a minimum to properly install and run the Manipulator. These requirements are as follows:

- 1. Microsoft Windows NT/2000/XP Operating System.
- 2. Microsoft .NET framework version 1.1.
- 3. A monitor, mouse and keyboard for the host computer.
- 4. 100 MB of free Hard Disk space.
- 5. 300 MHz or faster CPU.
- 6. 64 MB of RAM.

#### **1.2 ISE Manipulator Recommended System Requirements:**

In addition to the minimum system requirements needed for the ISE Manipulator, Team ISE also recommends several minimum system requirements, to ensure performance. These requirements are as follows:

- 1. 1 GHz or faster CPU.
- 2. 256 MB of RAM.

#### **1.3 Installing the ISE JPEG Manipulator:**

The Manipulator comes prepackaged with a full installation script that performs all necessary functions to properly install the Manipulator with a minimal degree of user effort. If a previous version of the Manipulator was installed prior to this installation, be sure to completely remove the previous version before preceding with this installation, otherwise this installation will not complete properly. Uninstalling the Manipulator is covered under section 1.4 of this manual.

If your system meets all minimum system requirements and there is no previous version of the Manipulator installed, you are now ready to begin the installation process. To install the ISE JPEG Manipulator, complete the following steps:

- 1. If the ISE Manipulator has not been previously downloaded, be sure to download it to the computer where it is to be installed to. Be sure to save it some place that can be easily remembered, like your "Desktop."
- 2. Once the file has completed downloading, double-click on the file to begin the installation process.

3. Follow through on-screen installation instructions of the installer to successfully complete the installation (see figure 1.3.1). Team ISE recommends using all of the default settings for installation.



Figure 1.3.1 – JPEG Manipulator Installer Start Screen

#### **1.4 Uninstalling the ISE JPEG Manipulator:**

The Manipulator installation package also includes an uninstall script to remove all of the application files, if the need arises. The uninstaller will remove all data copied and created during the installation process. Please note that images created by the user or any original JPEG pictures used will not be removed, unless they are saved within the ISE program folder (the folder created in the programs files during installation). To perform the uninstall process, complete the following steps:

- 1. Go to Start >> Settings >> Control Panel
- 2. Once you click on the "Control Panel," you should see the contents of the Control Panel folder. Double-click on the "Add/Remove Programs" icon (See figure 1.4.1).
- 3. Within the Add/Remove Programs utility, find the "JPEG Manipulator" entry and click on it to highlight it in blue and then click on the "Remove" button next to it.
- 4. Follow through the on-screen instructions to successfully complete removal of the application.

Please note that you must uninstall any previous versions of the ISE JPEG Manipulator before installing any updated versions.

🐻 Add or Ren	nove Programs		
5	Currently installed programs:	Sort by: Name	•
C <u>h</u> ange or Remove	🔁 Adobe Acrobat 6.0 Professional	Size	356.00MB 🔺
Programs	Adobe Download Manager 1.2 (Remove Only)		
<b>1</b>	Adobe Photoshop 7.0	Size	136.00MB
Add New	🔛 Adobe Reader 6.0	Size	43.52MB
Programs	🔛 Adobe Reader Korean Fonts	Size	10.22MB
6	🔂 Advanced Networking Pack for Windows XP		
Add/Remove	Nattlefield 1942	Size	724.00MB
Windows	ESOps PuTTY 1.1	Size	2.39MB
components	🔀 Easy CD & DVD Creator 6	Size	706.00MB
	🐲 Emulator Driver for Visual Studio .NET 2003		
Set Pr <u>o</u> gram	🥭 Google Toolbar for Internet Explorer	Size	0.72MB
Access and Defaults	🔀 HighMAT Extension to Microsoft Windows XP CD Writing Wizard	Size	2.15MB
	[6] Internet Explorer Q832894	Size	16.38MB
	🔂 Java 2 Runtime Environment, SE v1.4.2_03	Size	129.00MB
	뤔 JPEG Manipulator Installer	Size	<u>1.07MB</u>
	Click here for support information.	Used	frequently
	To change this program or remove it from your computer, click Change or Remove.	Last Used On Change	Remove
	E MathType 5	Size	8.64MB
	Microcoft NET Eramework 1 1	Cino	27 07MD

Figure 1.4.1 – Add/Remove Program Example

### **1.5 Running the ISE JPEG Manipulator Application:**

Once the Manipulator has successfully completed installation, you should be able to instantiate the application without further delay. If the default installation was chosen, then the program can be invoked by going to:

Start Button >> Programs >> ISE >> JPEG >> JPEG Manipulator

Otherwise, if a different location for the program menu has been chosen, then go to the folder where the program was installed and then go to:

ISE >> JPEG >> JPEG Manipulator

Once the JPEG Manipulator icon has been clicked, the program will begin execution.

# **Chapter 2: JPEG Manipulator Functionality**

Once you've successfully installed the ISE JPEG Manipulator application, you can begin to work with different types of JPEG images. Although the application is designed to work specifically with the Baseline compression standard for JPEG images, the application should work with other JPEG formats as well. This section of the manual is devoted to describing the interface and functionality of the JPEG Manipulator.

#### 2.1 Understanding How Data is Represented in the Application:

Understanding the way JPEG image data is represented in the application is extremely important to use this as an effective tool. When the Manipulator loads a JPEG image, the data is broken down into each of the frames of data and then distributes to its corresponding text box on the interface. When each text box is loaded, the byte data is converted to its hexadecimal value in ASCII characters. For example, if a byte has the value of: 1111 1111 binary (0xff in Hexadecimal), then in the interface, the data will be displayed as "ff." If we look at an actual example of a Huffman table (shown in figure 2.1.1), we see that the data is loaded as 2 characters followed by a space for each byte of data in the table.



Figure 2.1.1 – Example Data in the Manipulator

As you can see by the picture above, there has been a change to 7 bytes of data in this particular Huffman table. Note that when we make a change to the data, we must represent it as a character of '0' to '9' or 'a' to 'f' for the Manipulator to properly interpret the data we have changed. Now that it has been said that data is represented in this way, we should mention that this is true for MOST of the text boxes, but not all. Certain text boxes, like the File Comments text box, outputs the data as its byte value, which allows the user to see what the data actually says, not just the byte data itself. In the following sections, each of the different text fields will be defined as to which way data is represented. In most cases, the text fields will contain the byte data converted to the ASCII representation of its value.

#### **2.2 The Manipulator's Main Window:**

Upon invoking the JPEG Manipulator application, the main window will be displayed for the user to see. The main window is also known as the Console tab (which is described in depth in section 2.6) and will provide the user access to almost every function of the Manipulator. The main window is designed to look and feel like a standard, Windows-style application. Notice that there is a main menu, tab controls to switch between the different components of the interface and closing and sizing controls for the window itself.

#### 2.3 The Main Menu of the Manipulator:

The JPEG manipulator contains a Windows-style application menu bar for performing common tasks within the Manipulator. This main menu consists of four sub-menus: File, Edit, View and Help sub-menus. Each of these sub-menus is defined in this section of the manual.

#### 2.3.1 The File Menu:

The File menu is the leftmost menu on the main window of the JPEG Manipulator application. This menu provides the user with options to Load a Picture, Update a Picture, Create a New Project, Save a Project, Load a Project and Exit the application. Please note that all of these functions (except exiting the application) can be preformed on the Project sub-tab by using the buttons available. In addition, the application can also be exited at any time by clicking on the "X" button on the top right of the main window of the application. Figure 2.3.1 shows an example of the File menu and the following is a description of each of the File menu options:

- 1. <u>Load Picture</u> Allows the user to Load a new JPEG image into the manipulator as an original image for editing. If there is an existing picture previously open, the user will be warned before the Load Picture action is taken.
- 2. <u>Update Picture</u> Allows the user to create a new manipulated image, based upon the current data in the Manipulator data fields.
- 3. <u>New Project</u> Clears out the current project data and picture data in the Manipulator.
- 4. **Open Project** Allows the user to open an existing project. If a project is already open, then the user will be warned before the Open Project action is taken.
- 5. <u>Save Project</u> Allows the user to save all of the data currently loaded in the Manipulator in an SEP project file for future use.
- 6. <u>Exit</u> Allows the User to close the JPEG Manipulator at any time.



Figure 2.3.1 – The File Menu

#### 2.3.2 The Edit Menu:

The Edit menu is the second from the leftmost menu on the main window of the JPEG Manipulator application. This menu provides the user with options to Copy, Cut and Paste data to and from the system clipboard. Please note that all of these functions can be preformed by keying in the combinations: 'ctrl+c', 'ctrl+x' or 'ctrl+v' respectively. Figure 2.3.2 shows an example of the Edit menu and the following is a description of each of the Edit menu options:

- 1. <u>**Copy**</u> Copies the currently highlighted text to the system clipboard.
- 2. <u>Cut</u> Cut the currently highlighted text and copies it to the system clipboard.
- 3. **Paste** Pastes any text currently on the system clipboard into the field currently indicated by the text cursor.



Figure 2.3.2 – The Edit Menu

#### 2.3.3 The View Menu:

The View menu is the second from the rightmost menu on the main window of the JPEG Manipulator application. Under the View menu is the Stretch Mode sub-menu. This sub-menu provides the user with options to toggle each of the picture box controls between normal and stretch mode. Figure 2.3.3 shows an example of the View menu and Stretch Mode sub-menu and the following is a description of each of the options:

- 1. <u>Large Original</u> Toggles the stretch mode for the picture box on the Original Picture tab.
- 2. <u>Large Manipulated</u> Toggles the stretch mode for the picture box on the Manipulated Picture tab.
- 3. <u>Small Original</u> Toggles the stretch mode for the Original picture box on the Console tab.
- 4. <u>Small Manipulated</u> Toggles the stretch mode for the Manipulated picture box on the Console tab.
- 5. <u>All Pictures</u> Toggles the stretch mode for each of the picture box controls in the Manipulator application.



Figure 2.3.3 – The View Menu

### 2.3.4 The Help Menu:

The Help menu is the rightmost menu on the main window of the JPEG Manipulator application. This menu provides the user with options to view the Manipulator Tutorial, Manipulator User Manual (this document) and the About window. Figure 2.3.4 shows an example of the Help menu and the following is a description of each of the Help menu options:

- 1. <u>**Tutorial**</u> Opens a new window containing the Manipulator's user tutorial document.
- 2. <u>Manual</u> Opens a new window containing the Manipulator's user manual document (i.e. this document).
- 3. <u>About</u> Opens a new window containing the Manipulator's about window to display the application and project information.



Figure 2.3.4 – The Help Menu

### 2.4 The Original Picture Tab:

The Original Picture tab allows the user to see, in a large window, the JPEG image that is currently loaded in the Manipulator. Also, this picture can be viewed in its actual size, or stretched to fit the exact size of the picture box control on this tab, by making changes in the View menu settings (as described in section 2.3). An example image is shown in figure 2.4.1 (shown in stretch mode):



Figure 2.4.1 – The Original Picture Tab

### 2.5 The Manipulated Picture Tab:

The Manipulated Picture tab allows the user to see, in a large window, the altered JPEG image that is currently loaded in the Manipulator. Also, this picture can be viewed in its actual size, or stretched to fit the exact size of the picture box control on this tab, by making changes in the View menu settings (as described in section 2.3). An example image is shown in figure 2.4.1 (shown in stretch mode):



Figure 2.5.1 – The Manipulated Picture Tab

#### 2.6 The Console Tab:

The Console tab is the real heart of the JPEG Manipulator. From the Console tab, the user can load images and projects, manipulate the data of images, create new images and save project data. The Console tab provides two smaller picture boxes for viewing the original and manipulated images, which can also be stretched by adjusting the settings on the View menu. The Console tab also has a series of sub-tabs that contain a number of text fields that store all of the image and project data. An example of the Console tab is shown in the figure 2.6.1 (note that the small original and manipulated pictures are in stretch view mode).

ISE JPEG Manipulator - Version 1.0.7 - C:\splash.jpg     File Edit View Help     Console: Driving Ricking Manipulated Ricking	
Ingina Picule Wanpulated Ficule	
Project File Information Headers Huffman Tables 1 Huffman Tables 2 Quantizer Tat	ole Encoded Data Application Data Misc
Huffman 1:         If c4         00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00	Huffman 2:         ffc4         10 00 02 01 03 03 02 04 03 05 05 04 04 00 00 01 7d 01 02 03 00           Clear         Random         42 b1 c1 15 52 d1 f0 24 33 62 72 82 09 0a 16 17 18 19 1a 25 26
Original 1: ffc4 Restore 00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00	Original 2:           Restore
Huffman 3: ffc4 Clear Random 01 00 03 01 01 01 01 01 01 01 01 01 00 00 00 00	Huffman 4:         ffc4         11 00 02 01 02 04 04 03 04 07 05 04 04 00 01 02 77 00 01 02 03           Liter         Random         11 04 05 21 31 06 12 41 51 07 61 71 13 22 32 81 08 14 42 91 a1           b1 c1 09 23 33 52 f0 15 62 72 d1 0a 16 24 34 e1 25 f1 17 18 19 1a
Original 3: Restore	Original 4:

Figure 2.6.1 – The Console Tab

#### 2.7 The Console Tab JPEG Data Sub-Tabs:

As shown in the picture above, the Console tab has a number of sub-tabs that contain all of the data loaded from the JPEG image and any project information. From these data sub-tabs, it is possible to change any of the data within the JPEG image. Each of these tabs are described in detail in this section of the manual. As stated in section 2.1, the data is represented a few different ways, depending on the type of image data that it happens to be. The following subsections will specify how the data for each of the data fields in the Manipulator is handled the data and how it will be treated when a manipulated image is encoded.

#### 2.7.1 The Project Sub-Tab:

The Project sub-tab allows the user to view the path and file name of the current project loaded (if any) and any project notes. Notice that the Project file name and the Project notes each have their own text fields. In addition, you can see six buttons here, which perform some of the functionality that can be found on the main menu (as described in section 2.3). Figure 2.7.1 is an example of what Project sub-tab looks like:

roject File Path:	C:\SomeProjectName.SEP	New Project
roject Notes:	This is a test of the notes field in the JPEG Manipulator. Notice that the notes can appear on multiple lines of the text field. Notice that the data in this field is NOT required to be in Hexadecimal format! Any notes in this field will be saved in the project file for future use =]	Open Project Save Project Load Picture Save Picture

Figure 2.7.1 – The Project Sub-Tab

Notice that data stored in both of these fields can have any ASCII value and are not required to be entered as the hexadecimal characters as explained in section 3.1 of this document. Each of the six buttons found on this sub-tab performs the each of the respective functions:

- 1. <u>New Project</u> Clears out the current project data and picture data in the Manipulator.
- 2. <u>Open Project</u> Allows the user to open an existing project. If a project is already open, then the user will be warned before the Open Project action is taken.
- 3. <u>Save Project</u> Allows the user to save all of the data currently loaded in the Manipulator in an SEP project file for future use.
- 4. <u>Load Picture</u> Allows the user to Load a new JPEG image into the manipulator as an original image for editing. If there is an existing image previously open, the user will be warned before the Load Picture action is taken.
- 5. <u>Save Picture</u> Allows the user to save the currently manipulated picture for future use.
- 6. <u>Update Picture</u> Allows the user to create a new manipulated picture, based upon the current data in the manipulator data fields.

#### 2.7.2 The File Information Sub-Tab:

The File Information sub-tab contains the file names of both the original image and the manipulated image, the size of the original file and the file comments in the original file. The data contained in these fields can be any ASCII character and is not limited to Hexadecimal format, as outlined in section 3.1 of this document. Also, any changed made to the data in the file comments will not be applied when the Manipulated image is generated. Figure 2.7.2 is an example of the File Information sub-tab:

Project File Information	Headers   Huffman Tables 1   Huffman Tables 2   Quantizer Table   Encoded Data   Application Data   Misc
Original File Name:	C:\splash.jpg
Manipulated File Name:	c:\splashchanged2.jpg
File Size:	33951 bytes
File Comments: (Not Saved)	

Figure 2.7.2 – The File Information Sub-Tab

### 2.7.3 The Header Data Sub-Tab:

If the current image loaded is indeed a Baseline standard compression format, then the compression table header data will be broken up and loaded into the fields on the Header sub-tab. The information displayed here will be the SOF0 frame data (which should always be the ffc0 marker), the size of the data contained in the frame, the precision data, height and width data, the number of components and finally the component definitions. If the image loaded is not the Baseline standard compression format, then the Headers tab shouldn't contain any of the JPEG image data.

In addition, it is important to understand that the data on the Header sub-tab is represented as its Hexadecimal value. This type of representation is outlined in section 3.1 of this document and any changes made to these fields should be made in the same way, as the hexadecimal format of the values you wish to change the data to. For example, the Marker field here should say "ff c0" which means that the binary value of these two bytes in the file is: "1111 1111 0110 0000." Also, any changes in the size of the data in this frame does not need to be made by the user manually as the Manipulator will calculate the new size of the frame (and number of components) when generating the manipulated image. The picture below is an example of the Header data sub-tab:

larker:	ffc0	Number Components:	03		
ize:	17	Components: Identifier, Horizontal, Vertic	cal, Q-Table:		
recision:	08	01, 2, 2, 00 02, 1, 1, 01 03, 1, 1, 01			
eight:	01 1e	0.000.000.000000			
/idth:	02 03				

Figure 2.7.3 – The Headers Sub-Tab

#### 2.7.4 The Huffman Table Sub-Tabs:

As mentioned previously, the Manipulator was designed specifically for use with the Baseline standard JPEG compression format. This format uses the Huffman compression algorithm to create the entropy encoded data stream for the JPEG image. Thus, these compression tables can be found on the two Huffman Table sub-tabs. However, even if the image doesn't use the baseline compression, any compression data encountered in the JPEG file will be stored on this sub-tab.

The Huffman Table sub-tabs contain a number of text fields and buttons for the user to interact with the image data. Each tab contains data fields for four tables, original and manipulated compression table data, the compression table maker data, and a series of buttons for manipulating the tables. When a compression frame is loaded in the manipulator, the marker is loaded in the marker field, and then the rest of the frame's data, except for the size information, is loaded into the corresponding Huffman table field. The data is loaded in Hexadecimal format and any changes made to this data should be as hexadecimal ASCII characters that represent the values of each of the bytes of the data. Section 3.1 describes in-depth how this data is represented. Also, the size of the table, then the new size data will be recalculated when the new image is generated. Notice that once the user tries to manipulate the values of one of the tables, the original data is stored in the corresponding, grayed out field just below it. There are three buttons corresponding to each of these tables:

- 1. <u>Clear</u> this button clears out all of the data in the corresponding table field.
- 2. **<u>Random</u>** this button adds a random byte of data onto the end of the field.
- 3. **<u>Restore</u>** this button restores the original data from the image to the corresponding compression table.

Figure 2.7.4 is an example of the Huffman Table 1 data sub-tab:

Project File Informati	on Headers Huffman Tables 1 Huffman Tables 2 Quantizer Ta	ble 📔 Encoded Data 🗍 /	Application Data Misc
Huffman 1: ffc4 Clear Random	00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00	Huffman 2: ffc4	10 00 02 01 03 03 02 04 03 05 05 04 04 00 00 01 7d 01 02 03 00 04 11 05 12 21 31 41 06 13 51 61 07 22 71 14 32 81 91 a1 08 23 42 b1 c1 15 52 d1 f0 24 33 62 72 82 09 0a 16 17 18 19 1a 25 26
Original 1: ffc4 Restore	00 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00	Original 2: Restore	
Huffman 3: ffc4 Clear Random	01 00 03 01 01 01 01 01 01 01 01 01 01 00 00 00	Huffman 4: ffc4 Clear Random	11 00 02 01 02 04 04 03 04 07 05 04 04 00 01 02 77 00 01 02 03 11 04 05 21 31 06 12 41 51 07 61 71 13 22 32 81 08 14 42 91 a1 b1 c1 09 23 33 52 f0 15 62 72 d1 0a 16 24 34 e1 25 f1 17 18 19 1a
Original 3: Restore		Original 4: Restore	

Figure 2.7.4 – The Huffman Tables Sub-Tabs

#### 2.7.5 The Quantizer Table Sub-Tab:

The Quantizer table sub-tab contains all of the data for the DQT frames (i.e. the Quantizer tables) contained in the JPEG image. As defined by the JPEG standard, each of these tables should be exactly 64 bytes long for all Baseline compression images. Each of the Quantizer's marker data, table data and table number is displayed here. In addition, there are several buttons to interact with the data.

When the Quantizer frame is loaded in the manipulator, the marker is loaded in the marker field, and then the rest of the frame's data, except for the size information and the table information (i.e. all of the DTC coefficients) are loaded into the corresponding Quantizer table field. The data is loaded in Hexadecimal format and any changes made to this data should be as hexadecimal ASCII characters that represent the values of each of the bytes of the data. Section 3.1 describes in-depth how this data is represented. Also, the size of the frame is purposefully removed from this table so that if the user changes the size of the table (even though the Baseline standard dictates that these tables have exactly 64 coefficients), then the new size data will be recalculated when the new image is generated. Notice that once the user tries to manipulate the values of one of the tables, the original data is stored in the corresponding, grayed out field just below it. The three buttons corresponding to each of these tables has the following function:

- 1. <u>Clear</u> this button clears out all of the data in the corresponding table field.
- 2. **<u>Random</u>** this button adds a random byte of data onto the end of the field.
- 3. <u>**Restore**</u> this button restores the original data from the image to the corresponding Quantizer table.

uantizer 1: ffdb		Quantizer 2: ffd	
adriator 1. map	08 06 06 07 06 05 08 07 07 07 09 09 08 0a 0c 14 0d 0c 0b 0b 0c	Gudrinzor z. Indi	<sup>6</sup> 09 09 09 0c 0b 0c 18 0d 0d 18 32 21 1c 21 32 32 32 32 32 32 32 32 32 32 32 32 32
able #: 00	1c 28 37 29 2c 30 31 34 34 34 1f 27 39 3d 38 32 3c 2e 33 34 32	Table #: 01	32 32 32 32 32 32 32 32 32 32 32 32 32 3
lear Random		Clear Randon	
lateral 1.		04444.0	
iginal I.		Unginal 2.	
Restore	1	Restore	
uantizer 3:		Quantizer 4:	1
ible #:		Table #:	1
lear Random		Clear Randon	n
ininal 2		Original 4:	

The picture below is an example of the Quantizer Table sub-tab:

Figure 2.7.5 – The Quantizer Table Sub-Tab

#### 2.7.6 The Encoded Data Sub-Tab:

The Encoded Data sub-tab contains all of the SOS frame data (i.e. Scan Header data), as well as up to the first 20 KB of the entropy encoded data stream. Although there are fields for the original data, any changes made to the data on this sub-tab will not be applied to the Manipulated picture as this functionality will have to be left for a future

software enhancements. Therefore, any changes made on the text fields will not be accounted for.

When the Scan Header frame and the Entropy Encoded data is loaded in the manipulator, everything except the frame marker is stored on Encoded Data sub-tab. The data is loaded in Hexadecimal format, although any changes made to this data will not be accounted for. Section 3.1 describes in-depth how this data is represented. Also, keep in mind that only the first 20 KB of the Encoded Data is shown in the encoded data field. Figure 2.7.6 is an example of Encoded Data sub-tab:

ncoded Data:	Scan Header.	03 01 00 02 11 03 11 00 37 00
te8a28a0028a28a0028a5029769ed400d 1460d020a052e28c5200a29714b8a0425 51494b484385381a68a514896480d480d 158b461244ea6a65a85063ad4c98ac9a3	a29fb49ed46c3d718a006514e2b8a3026 029db6942d00369714f0b4ed94808f14b 4429e2a5a25a265352a9a80548b50d19 192265a916a341935305acd98b438548	801b453/6526ded400da29db69769a006d14ec7b51b681094669714a0520128a5c518a004a5a s8a90252eca008b14b8a93652eca04458a31526ca365210cc528a5da68db40094b46297148419 jb45946a9d4d555353a564d19491690d5846aa89d78ab282b29230922ca1a9d0d564cfa55841 329aab9e7b54817d2a190c514ea02d28a86431c0538520a70152431714518a2908f07a28a2beb8
iginal Encoded Data:	Original Header:	<u></u>

Figure 2.7.6 – The Encoded Data Sub-Tab

#### 2.7.7 The Application Data Sub-Tab:

The Application Data sub-tab contains all of the data found in APP frames (i.e. Application Data frames) of the JPEG image. Although there are up to sixteen application frames allowed in the Baseline compression standard, there are only ten spaces available on the Application data tab. Since application data isn't critical to the representation of the particular image, the decision was made to only include up to ten frames in the Manipulator (in all the random samples collected during the development of the Manipulator, none contained more than ten, in fact all of the images contained much less).

When the Application Data frame is loaded in the manipulator, the marker is loaded in the Marker field, and then the rest of the frame's data, except for the size information is loaded into the corresponding text field. The data is loaded in Hexadecimal format and any changes made to this data should be as hexadecimal ASCII characters that represent each of the bytes of the data. Section 3.1 describes in-depth how this data is represented. The size of the frame is purposefully removed from this table so that if the user changes the size of the table, then the new size data will be recalculated automatically when the new image is generated. Figure 2.7.7 is an example of the Application Data sub-tab:

Project   File Information   Headers   Huffman Tables 1   Huffman Tables 2	Quantizer Table Encoded Data Application Data Misc	
App Data 1: ffe0 4a 46 49 46 00 01 01 01 00 48 00 48 00 00	App Data 2:	
App Data 3:	App Data 4:	
App Data 5:	App Data 6:	
App Data 7:	App Data 8:	
App Data 9:	App Data 10:	

Figure 2.7.7 – The Application Data Sub-Tab

#### 2.7.8 The Misc Data Sub-Tab:

The Misc sub-tab contains the data of the rest of the possible frames allowed within a JPEG image. Much of this data is not allowed for the Baseline compression standard, but to make sure that the Manipulator would be compliant for all standards currently defined, these data fields were included. The Misc sub-tab contain data fields for the marker data and frame data for the restart interval frame, marker data and frame data for the number of lines frame, marker data and frame data for any hierarchical progression frame, data for the Restart Mod 8 frame and a field for any program errors generated by the Manipulator during execution.

When the frames on the Misc sub-tab are loaded in the Manipulator, the marker is loaded in the marker field (except for the Restart Mod 8 field), and then the rest of the frame's data, except for the size information is loaded into the corresponding text field. The data is loaded in Hexadecimal format and any changes made to this data should be as hexadecimal ASCII characters that represent the values of each of the bytes of the data. Section 3.1 describes in-depth how this data is represented. The size of the frame is purposefully removed from these tables so that if the user changes the size of the table, then the new size data will be recalculated automatically when the new image is generated. The picture below is an example of the Misc Data sub-tab:

Project   File Information   Headers   Huffman Tables 1   Huffman Tables 2	Quantizer Table Encoded Data Application Data Misc
Restart Interval:	Restart Modulo 8 occured at byte index. Hierarchial Progression:
Expand Image Program Errors:	

Figure 2.7.8 – The Misc Sub-Tab

#### **Chapter 3: References and Related Readings**

There were a number of documents and research papers used by Team ISE in completing this piece of software and this entire project. This section of the document is devoted to giving credit to all of those extremely helpful references. Team ISE recommends looking over the following data if you are unfamiliar with the JPEG file formats or the concept of Selective Encryption.

- Chang, H. and Li, X. <u>On the Application of Image Decomposition to Image Compression and Encryption.</u> Research Paper. 1996.
- Chang, H. and Li, X. <u>Partial Encryption of Compressed Images and Videos.</u> Research Paper. 2000.
- Droogenbroek, M. and Benedett, R. <u>Techniques for Selective Encryption of Uncompressed and</u> <u>Compressed Images.</u> Research Paper. 2002.
- Kailasanathan, C. and Naini, R. <u>Compression Performance of JPEG Encryption Scheme</u>. Research Paper. 2003.
- Daigaku, S., et al. Requirement Specification. 2003.

Daigaku, S., et al. Selective Encryption of JPEG Standard Baseline Compression Images. 2004.

- Daigaku, S., et al. System Architecture. 2003.
- Li, X., Knipe, J. and Cheng, H. <u>Image Compression and Encryption Using Tree Structures</u>. Research Paper. 1997.
- Lookabaugh, T., et al. <u>Security Analysis of Selectively Encrypted MPEG-e Streams.</u> Research Paper. 2003.
- Miano, J. <u>Compressed Image File Formats.</u> Massachusetts: Addison Wesley Longman, Inc., 1999.
- Norcen, R. and Uhl, A. <u>Selective Encryption of the JPEG2000 Bitstream.</u> Research Paper. 2003.
- Pennebaker, W. and Mitchell J. JPEG Still Image Data Compression Standard. New York: Van Nostrand Reinhold, 1993.
- Podesser, M., Schmidt, H. and Uhl, A. <u>Selective Bitplane Encryption for Secure Transmission</u> of Image Data in Mobile Environments. Research Paper. 2002.
- Seo, Y., et al. <u>Wavelet Domain Imag Encryption by Subband Selection and Data Bit Selection</u>. Research Paper. 2003.

# Final Demo Presentation

# Team ISE Image Selective Encryption



# Team ISE Image Selective Encryption

**Joe Jarchow Joseph Kadhim Geoffrey Griffith** Shinya Daigaku **Andrew Pouzeshi** 



**Presentation Overview:** 

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- Conclusion

**Outline of Presentation** 

Joe J


# **Presentation Overview:**

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- •Future efforts



Overview

Joe J

### **Problem:**

- Multimedia files are very large
- Encryption is expensive
  - Processing time
  - •File size
- No widely accepted solutions
  - •Encrypt entire file
  - No encryption



**Problem Statement** 

### **Solution:**

Selective Encryption

**Definition from MPEG paper:** 

Selective encryption applies encryption to a subset of a file with the expectation that the entire file will be rendered useless to anyone who cannot decrypt that subset.

Solution

Joe J





Requirements

Joe J

### **Criteria For Bad Targets:**

- Optional markers
- Not used in Baseline JPEG images
- No effect on image quality
- Easily guessed or forged by a hacker



### **Statistical Analysis**

Joe J



## **Presentation Overview:**

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- Conclusion





Outline

### **Demonstration of Manipulator:**

- Layout vs. JPEG standard
- Show new features (project file, etc.)
- Cover earlier research
- Propose possible attacks
- Show table manipulation
- Show table replacement

### Demonstration





**Outline** 

# **Presentation Overview:**

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- Conclusion



## Joseph K

### **Demonstration of Production Code:**

- Run demonstration script
- During run, show code (h, cpp, scripts)
- Explain tests run in script (diff)
- Show images for comparison
- Show .ise will not work

Joseph K



# **Presentation Overview:**

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- Conclusion



**Statistical Analysis** 



## **Demonstration of Manipulator:**

- Show menu bar links
- Show each page
- Show message board
- Show message board administration
- Show HTML code
- •Plead for domain NAME!

### Demonstration





# **Presentation Overview:**

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- Conclusion



**Statistical Analysis** 



# **Encryption Algorithm:**

- •Write file-type-byte to ".ise" file
  - •'1' for JPEG
- Read from input file
- •Write unencrypted to ".ise" file
- Read/Write until Huffman
  - •[FF CO or FF C4]
  - baseline standard Huffman tables





Selective Encryption Algorithms

Shinya





### **Encryption Algorithm:**

- •Keep encrypting until encoded data •[FF DA]
  - Start of encoded data stream
- •Hide marker inside encrypted area
- Hide random length of encoded data







## **Decryption Algorithm:**

- Read file-type-byte from ".ise" file
  - '1' for JPEG
- Read/Write until Huffman
  - •[FF CO or FFC4]
  - baseline standard Huffman tables
- Start decrypting





### **Selective Encryption Algorithms**

Shinya



**Selective Encryption Algorithms** 

Shinya



# **Presentation Overview:**

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- Conclusion

**Statistical Analysis** 

Joe J

### **Potential Attacks:**

- Brute force replacement
- Inside knowledge
  - Password
  - Image editor
- Could implement AES with larger
  - •Key
  - Block length (further into data)
- Data is relatively untouched
  - •Except at head

Vulnerability

Joe J



# **Presentation Overview:**

- Overview of Project
- Demonstration of
  - Manipulator
  - Production Code
  - •Web Site
- Algorithm Design
- Potential Attacks
- Conclusion

# 0011 1400 0100 0101

**Statistical Analysis** 

All

# Questions

### Questions



A

# Developer's Reference

# Team ISE Image Selective Encryption

Developer's Reference 30 April 2004

## Team ISE Image Selective Encryption

CSCI 4308-4318, Software Engineering Project Department of Computer Science University of Colorado at Boulder

Sponsored by: Tom Lookabaugh Assistant Professor of Computer Science Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi

### **Table of Contents**

Introduction	1
ISE Moninulator	2
	2
What is the ISE JPEG Manipulator?	2
What does each of the code files do?	2
Where can I find an in-depth design of the Manipulator?	2
What do I need to compile the Manipulator code?	3
What version of .NET was used for the Manipulator?	3
ISE Production Code	4
What is the ISE Class?	4
How can I extend the ISE Class?	4
Do I need to make my own constructor?	4
Why are there only two files associated with an ISE object?	5
What methods does my inherited class have available to it?	5
What methods will my inherited class need to implement?	5
What changes to the ISE class should I make to implement my selective encryption class?	5
How do I design a selective encryption algorithm for my media file type?	6
What encryption should I use in my algorithm?	6
Is there an example of a class inherited from the ISE Class?	6
Is there an example of how to use the JPEG ISE Class?	6
Where can I find more information about Team ISE and the ISE Production Code?	6
ISE Website	7
<u>How are the passwords obtained?</u>	0
Whet server is used?	0
What OS is the server music or 2	0
What US is the server running on?	ð
What is the pain to the directory with the web pages of the server?	0
Whet is the ID address of the website?	9
What is the IP address of the website?	9
What vargion of LTML is used?	10
what version of HTML is used?	10
Are the pages written in valid HTML?	10
What character set are the pages coded in?	10
Do the pages make use of a CSS?	11
Do the pages follow a particular format?	11
What language was used to create the menu bar?	12
How can the menu bar be modified?	12
How is the menu bar included in the HTML pages?	12
Where was the message board obtained?	13
What is the message board coded in?	13
What type of database does the message board use?	13
Can the database be modified?	13

How can the database be copied?	13
How are the message board administrator options accessed?	13
How are new categories added?	14
How are categories deleted or edited?	14
How are new forums added?	14
How are forums edited?	14
How are forums deleted?	15
Can threads be automatically deleted?	15
Can the administrator manage the user accounts?	15
How are E-mail addresses banned?	15
How are E-mail addresses un-banned?	16
How are IP addresses banned?	16
How are IP addresses un-banned?	16
How are ranks set?	16
How are ranks deleted?	16

Conclusion	

#### **TEAM ISE DEVELOPER'S REFERENCE**

Team ISE has deemed it necessary to supply those who will follow in our footsteps with answers to questions that will inevitably arise. Therefore, we have created a document broken down into three sections, one for each of the main products that we produced over the last year. The sections cover questions over the ISE Manipulator, Production Code, and Website. We also recommend viewing all of the documentation on our <u>website</u>. You can view to each section by clicking on one of the following links:

#### ISE Manipulator ISE Production Code ISE Website

Also, Dr. Tom Lookabaugh at the University of Colorado at Boulder is responsible for maintaining all of the code and research produced by Team ISE. If you have questions or comments about any of the code, documentation or other items created by Team ISE, please contact Dr. Lookabaugh using the contact information given on the <u>ISE Website</u>. We hope you have as much fun and learn as much as we did – Good Luck!!!

Sincerely,

Team ISE CSCI 4308 – 4318: Senior Project University of Colorado at Boulder

#### **ISE Manipulator**

What is the ISE JPEG Manipulator?What does each of the code files do?Where can I find an in-depth design of the Manipulator?What do I need to compile the Manipulator code?What version of .NET was used for the Manipulator?

#### What is the ISE JPEG Manipulator?

The ISE JPEG Manipulator is a Windows application written in the C# (pronounced "Cee-Sharp") programming language and is design to allow the user to analyze and create JPEG images, based on the data from pre-existing JPEG images. The JPEG Manipulator was designed to help Team ISE with research during the course of the project. The Manipulator is extremely useful in determining how data (random or chosen) will react when integrated into a JPEG image. It is also a nice tool for analyzing the file structure of a JPEG image.

#### What does each of the code files do?

There are a total of four necessary code files included with the JPEG Manipulator. First, the most used code file is the frmMain.cs. This is the Main Form of the JPEG Manipulator and contains the frmMain form class and all of the code necessary to implement the base functionality of the Manipulator. The frmMain form class is directly inherited from the System. Windows. Forms class and provides the additional functionality as a series of methods in the class. Second, frmLoad.cs is the code file for the frmLoad form class inherited directly from the System.Windows.Forms class. This code was developed to provide a loading form to be displayed when a new image is loaded into the Manipulator. All of the functionality for this class is contained within the frmLoad.cs file and can be used as a separate .NET component. Third, the frmAbout.cs contains all of the code necessary to implement the frmAbout form class in the Manipulator. This class is also inherited directly form the System.Windows.Forms class. The fourth file is the frmSplash.cs code file. This file contains all of the necessary code to implement the frmSplash form class and is directly inherited from the System.Windows.Forms class. Please note that the latter 3 files are quite small, only a couple hundred of lines of code each, and only implement the functionality for the form's purpose. back∌

#### Where can I find an in-depth design of the Manipulator?

There are several documents produced by Team ISE that explain the design for the ISE Manipulator. The ISE System Architecture document provides a nice high-level design of the ISE Manipulator. In addition, the ISE Design Specification provides an in-depth, high-level design as well as a description of each of the Methods necessary to fully develop the frmMain form. Also, for the convenience of future developers, the ISE Manipulator code, object and DLL files contain XML comments for each of the Methods

in the frmMain class, so that Visual Studio's Intellisense will display those comments (along with parameter details and return value information) in the comment viewer and in the Visual Studio Object Brower. This documentation is available at the <u>website</u>. <u>back  $\frac{1}{2}$ </u>

#### What do I need to compile the Manipulator code?

Although it is not required, Team ISE highly recommends working with Visual Studio .NET when creating and compiling this code. We have included a Visual Studio Solution file that contains all of the necessary file data to create, manage and build these codes very easily. This Solution file contains two Project files, one for the Manipulator itself and one for the Manipulator Installer package. But, since this is a .NET application, the only software truly necessary to build this program is the .NET framework. If the development machine does not have Visual Studio available, the developer always has the option of building the executable from calls to the command line compiler built into the .NET framework. For more information about using the command line compiler, please see <u>www.microsoft.com/msdn</u>. Again, Team ISE recommends using Visual Studio to manage and build these code files using the Solution file provided with the final distribution package.

<u>back∳</u>

#### What version of .NET was used for the Manipulator?

Version 1.1 of the .NET framework was used to develop the ISE Manipulator, but we believe it could be compiled with ANY version of the .NET framework (including the older 1.0 version). We have included the .NET framework re-distributable package for Windows with the final distribution package. Also, the .NET framework and can be attained by downloading it from <u>Microsoft</u>, if unavailable otherwise. back  $\frac{1}{2}$ 

Back to top €

#### **ISE Production Code**

What is the ISE Class?
How can I extend the ISE Class?
Do I need to make my own constructor?
Why are there only two files associated with an ISE object?
What methods does my inherited class have available to it?
What methods will my inherited class need to implement?
What changes to the ISE class should I make to implement my selective encryption class?
How do I design a selective encryption algorithm for my media file type?
What encryption should I use in my algorithm?
Is there an example of a class inherited from the ISE Class?
Where can I find more information about Team ISE and the ISE Production Code?

#### What is the ISE Class?

The <u>ISE Class</u> is a C++ super-class, specifically designed to serve as a base for any selective encryption/decryption class targeted for compressed media. It contains all of the necessary methods and data members needed to construct an object for selective encryption, except for the actual encrypt and decrypt methods. Because the algorithms for selective encryption and decryption are uniquely tailored to a specific file type, these methods must be implemented in an inheriting class, such as the <u>JPEG\_ISE Class</u>. When the JPEG\_ISE class was implemented the code was written only to deal with standard baseline JPEG images.

<u>back</u>€

#### How can I extend the ISE Class?

There are a number of steps required to inherit the ISE super class. First, one must design an algorithm to selectively encrypt a specified file type. These algorithms are then implemented within the inheriting class by defining the two virtual methods: <u>encrypt\_file()</u> and <u>decrypt\_file()</u>. All other protected methods within the super class are accessible by the inheriting class as needed. <u>back  $\pm$ </u>

#### Do I need to make my own constructor?

No, the ISE super class implements an overloaded constructor that should be used to construct any object inherited from ISE. **Note:** that the default constructor is declared protected and forces the class user to call the overloaded constructor. Also note that there are many ways to use the overloaded constructor. The only necessary parameter is the encryption/decryption key, but the user may also specify an input and output file. **back**  $\frac{1}{2}$ 

#### Why are there only two files associated with an ISE object?

An ISE object can be used for either encryption or decryption. In each case, the only necessary data members are the input and an output file names. Setting these data members is dependent on the direction (encrypt or decrypt) of the object. back  $\frac{1}{2}$ 

#### What methods does my inherited class have available to it?

There are a number of methods implemented in the super class. An inheriting class may use all of these methods. The methods available are:

ise constructor set\_key set\_input\_file\_name set\_output\_file\_name get\_output\_file\_name make\_ise\_file\_name make\_output\_file\_name get\_ise\_file\_type

These functions allow for the manipulation of all the ISE object data members. A full description of each of these methods can be found at the associated link to the <u>production</u> <u>code reference</u>.

<u>back</u>∌∕

#### What methods will my inherited class need to implement?

There are only two methods an inheriting class needs to implement: <u>encrypt file()</u> and <u>decrypt file()</u>. Both of these methods do not take any parameters and should use the data members of the object to selectively encrypt or decrypt a file. The algorithm for the encryption must be discovered through researching a given file type. <u>back  $\frac{1}{2}$ </u>

# What changes to the ISE class should I make to implement my selective encryption class?

Aside from implementing the <u>encrypt\_file()</u> and <u>decrypt\_file()</u> functionality, there are minor changes that must be made to the super class. Every ISE file type must have a unique ID to be appended to the front of an ISE encrypted file in order to determine the original file type. JPEG ISE files have a '1' as the ID. Also, the IDs '2' and '3' are reserved for mp3 and zip respectively, though their corresponding classes are yet to be defined. These changes need to be made in the ise <u>get\_ise\_file\_type()</u> method and should be considered when creating the encrypt and decrypt functionality.

# How do I design a selective encryption algorithm for my media file type?

There are a number of steps needed to design a new selective encryption algorithm. First, decide on which compressed media you would like encrypted and study its compression standard. Next, identify what portions of the file make good/bad targets for encryption. These targets should be evaluated for properties conducive to selective encryption. The factors to consider might be the portion of the file dedicated to the target, amount of damage to the file when the target is encrypted and how hard the encryption would be to break. The third step to designing a selective encryption algorithm is choosing the encryption. The last step is to inherit the <u>ISE super class</u> and implement the <u>encrypt\_file()</u> and <u>decrypt\_file()</u> methods.

<u>back∳</u>

#### What encryption should I use in my algorithm?

Team ISE decided to use <u>Rijndael AES encryption</u> in the JPEG\_ISE class because it does not increase the file size and the block cipher allows the randomization of number of bytes encrypted at the end of a section. Also, it came highly recommended by <u>Professor</u> <u>Black</u> of the University of Colorado at Boulder. However, if an encryption other than <u>AES</u> is well suited for your purpose, it can be used and will have no problems inheriting from the <u>ISE Class</u>.

<u>back</u>€

#### Is there an example of a class inherited from the ISE Class?

Yes, Team ISE produced an inheriting class with the ISE production code. The <u>JPEG\_ISE</u> Class is designed to implement selective encryption of standard baseline JPEG images. The class is freely available with the <u>ISE super class</u>. <u>back  $\frac{1}{2}$ </u>

#### Is there an example of how to use the JPEG\_ISE Class?

Team ISE has included a <u>main program</u> using the functionality of the <u>JPEG\_ISE class</u>. It was created to be an example of how the class could be used to selectively encrypt a JPEG image.

<u>back∳</u>

# Where can I find more information about Team ISE and the ISE Production Code?

We have created a companion <u>website</u> that includes every research document we have created. This Website also contains links to many useful resources and even a bulletin board for discussions. back  $\Rightarrow$ 

Back to top £

#### **ISE Website**

How are the passwords obtained? Where is the computer running the web page located? What server is used? What OS is the server running on? What is the path to the directory with the web pages on the server? Where are the images, documents, etc. located? What is the IP address of the website? Was an editor used to code the pages? What version of HTML is used? Are the pages written in valid HTML? What character set are the pages coded in? Do the pages make use of a CSS? Do the pages follow a particular format? What language was used to create the menu bar? How can the menu bar be modified? How is the menu bar included in the HTML pages? Where was the message board obtained? What is the message board coded in? What type of database does the message board use? Can the database be modified? How can the database be copied? How are the message board administrator options accessed? How are new categories added? How are categories deleted or edited? How are new forums added? How are forums edited? How are forums deleted? Can threads be automatically deleted? Can the administrator manage the user accounts? How are E-mail addresses banned? How are E-mail addresses un-banned? How are IP addresses banned? How are IP addresses un-banned? How are ranks set? How are ranks deleted?
# How are the passwords obtained?

Passwords are needed to log onto the computer the <u>website</u> is located on. Passwords are also needed to access the administrative options of the message board, and to log to the MySQL database used by the message board. These passwords were only provided to <u>Dr. Tom Lookabaugh</u>. In order to obtain these passwords, you must contact Dr. Lookabaugh. back  $\Rightarrow$ 

## Where is the computer running the web page located?

The web page is run off of a machine in <u>Dr. Lookabaugh's</u> Pervasive Computing Lab in the <u>Discovery Learning Center</u> at the University of Colorado at Boulder. To access this lab, you must obtain keys from Dr. Lookabaugh and the <u>University of Colorado</u>. <u>back  $\frac{1}{2}$ </u>

## What server is used?

Apache/2.0.48 (Fedora) Server is used to serve this web page. This server comes with the Fedora Core Operating System, or can be downloaded from <u>http://www.apache.org/</u>. The particular version used to serve the web page was included in the Fedora Core release. The Apache server is configured to serve pages from a specific folder. The folder on the sponsor's computer is the "html" folder, which can be reached by following path from the root directory:

#### ..\var\www\html

All pages that are to be served must be located in this folder. back  $\frac{1}{2}$ 

## What OS is the server running on?

The Operating System used by the server is the Red Hat Fedora Core Release. For more information on this operating system, please visit <u>http://fedora.redhat.com/</u>. back  $\frac{1}{2}$ 

## What is the path to the directory with the web pages on the server?

To reach the directory with the web pages, you must first log onto the computer where the server is located. You must log on as the root user. Obtain all <u>passwords</u> and usernames from <u>Dr. Lookabaugh</u>. Once logged on, the path to the directory with the pages is:

#### ..\var\www\html

All of the pages are located in the "html" directory. back  $\frac{1}{2}$ 

#### Where are the images, documents, etc. located?

The images, documents, and code accessible to the pages are located in subdirectories of the html directory. These subdirectories were created to organize all of the data used by the web pages.

All of the images are located in the "images" subdirectory. All images should be placed into this directory. Images can then be displayed on the web pages using the following HTML code:

<img src="images/ImageName">

ImageName refers to the actual name of the image plus its extension.

All of the documents are located in the "documents" subdirectory. All documents should be placed into this directory. Documents can then be called using the following HTML code:

<a href="documents/DocumentName">LinkName</a>

*DocumentName* refers to the actual name of the document plus its extension. *LinkName* refers to the name of the desired link. It is recommended that all documents be in PDF format due to its portability.

All of the code is located in the "code" subdirectory. All downloadable code should be placed into this directory. Links to download the code can be created using the following HTML code:

<a href="code/CodeName">LinkName</a>

*CodeName* refers to the actual name of the code download plus its extension. *LinkName* refers to the name of the desired link.

All of the message board code is located in the "board" subdirectory. The following path will lead to the menu bar code from the "html" directory:

#### images\menu\

The images and JavaScript files used to create the menu bar are located in this directory. back  $\frac{1}{2}$ 

## What is the IP address of the website?

The IP address of the website and the computer it is located on is 128.138.75.184. The website is served off of port 80. back  $\Rightarrow$ 

#### Was an editor used to code the pages?

Quantum Plus 3.1 was used to edit the HTML code of the web pages. It is recommended that this editor be used to modify the HTML code. It is not necessary to use this editor. However, the code was formatted using the spacing of this editor, and will be formatted properly in this editor. back  $\Rightarrow$ 

#### What version of HTML is used?

The web pages are coded in HTML 4.01 Transitional. This allows some backwards compatibility with old HTML features. All pages associated with the website define the version of HTML being used by including the following header:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

Any new pages added to this site should include this header and comply with HTML 4.01 Transitional standards. back  $\Rightarrow$ 

## Are the pages written in valid HTML?

The HTML used to code the pages is valid HTML 4.01 Transitional. The pages were validated using the HTML validator located at <u>http://validator.w3.org</u>. Any new pages should be validated using this validator. Additionally, any modifications made to the existing pages should be revalidated. To use the validator, visit the site located at the URL above and supply it with the address of the page that is to be validated. For example, if the download page is to be revalidated, type

http://128.138.75.184/DocumentIndex.html

into the address box on the validator website and click on the "Validate URI" button. The validator will return a report on any necessary fixes required to make the page valid. back  $\frac{1}{2}$ 

#### What character set are the pages coded in?

The pages are coded in the ISO-8859-1 character encoding. All additional pages should include the following code within the header element to define the encoding:

<META http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"> <META http-equiv="Content-Language" content="en-US">

These elements should be included inside of the  $\langle HEADER \rangle$  element. back  $\frac{1}{2}$ 

#### Do the pages make use of a CSS?

A cascading style sheet was not used to define the style of the web pages. The website was intended to be a very simple design that is only used to distribute code and documentation. Thus, no style sheet was defined, and the pages were constructed in a very simple manner.

<u>back</u>∌

#### Do the pages follow a particular format?

All of the pages have a header and footer. These should be included in any additional pages to ensure conformity to the rest of the website. The header and footer code can simply be cut and pasted into any additional pages. The HTML for the header is:

```
<TABLE bgcolor="#003300" align="center" width="100%">
 \langle tr \rangle
    \langle P \rangle
            <FONT size="+1" color="#FFFFFF">
            <B>University of Colorado Computer Science Department</B><br>
            2003-2004<br>
            </FONT>
            <FONT size="+2" color="#FFFFFF">
            <B>ISE JPEG Selective Encryption Sponsor</B>
            </FONT>
        </P>
 </TABLE>
<H1 align="center">
    <img src="images/ISE_icon_white.jpg" alt="Team ISE">
</H1>
```

This code should be included immediately after the *<*BODY*>* element.

#### The HTML for the footer is:

```
<P>
    <FONT size="2">
   This project was done by
    <a href="http://www.colorado.edu" target=" blank">University of Colorado</a>
    students under the supervision of the
    <a href="http://www.cs.colorado.edu" target="_blank">Computer Science Department</a>.
    </FONT>
</P>
<P>
    <FONT size="2">
    This Website is located on a sever at the University of Colorado at Boulder. Questions: Contact
    <a href="http://www.cs.colorado.edu/people/tom lookabaugh.html" target=" blank">Tom
        Lookabaugh</a>
   or
    <a href="mailto:TeamISE@hotmail.com">TeamISE@hotmail.com</a>.
    \langle FONT \rangle
</P>
```

```
<TABLE bgcolor="#003300" align="center" width="100%">
<P align="center">
<FONT color="#FFFFF">
<B>Team Image Selective Encryption Sponsored by Tom Lookabaugh</B><br>
<B>Department of Computer Science</B><br>
<B>Department of Colorado at Boulder</B><br>
<B>University of Colorado at Boulder</B><br>
<B>Boulder, CO 80309-0430</B><br>
<B>HTML 4.01 Transitional</B><br>
<B>Copyright &copy 2003-2004</B>
</FONT>
</P>
```

This code should be inserted immediately before the  $\langle BODY \rangle$  element. back  $\frac{1}{2}$ 

#### What language was used to create the menu bar?

The menu bar was created using the <u>Xara Menu Maker</u> tool. This tool generated the JavaScript used by the menu bar. back  $\Rightarrow$ 

#### How can the menu bar be modified?

The JavaScript file is located using the following path from the "html" directory:

#### images\menu\isemenu.js

<u>Xara Menu Maker</u> is not required to modify this file. The file can be opened in an editor and modified to add additional menu or submenu items. There are two .js files in this directory. Only the isemenu.js file needs to be modified in order to add new menu items. The menu.js file should not be modified. back  $\Rightarrow$ 

#### How is the menu bar included in the HTML pages?

The menu bar is included in the HTML pages using the following HTML code:

```
<SCRIPT src="images/menu/menu.js" type="text/JavaScript"></SCRIPT>
<SCRIPT src="images/menu/isemenu.js" type="text/JavaScript"></SCRIPT>
```

This code should be included immediately after the header. back  $\frac{1}{2}$ 

# Where was the message board obtained?

The message board was obtained from <u>http://www.chipmunk-scripts.com/</u>. This is a free software download that Team ISE made use of rather than creating their own message board.

<u>back</u>€

#### What is the message board coded in?

The message board is coded in PHP with embedded MySQL queries. Because Team ISE did not develop this code, it should not be modified. <u>back</u> $\pm$ 

## What type of database does the message board use?

The message board makes use of an underlying <u>MySQL</u> database. The installation script provided by the message board created all of the necessary tables utilized by the message board. This database software comes with the <u>Red Hat Fedora Core release</u>. <u>back</u> $\frac{1}{2}$ 

## Can the database be modified?

The database can be modified by logging in as the root user and running the MySQL database. Refer to the <u>password</u> section to learn how to obtain the proper username and passwords. However, the database should not be modified to ensure the proper working of the message board. Additions to the message board can be made using the administrator options provided by the message board. This code will create the proper relations in the database needed for any changes. All modification to the database should be done indirectly through the <u>administrator options</u>.

## How can the database be copied?

If the website is to be moved to another machine, the database tables must be copied from the MySQL server the database is currently in to the MySQL server on the new machine. If the new machine does not have a MySQL server, a server must be downloaded and installed by visiting the <u>MySQL</u> link. After the server is downloaded and installed, the database from the old server must be imported to the database on the new server. Refer to the <u>MySQL</u> online documentation for instructions on coping databases to another machine.

<u>back</u>€

## How are the message board administrator options accessed?

First, the administrator login name and password must be acquired. Refer to the section on <u>passwords</u> for information on how to obtain these items. Once these items are known, click on the "<u>Login</u>" link in the upper right corner of the message board page directly under the header. Provide the text boxes with the correct administrator username and password. The message board will then redirect you to the forum page. Centered at the bottom of the forum page is a link called "Admin CP". Clicking on this link will open a window in which all of the administrator options can be accessed by clicking on the links. Click on one of the options under the "Admin Options" table to use the option. back  $\frac{1}{2}$ 

#### How are new categories added?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Add category" link. Provide a name for the category and its rank. The category's rank determines the order in which the category is listed in relation to the other categories. The lowest number category is listed first. After entering this information click on the "submit" button. The message board now contains a new category.

## How are categories deleted or edited?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Delete/Edit Category" link. All existing categories will now be listed in a table. The category name will be listed in the left column, a "Delete?" link in the middle column, and an "Edit?" link in the right column of the table. To delete a category, click on the "Delete?" link found in the middle column of the table. The board will ask you if you are sure you want to delete the category. Click on the "Main" link on the "Admin Options" table to cancel the delete. To edit the category, click on the "Edit?" link in the right column of the table add category window. However, the category name and rank are already filled in. Make desired changes to the name and rank and click submit. The edited changes will now be applied.

## How are new forums added?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Add Forum(s)" link. This will redirect you to the add forum window. Once in this window, you can create a forum by filling in a name in the "name of forum" text box. Next, choose which category to associate the forum with by clicking on the pull down menu. The menu will be populated with all of the categories on the message board. Select the category and then set the permission level to read, post, and reply in the forum you are creating. Selecting one of the permissions from the pull down menu for each read, post, and reply can set the respective permission level. In the last text box, type a definition of the forum. After all text boxes are filled and all permissions set, click on the "Create Forum" button to create a new forum with the information you inserted. back  $s^2$ 

# How are forums edited?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Edit Forums(s)" link. A table will be displayed listing all of the different forums for each category. Click on the "Edit" link next to the forum you desire to change. This will open the same window as when clicking on the "<u>Add Forum(s)</u>" link, except all of the information will be filled in. Change any desired information and click on the "submit" button to make the changes take effect.

#### <u>back</u>€

#### How are forums deleted?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Delete Forums(s)" link. A table will be displayed with all of the forums available for the different categories. The name of each forum will be displayed in the left column of the table, and a "Delete this Forum" link can be found in the right column of the table. Click on the "Delete this Forum" link next to the forum you desire to delete. The board will ask you again if you want to delete the forum. Click on the "Delete this Forum" button again to delete the forum. Click on the "Main" link under the "Admin Options" table to cancel the deletion.

back £

#### Can threads be automatically deleted?

Threads without responses can be automatically deleted after a certain number of days. To adjust the amount of days, first log on as the <u>administrator</u>, click on the "Admin CP" link and then click on the "Prune Topics" link. The administrator can then enter the amount of days before a thread is to be deleted. If the thread does not receive a response after the allotted time has passed, it will automatically be deleted. back  $\Rightarrow$ 

#### Can the administrator manage the user accounts?

The administrator can manage user accounts. After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "User Management" link. Type the user name into the text box and click on the "Search for User" button. If you want all of the users to be displayed, leave the text box empty and click on the "Search for User" button. After clicking the button, the username(s) will be displayed along with the user's email address. To edit a user's profile click on the "Edit" link next to the user's name. The administrator can then edit the user's profile by changing the information in the text box fields. Clicking on the "Edit User" button will make the changes permanent. The administrator can also delete a user by clicking on the "Delete" link next to the user's name. The deletion permanent. Clicking on the "Delete This User" button will complete the deletion. To cancel the deletion, click on the "Main" link under the "<u>Admin Options</u>" table.

back∌

#### How are E-mail addresses banned?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Ban E-mail" link. The administrator can then fill in the text box with the E-mail address that is to be banned. Click on the submit button to make the ban take effect. Any users with a banned E-mail address cannot register to use the message board. **Note:** If the text box is filled with a generic E-mail, such as "@hotmail.com" all users will an E-mail address ending with "@hotmail.com" will not be allowed to register on the message board.

<u>back</u> ∌

# How are E-mail addresses un-banned?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Unban E-mail" link. A list of banned E-mail addresses will be displayed. Click on the "Delete" button next to an E-mail address to un-ban that address. Users with that E-mail address will now be allowed to register to use the message board. back  $\frac{1}{2}$ 

# How are IP addresses banned?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Ban IP" link. The administrator can then fill in the text box with the IP address that is to be banned. Click on the submit button to make the ban take effect. Any users trying to register from a banned IP address will not be able to use the message board. <u>back</u> $\frac{1}{2}$ 

# How are IP addresses un-banned?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Unban IP" link. A list of banned IP addresses will be displayed. Click on the "Delete" button next to an IP address to un-ban that address. Users working from that IP address will now be allowed to register to and use the message board. back  $\Rightarrow$ 

## How are ranks set?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Add Rank" link. The number of posts the user has made determines the rank of a user. To create a rank, the administrator creates a rank title by filling in the "Rank" text box with the title name. The administrator then fills in the "Number of Posts Needed to Achieve" text box with a number. Clicking the "submit" button will create this rank. A user will automatically be assigned the rank once he or she has submitted enough posts. back  $\frac{1}{2}$ 

## How are ranks deleted?

After logging in as the <u>administrator</u> and clicking on the "Admin CP" link, click on the "Delete Rank" link. A list of ranks will then be displayed along with the number of posts needed to achieve such a rank. Clicking on the "Delete" link next to the rank will allow the administrator to delete the rank. The message board will then prompt the administrator again about deleting the rank. Clicking on the "Delete Rank" button will delete the rank. To cancel the deletion, click on the "Main" button under the "<u>Admin Options</u>" table.

<u>back</u>∳

# Back to top £

# **Conclusion**

This Developer's Reference is intended for anyone who plans to modify or extend any of the ISE products or to continue with the research begun by Team ISE. The reference is provided as a supplement to the other design related documents provided in the ISE release. The intent of this document is to answer any questions about modifying or extending any of the Team ISE products or continuing with the ISE research. For further clarification, please refer to the Requirements document, the System Architecture document, the Design document and the reference manuals of the products.

Back to top £

# Research Paper

# Team ISE Image Selective Encryption

#### Selective Encryption of JPEG Standard Baseline Compression Images

Tom Lookabaugh, Andrew T. Pouzeshi, Geoffrey L. Griffith, Joe B. Jarchow, Joseph Z. Kadhim, Shinya Daigaku Department of Computer Science, University of Colorado, Campus Box 530, Boulder, CO, USA 80309-0530

#### Abstract

One of the ramifications of compressing a file is that vital data are localized in small, specific areas. Consequently, it is easy to exploit this property of compression to provide a high level of security as selective encryption focuses on encrypting only these vital portions of data to render a file unusable. Selective encryption results in a large savings in computationally intensive operations, while maintaining a reliable level of security. There have been a number of selective encryption methods proposed for the JPEG compressed image format. This paper describes a simple, yet secure method for selectively encrypting JPEG images that are compressed using the Baseline<sup>1</sup> standard. JPEG Selective Encryption has a high value for any application in which sensitive images may be at risk, from low power satellite imaging systems to securely transmitting images across the Internet.

**Keywords:** JPEG, image encryption, encryption, selective encryption, partial encryption, cryptography, cryptanalysis, compression, security.

#### **1. Introduction**

Selective Encryption is defined as applying encryption to a portion of a file's bit-stream with the assumption that the entire file will become useless without the proper decryptor. The attractiveness of selective encryption arises from the idea that a file can be securely encrypted and transmitted without spending the computational effort of encrypting the entire file. Selective encryption techniques range from encrypting a portion of the file, say a straight percentage of the data, to others that encrypt specific vital sections of a file. Selective encryption methods are never as secure as encrypting an entire file, because much of the data is not encrypted. The goal of selective encryption is to reduce the computational time of encryption, while maintaining a sufficient level of file protection.

The increase of multimedia applications and the transmission of data over public networks necessitate efficient methods of securing transmitted data. Because of the large size of multimedia files, Selective Encryption methods have been devised for various different types of multimedia compressions. The increasing of use of JPEG image encoding software and hardware and transmission across large public networks warrants a strong, yet simple, Selective Encryption scheme for JPEG images. The goal of the research behind this paper was the development of a simple yet secure method of Selective Encryption for the JPEG Baseline compression standard. Such a method would be applicable in situations ranging from encrypting transmitted satellite imagery to encrypting images generated by digital cameras to protecting images for transmission across the Internet.

<sup>&</sup>lt;sup>1</sup> As defined by Pennebaker and Mitchell in "JPEG Still Image Data Compression Standard."

#### 2. JPEG Image Selective Encryption Criteria

The goal of Selective Encryption for JPEG images is to minimize the amount of encryption applied to a file while maximizing the damage done to the image. As a bonus, this paper will define a method that is relatively fixed in size and will not require the amount of encryption to increase linearly as the image size increases linearly. Most of the file will remain unencrypted, which allows the retrieval of those data. However, the data that remains unencrypted will be useless without the encrypted data and the encrypted data will be reasonably difficult for a hacker to replace, reconstruct or calculate. The result will be a JPEG file structure, partially encrypted, that is impossible to use without the proper decryptor and key. The focus of this paper is to present the research and algorithm developed for selective encryption of standard Baseline compressed JPEG image files.

#### 3. Goals and Criteria for Selective Encryption

Selective encryption can be measured in several different ways and optimized for many different purposes. Confusion may arise from reading literature about selective encryption, as the method is usually specific to the type of file being encrypted. Thus, this confusion can be avoided by having a clear idea of our selective encryption criteria. The criteria used for selective encryption include:

#### **Security Criterion:**

Selective encryption has been proposed for a number of different user scenarios. For the purposes of this paper, we will define the security criterion as encryption of data sufficient to render the image unusable to a standard JPEG image decoder. The data must be vital enough to render the image unusable to an attacker's reconstruction or replacement of data to the point that the attacker would be forced to use an expensive brute force method to decode the image. Although the attacker will still be able to retrieve most of the data from the selectively encrypted image file, the image itself cannot be easily reconstructed.

#### Security Validation:

Security of any given encryption can be validated in a number of different ways. Some researchers validate security by choosing the criteria and then feeding the selectively encrypted data into a standard decoder and observing resulting reconstructions. Other researchers take a cryptanalytic approach by acting as an attacker and working with a modified decoder and other available information to design a method of defeating the selective encryption. Still others make mathematical calculations, such as RMS (root mean squared) or PSNR (peak signal-to-noise ratio), to find differences between the encrypted and unencrypted data values. This paper considers all three of these methods as valid and all three have been considered for this cryptosystem.

#### **Complexity:**

Often encryption can be complex and computationally expensive. The primary goal of selective encryption is to reduce the percentage of data that needs to be encrypted, while maintaining an acceptable level of security. This reduction of encryption operations must

be weighed against increased operations necessary to implement the selective encryption algorithm. If computing the data to encrypt and/or searching for those data is more expensive than simply encrypting the entire file, then the selective encryption system should not be considered as valid.

#### **Compression Efficiency:**

The primary goal of compression is to store a set of data in less space than the data representation requires by utilizing specialized algorithms. Image, video, and audio compression formats often exploit the fact that a human detects only a portion of the overall sensory input. Therefore, certain compression formats further reduce the number of bits needed to represent the data by approximating the data values such that the difference is relatively undetectable to human sense. Note that in these cases, the exact data are lost, but the compression efficiency will be much greater. However, the quality of the media is degraded exponentially by increasing the overall compression. For this reason, compressors often allow for varying degrees of data loss.

Some methods of selective encryption compromise compression efficiency by adding data overhead and/or by modifying the compression algorithm, causing a penalty in performance. For example, constantly searching a data stream for information about where to encrypt will add computational overhead. In addition, certain encryption algorithms greatly increase the size of the data, which is contrary to compression. Although there are newer encryptions that do not increase data size, circumstances may require use of less size efficient encryption algorithms. Any degradation in compression efficiency must be weighed against the constraints surrounding the particular need for selective encryption.

#### **Interaction with Compressors:**

There are methods of selective encryption that work with, and others that are independent, of the compression algorithm. It is important to be aware that there are potentially major differences in both performance and compression efficiency between these two methods. Ideally, the selective encryption algorithm would be implemented within the compression algorithm. This will minimize file parsing operations and reduce the overall number of operations needed to find the portion of data to encrypt.

#### **Selective Encryption Attacks:**

One final item to define and consider are the types of attacks on a selective encryption system. There is a clear difference between cracking a particular selective encryption system and cracking an encryption algorithm. If the encryption algorithm used to implement the selective encryption system is breakable, or found to be breakable in the future, we must assume the selective encryption system is invalidated and we must switch to a more secure encryption algorithm. For the purpose of this paper, we will assume the particular encryption algorithm used to implement JPEG selective encryption system is secure and we will discuss only attacks that pertain to selective encryption and not the various attacks on different encryption algorithms.

There are really three ways of attacking a selective encryption system. The first, and most well known, is a purely brute force attack. Since selective encryption systems only encrypt a portion of the file, usually the minimum possible to sufficiently protect the data, it will take much less time to either defeat the encrypted data or remove the encrypted data and try a systematic bit replacement of all possible values. For JPEG selective encryption, an attacker would work with a standard JPEG image decoder and would run each permutation of the replaced data through the decoder to try to rule out the most obvious, unviewable images. Then, once the automated process has produced a number of images that are viewable, the attacker would have to look at each one to find the correct, or at least understandable image. However, this will potentially take a large amount of time, depending on how many bits are encrypted. So, we will define the brute force attack as the most undesirable method.

The second method of attack is defined as a reconstruction attack. An expert in the particular file format that is selectively encrypted could devise a method for reconstructing the vital data that have been encrypted, given the unencrypted data in the file. In the case of JPEG selective encryption, the attacker would work with a modified JPEG image decoder that would compute the correct information, given the selectively encrypted file. Fortunately, the JPEG compression standards (along with many other compression standards) are designed specifically to decompose the original image into its vital components that allow the decoder to calculate each pixel value to reduce overall file size. Ironically, it is this reason that makes selective encryption of compression formats very attractive. Still, and in general, it is important to be extremely familiar with the data format of the selectively encrypted file, and measures must be taken to avoid this form of attack.

Finally, the third, and probably most effective, would be a hybrid attack. There are several possibilities for this type of attack, which would consist of doing research on real world instances of the particular file format to try to find consistency of vital data and having some understanding of how the data are structured. This could help the attacker by reducing the amount of "most likely" possibilities of that data. For JPEG selective encryption, this would probably entail a basic understanding of the JPEG image data components and some prior knowledge of a large number of real world instances of JPEG images and commonly used JPEG encoding schemes. Then, it could be determined if the encrypted data could be replaced by trial and error with a relatively small number of sets of real world data to try to reproduce (or even approximate) the original image to an acceptable level. Again, measures must be taken to ensure this type of attack will lead to failure.

#### 4. Previous Selective Encryption Attempts

There is currently a small amount of existing research on the topic of Selective Encryption available for various multimedia formats. Much of the research pertinent to this paper is based on previous MPEG and JPEG Selective Encryption techniques and research. Through out this literature there is much indecision as to which file components are the best target(s) for Selective Encryption. This paper attempts to evaluate all possible targets and previous attempts of Selective Encryption for JPEG image formats and any possible attacker's counter measures.

In their paper "Selective Encryption of the JPEG2000 Bitstream," Norcen and Uhl [8] outline a selective encryption method for the JPEG2000 compressed file format. The proposed method uses an AES block cipher to encrypt 20% of the visual information in JPEG2000 files, providing relatively secure file transmission without the computational costs of encrypting the entire file. While this method is more efficient than encrypting the entire file, the algorithm fails to exploit the relationship between compression and isolation of vital data. Moreover, the amount of data that is encrypted in the file increases linearly as the JPEG file size increases. Ideally, the amount of data encrypted would be relatively fixed and would include only vital components that would render the image unusable.

By carefully selecting vital components of the file to encrypt, it is possible to provide security while encrypting an even smaller, and ideally fixed, portion of the file. Several other research papers (mostly concerning MPEG Selective Encryption) suggest targeting the DCT (Discrete Cosine Transform) Quantizer tables found in many compressed multimedia file formats, including JPEG formats. The DCT is a mathematical technique used for decomposing wavelengths into elementary frequency components. For a JPEG image, these coefficients are stored in the Quantizer table. Encrypting the Quantizer tables are an attractive target because there is no variance in table size and the number of tables allowed is small, yet the minimum amount of Quantizer data is not so small that it could easily be permuted or guessed. Each Quantizer table must be exactly 64 bytes, and there are no less than 1 and no more than 4 allowed. Thus, there is a minimum of  $2^{512}$ possibilities and up to  $2^{2048}$  possibilities to guess the exact Quantizer table(s) encrypted. This target is also large enough that a non-intelligent brute force attack of simply substituting values for these tables would take a considerable amount of time to reproduce the original image. Even though the Quantizer table looks promising at first glance, it proves to be an extremely weak target for JPEG selective encryption, as we'll see in section 5 of this paper.

In their paper "Secure Compression using Adaptive Huffman Encoding," Kailasanathan, Naini, and Ogunbona [4] propose the Huffman encoding tables, found in the Baseline JPEG format, as a viable target for selective encryption. This selective encryption algorithm offers two possible solutions. The first involves removing the compression tables from the image, securely transmitting the tables separately, and then reintegrating the tables when received. The second, more appealing solution, is to encrypt the compression table and send it along with the file and then securely transmitting a key to decrypt on the other side. As with the Quantizer tables, the Huffman tables appear to be a good target for selective encryption, because these tables have a relatively small variance in size, yet the minimum size is sufficient to repel brute force attacks. After further research discussed in section 5 of this paper, the Huffman tables prove to be a more valuable a target for selective encryption. Unlike the Quantizer table values, it is not as easy to produce an image by replacing the Huffman values of an optimized JPEG image. However, because many JPEG compression applications use default Huffman tables, an attacker may have success by trying a series of popular default tables used by the more common graphical editing applications, digital cameras (JPEG encoding chips) or the example tables in the JPEG standard. Still, both Quantizer and Huffman seemed to have potential, and in the end, the research finally yielded a solid solution.

#### 5. Cryptanalytic Approach to JPEG Selective Encryption

To devise an algorithm for selectively encrypting JPEG images effectively, the team researched the feasibility of this project from several different angles. Since there is no universal method for selective encryption, the team thought it appropriate to examine previous research on subject for multiple compression formats, review the JPEG baseline compression standard<sup>2</sup>, research of common implementations of the JPEG encoders, and collect a large sample of real world JPEG images to be used for statistical analysis. By the end, the team was able to devise a method of selective encryption that will sufficiently protect JPEG images against any of the possible attacks mentioned in this paper.

The team began by researching the Baseline standard compression for JPEG images. Although there is a large amount of data included in the format, much of it is not vital to the image, or can be replaced, or even calculated. The team narrowed the possible targets for selective encryption to three pieces: the Encoded Data stream, the Quantizer tables, and the Huffman tables (which coincided with previous research available).

As mentioned above, a previous attempt at the Selective Encryption of JPEG images was to encrypt a percentage of the entire Encoded Data. While this method will definitely work, it was ruled out for two reasons. The first, and the most important reason, is that non-intelligently encrypting a percentage of the Encoded Data fails to exploit the relationship between a compression format and the concentration of vital data. Second, the amount of encryption needed will linearly increase as the size of the file increases. The Encoded Data makes up the largest percentage of the file size (on the order of 96% for JPEG images under 20 KB and 99%+ for files of 200 KB or more). The goal is to have a relatively fixed amount of data that needs to be encrypted and ideally that size will not be dependent upon the image size. Thus, the Encoded Data was ruled out as a viable target.

Another possible target found from both analyzing the JPEG standard and reviewing previous research is the Quantizer tables. There was a considerable amount of selective encryption research available for methods that utilize the Quantizer tables, but much of it was for other compression formats. However, there were at least two research papers on the topic of selective encryption for JPEG images that suggested the Quantizer tables are good targets. With this in mind, the team decided to try working with this Quantizer to see what effect, if any, altering these values had on various images. During the course of the research, over 2500 random JPEG images were gathered from the Internet and over 200 were tested directly. Unfortunately, it was determined that this target was neither vital enough nor unique enough to provide ample security. Altering the DCT coefficients only distorts the resolution, brightness, or color. Even a completely random table would

<sup>&</sup>lt;sup>2</sup> As defined by Pennebaker and Mitchell in "JPEG Still Image Data Compression Standard."

yield a viewable image of the original only slightly degraded. In many cases, the team was able to reconstruct most images by simply replacing the entire table with a single value for each of the DCT coefficients, allowing the image to decode with a negligible degradation of quality. Although the images were often slightly discolored and/or the resolution was distorted, these images were certainly not damaged enough to render them incomprehensible. For this reason, the Quantizer tables were ruled out as a viable target.

Finally, the team focused on the Huffman (compression) tables as a target for selective encryption. The image was found to be extremely sensitive to minor changes in the Huffman tables, as these tables are used to generate/decode the Encoded Data stream. If even one encoding value is altered, then the resulting image will be considerably damaged. Furthermore, it will be impossible to reconstruct images by replacing Huffman tables with random values or even different Huffman tables from other images. Unlike the encoded data stream, the size of these tables is relatively fixed, as the Baseline standard dictates that there can be a maximum of four of these tables. So, on the surface, and as other research pointed out, the Huffman tables seem to be the most attractive target for JPEG selective encryption. However, it is necessary to look more into JPEG compressors and common instances of JPEG images to validate the security of a selective encryption method that targets the Huffman tables.

There are a wide variety of different JPEG encoders available, such as the IJG<sup>3</sup> JPEG encoding/decoding classes, Adobe Photoshop (a professional image editing application) or even the common Microsoft Paint (included with every copy of Microsoft Windows). While each encoder provides a different level of features, they all work with the JPEG Baseline compression standard. The main differences among these encoders can be measured by how they actually encode the image itself. While some encoders will actually calculate an optimized Huffman table, others use a series of default tables that are pre-calculated. Although these pre-calculated tables reduce computation, they pose a problem to security, because if an attacker had "inside information" on which JPEG encoder was used, they might be able replace the encrypted compression table. Due to the existence of default compression tables, a selective encryption method that only encrypted the Huffman tables would be insecure.

A remedy to solve the problem with default Huffman tables would be to optimize the compression of every JPEG image, before selectively encrypting. However, there are two potential problems with this remedy. First, using the IJG compressor with a flag to optimize images, the team produced approximately 470 optimized JPEG images. These images were randomly collected from the Internet. Even after optimization, there were still a large number of duplicate Huffman tables. Of these non-optimized images, 76.3% contained duplicate Huffman data. After optimizing these same images, 39.6% contained duplicate tables still existed, meaning that even if images are optimized, attackers may still be able to replace these values. Secondly, a goal of selective encryption is to reduce the amount of computation necessary to protect the file. However, by optimizing JPEG

<sup>&</sup>lt;sup>3</sup> The IJG Organization is one of the most common providers of a C++ API for encoding and decoding JPEG images.

images (i.e. not making use of pre-calculated tables), there is an increase in the amount of computation needed to assure security of the image. Moreover, many of the JPEG compression chips used in digital cameras or satellite systems do not have the capability of calculating an optimized table. So, although the Huffman tables seem like the perfect target, they alone do not provide the level of security selective encryption hopes to achieve.

After spending a considerable amount of time researching, it became increasingly apparent that just encrypting one or two frames of data in the image wasn't going to solve all of the problems. The attacker could know at least the size of the table and the number of tables for both the Quantizer and the Huffman tables by counting encrypted frames in the image. Moreover, the Huffman tables have an ordering which greatly reduces the number of possible permutations and the Quantizer tables by themselves are much too weak because even a randomized table will often produce a degraded image, but not damaged enough to make it completely unusable. The team realized that we needed to hide the exact size and number of the compression tables.

To overcome all of these drawbacks, Team ISE devised an algorithm that encrypts not only the compression data frames, but also all the data between the compression tables and the beginning of the Encoded Data stream. The Team ISE algorithm can be implemented in cooperation with compression or independent of compression, as well as in software or in hardware. The algorithm is as follows:

- 1. Choose a block size of some number of bytes (for example, 32 bytes work well with the AES block cipher encryption system).
- 2. Write the file as normal until the FFC0 (SOF0 frame) or FFC4 (DHT frame) marker (whichever is written first for the particular encoder).
- 3. Write this 2 byte marker and then start encrypting in blocks of the pre-chosen block size until the FFDA marker (SOS frame) is to be written.
- 4. Encrypt the FFDA marker and fill out the rest of the current block and write it to file.
- 5. Encrypt one final block and write it to file.
- 6. Write the rest of the Encoded Data stream and file as normal.

This effectively hides the size of the Huffman tables within the file. This causes the encryption to run directly into the Encoded Data stream. Since both the encryption and the encoded data stream appear to be random values, it is now impossible to tell where the Huffman tables end and the Encoded Data begins. Thus we have overcome the problem of direct table replacement. Furthermore, a brute force attack would be extremely expensive, because the average size of these tables for a small image would yield about 2<sup>2400</sup> possibilities! This leaves only the problem of the Hybrid attack with (1) "inside information" of a compressor that (2) uses pre-calculated or default compression tables that are unchanging. In this case, an attacker could replace the encrypted table and recalculation of the Scan header frame. Any data that was encrypted at the beginning of the Encoded Data stream could be systematically substituted until the correct solution is found. At a minimum, the Hybrid attack method would have (assuming a 32 byte block

size) at least  $2^{256}$  possibilities and at most, there would be at most  $2^{512}$  possibilities. Thus, this particular Hybrid attack would still be very expensive and take quite a bit of time and effort by the attacker. However, the key to overcoming this attack is to use an optimized compression algorithm for the table. Moreover, this cryptosystem encrypts only about 3% of the JPEG image data for a very small image around 20 KB and for the case of a image produced by a digital camera (of about 1 MB in size), this selective encryption algorithm will encrypt only about 0.001% of the file.

#### 6. Conclusion

After researching previous attempts at JPEG selective encryption, we found that although previous researchers were definitely on the right track, there are many weaknesses in the other approaches. The algorithm developed by Team ISE overcomes these weaknesses while adhering to the original goals of selective encryption defined in this paper. The algorithm performs in such way that the number of computational operations needed to encrypt the data does not increase as file size increases. Furthermore, the algorithm is simple enough that it can be easily implemented in both software and hardware, in cooperation or independent of the compressor, thereby lending itself to provide high flexibility for many different applications. The Team ISE selective encryption algorithm will only be vulnerable to a brute force attack. The algorithm defined here has met all of the goals set out in this paper and finally, but most importantly, the algorithm is secure.

#### **Bibliography**

- 1. Chang, H. and Li, X. On the Application of Image Decomposition to Image Compression and Encryption. 1996.
- 2. Chang, H. and Li, X. Partial Encryption of Compressed Images and Videos. 2000.
- 3. Droogenbroek, M. and Benedett, R. *Techniques for Selective Encryption of Uncompressed and Compressed Images*. 2002.
- Kailasanathan, C. and Naini, R. Compression Performance of JPEG Encryption Scheme. 2003.
- 5. Li, X., Knipe, J. and Cheng, H. *Image Compression and Encryption Using Tree* Structures. 1997.
- 6. Lookabaugh, T., Sicker, D., Keaton, D., Guoand, W. and Vedula, I. Security Analysis of Selectively Encrypted MPEG-e Streams. 2003.
- 7. Miano, J. *Compressed Image File Formats*. Addison Wesley Longman, Inc., Reading, Massachusetts, 1999.
- 8. Norcen, R. and Uhl, A. Selective Encryption of the JPEG2000 Bitstream. 2003.
- 9. Pennebaker, W. and Mitchell J. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, New York, 1993.
- 10. Podesser, M., Schmidt, H. and Uhl, A. Selective Bitplane Encryption for Secure Transmission of Image Data in Mobile Environments. 2002.
- 11. Seo, Y., Kim, D., Yoo, J., Dey, S., Agrawal, A. Wavelet Domain Image Encryption by Subband Selection and Data Bit Selection. 2003.

# Source Code

# Team ISE Image Selective Encryption

# **ISE Production Code Files**

Apr 21, 04 17:48	ise.h	Page 1/2	Apr 21, 04 17:48	ise.h	Page 2/2
			#ifndef JPEG ISE H		
ise.h			#define JPEG_ISE_H	public ise	
Authors: Joe Jarchow Andrew Pour	1, Geoffrey Griffith, Joseph Kadhim, Shiny zeshi	ya Daigaku	{ public:	····	
Sponsor: Tom Lookab	augh. Assistant Professor of Computer Sol	ence	public.	<pre>jpeg_ise(char*, char* = NULL, char* = NULL ~ipeg_ise():</pre>	);
University	of Colorado			<pre>int encrypt_file(); int decrypt_file();</pre>	
Senior Project: Tear	a ISE (Image Selective Encryption)		protected:	inc decrypt_life();	
For more information	a do to. http://128 138 75 184		}; #endif //TPEC ISE H	)peg_ise(),	
			#endii //orbo_ios_n		
This code is open so The authors are in a	ource and may be used with no cost.				
from the usage of the	is code. It is provided as is with				
support. The author for good purposes	s would hope it would only be used				
The nurness of this	file is to define what functions and memb	bars			
are to be exported :	for a programmer using the ISE class. ISE	S			
jpeg images. class	is intended to be the super class and	d the			
jpeg_ise class at the	is time but other classes could be implemented	nented			
are various function	is for setting and getting the class member	ers			
in the ise.cpp file.					
nclude «stdlib b>					
nclude <iostream></iostream>					
ing std::ifstream;					
fndaf ISE U					
efine ISE_H					
{ public:					
public.	<pre>ise(char*, char* = NULL, char* = NULL); virtual ~ise();</pre>				
	<pre>virtual int encrypt_file() { return 0; } virtual int decrypt_file() { return 0; }</pre>				
	<pre>int set_key(char*); int set_input file name(char*);</pre>				
	<pre>int set_output_file_name(char*); char* get_input_file_name();</pre>				
protect	<pre>char* get_output_file_name(); dd.</pre>				
proceed	<pre>ise(); int get ise file type(char*):</pre>				
	<pre>int make_ise_file_name(); int make_output file_name();</pre>				
nrivato.	<pre>char* get_key();</pre>				
privace.	<pre>char* input_file_name; char* output_file_name;</pre>				
1.	char* key;				
ndif //ISE_H					
nday May 02, 2004		Tea	im ISE		

ise.cpp Apr 21, 04 17:48 ise.cpp Page 1/23 Apr 21, 04 17:48 Page 2/23 ise::ise() 11 ise.cpp 11 Authors: Joe Jarchow, Geoffrey Griffith, Joseph Kadhim, Shinya Daigaku 11 Andrew Pouzeshi 11 11 // Overloaded Constructor 11 Sponsor: Tom Lookabaugh, Assistant Professor of Computer Science // Pre-conditions: The key must be a pointer to a character string. University of Colorado // Post-conditions: An ISE object is created containing the specified 11 data members. 11 Senior Project: Team ISE (Image Selective Encryption) // Parameters: The first argument is a pointer to the key. December 2003 The second argument is the name and path of the input file 11 11 to be encrypted or decrypted. The third argument is 11 11 For more information go to: http://128.138.75.184 11 11 the file name and path for the output file generated by 11 encryption or decryption. 1 // Return values: Constructor, no return type. // Description: 11 This code is open source and may be used with no cost. An ISE object is constructed with the data necessary to encrypt or decrypt a file. This overloaded The authors are in no way responsible for any effects 11 11 from the usage of this code. It is provided as is with 11 constructor only requires that the first argument 11 // no warranties, protections, promises or any form of 11 be provided. The second and third arguments are optional // support. The authors would hope it would only be used 11 and will be set to a default value of NULL. 11 for good purposes. Thank you. 11 //--11 ise::ise(char\* key, char\* input\_file\_name, char\* output\_file\_name) // The purpose of this file is to define what functions and members size\_t length; // are to be exported for a programmer using the ISE class. ISE char \* key\_copy; // is a class defined to implement image selective encryption for char \* temp; jpeg images. class ise is intended to be the super class and // must be inherited by sub classes. We have only implemented the // check that the key in not NULL jpeg\_ise class at this time but other classes could be implemented if (key == NULL) // following the outline used. Along with constructors there are // are various functions for setting and getting the class members exit(1); 11 each is defined in detail preceeding the appropriate function in the ise.cpp file. // check that the input and output files are of type jpeg or ise char \* index; if (input file name != NULL) #include <stdlib.h> **#include** <string> index = strstr(input\_file\_name, ".jp"); **#include** <iostream> if (index == NULL) **#include** <stack> { **#include** <cstdlib> index = strstr(input\_file\_name, ".JP"); **#include** "rijndael-api-fst.h" // use for block cipher encryption/decryption if (index == NULL) using namespace std; index = strstr(input\_file\_name, ".ise"); using std::cerr; if (index == NULL) using std::endl; { using std::nothrow; index = strstr(input\_file\_name, ".ISE"); if (index == NULL) const int JPEG\_TYPE = 1; // specify jpeg ise const char JPEG\_FILE\_TYPE = '1'; // specify jpeg file type const unsigned int MIN\_KEY\_LENGTH = 32; // minimum length of the key exit(1); **const** int BUFFER\_LENGTH = 16; // size of Rijndael encryption block } typedef unsigned char byte; } #include "ise.h" if (output\_file\_name != NULL) 11 index = strstr(output\_file\_name, ".jp"); // Default Constructor if (index == NULL) // Pre-conditions: None. { // Post-conditions: None. index = strstr(output\_file\_name, ".JP"); // Parameters: None. if (index == NULL) // Return values: Constructor, no return type. // Description: Default constructor is not used by users. index = strstr(output\_file\_name, ".ise"); if (index == NULL)



```
Apr 21, 04 17:48
                                       ise.cpp
                                                                        Page 4/23
               case -6:
                        temp[i] = 'a';
                       break;
               case -5:
                        temp[i] = 'b';
                        break;
               case -4:
                       temp[i] = 'c';
                       break;
               case -3:
                        temp[i] = 'd';
                       break;
               case -2:
                       temp[i] = 'e';
                       break;
               case -1:
                        temp[i] = 'f';
                       break;
               }
       // extend the key length to 32 bytes
       if (length < MIN_KEY_LENGTH)</pre>
               this->key = new (nothrow) char[MIN_KEY_LENGTH + 1];
               if (this->key == NULL)
               {
                       exit(1);
               strcpy(this->key, temp);
               for (size_t i = length; i < MIN_KEY_LENGTH; i++)</pre>
                        this->key[i] = '0';
               this->kev[MIN KEY LENGTH] = '\0';
       else
               this->key = new (nothrow) char[length + 1];
               if (this->key == NULL)
               {
                       exit(1);
               strcpy(this->key, temp);
      delete [] key_copy;
delete [] temp;
       // set the input file name
       if (input_file_name != NULL)
       {
               length = strlen(input_file_name);
               this->input_file_name = new (nothrow) char[length + 1];
               if (this->input_file_name == NULL)
               {
                        exit(1);
               strcpy(this->input_file_name, input_file_name);
       else
               this->input_file_name = NULL;
       // set the output file name
       if (output_file_name != NULL)
       {
               length = strlen(output_file_name);
               this->output_file_name = new (nothrow) char[length + 1];
```

Apr	21, 04 17:48	ise.cpp	Page 5/23	Apr 21, 04 17:48	ise.cpp	Page 6/23
	if	(this->output_file_name == NULL)		strcpy(name_copy, n	ame);	
	1	exit(1);		// split each chara	cter into four bit values.	
	, st:	<pre>rcpy(this-&gt;output_file_name, output_file_name</pre>	);	{ { {		
	else			name_copy[i] = 1	name_copy[i] << 4;	
	th:	<pre>is-&gt;output_file_name = NULL;</pre>		temp[1 * 2 + 1] }	= name_copy[1] >> 4;	
}	}			length = length * 2	;	
ise:	:~ise()			temp[length] = $' \setminus 0';$		
{	<pre>if (key !=</pre>	NULL)		<pre>// convert four bit for (size_t i = 0;</pre>	<pre>values to hexadecimal characters i &lt; length; i++)</pre>	
	{ 	lete [] key;		{     switch((int)tem)	p[i])	
	} if (input_:	file_name != <b>NULL</b> )		{ <b>case</b> 0:		
	{ de:	<b>lete</b> [] input_file_name;		<pre>temp[i] = ' break;</pre>	0';	
	} if (output	_file_name != NULL)		<pre>case 1: temp[i] = '</pre>	1';	
	{ de:	lete [] output file name;		break; case 2:		
}	}			<pre>temp[i] = 'i break;</pre>	2';	
//				<pre>case 3: temp[i] = '</pre>	3':	
// // P1	re-conditions:	The key must point to a character string.		break; case 4:		
// Pc	ost-conditions	The key will be set using the new strin Any previous information in key will be los	g specified.	temp[i] = '	4';	
// Pá	arameters:	The only argument to this method is a point	er to	case 5:	51.	
// // P/	oturn waluos.	for either encryption or decryption.	veges or failure	break;	,	
// 1/6	ecuin varues.	A zero will indicate a success.	cess of failure.	temp[i] = '	6';	
//	~	A two will indicate an invalid key. A two will indicate a m	memory allocation	case 7:	77.	
// De	escription:	The method will use the specified character	string to	break;	/ <i>;</i>	
11		decryption methods.		temp[i] = '	8';	
//		· · · · · ·		case -7:		
1 1 1	lse::set_key(c	nar* name)		<pre>temp[1] = '! break;</pre>	9';	
	char * name	gtn; e_copy;		<b>case</b> -6: temp[i] = ':	a';	
	<i>char</i> * tem	p;		break; case -5:		
	// check t. if (name ==	hat the key is not NULL = <b>NULL</b> )		temp[i] = '	b';	
	{ ret	turn 1;		<b>case</b> -4: temp[i] = '	c';	
	}			break; case -3:		
	length = st name_copy =	trlen(name); = <b>new</b> (nothrow) <i>char</i> [length + 1];		<pre>temp[i] = 'o break;</pre>	d';	
	if (name_co {	opy == NULL)		<b>case</b> -2: temp[i] = '	e';	
	ret }	turn 2;		<pre>break; case -1:</pre>		
	temp = <b>new</b> if (temp ==	<pre>(nothrow) char[length * 2 + 1]; = NULL)</pre>		<pre>temp[i] = 'i break;</pre>	f′;	
	{ ret	turn 2;		}		
	}					

```
Apr 21, 04 17:48
                                       ise.cpp
                                                                       Page 7/23
                                                                                       Apr 21, 04 17:48
                                                                                                                              ise.cpp
                                                                                                                                                              Page 8/23
    // delete the previous key information
                                                                                                       index = strstr(name, ".JP");
        delete [] key;
                                                                                                      if (index == NULL)
        // extend the key length to 32 bytes
        if (length < MIN_KEY_LENGTH)</pre>
                                                                                                               index = strstr(name, ".ise");
                                                                                                               if (index == NULL)
                key = new (nothrow) char[MIN_KEY_LENGTH + 1];
                if (key == NULL)
                                                                                                                       index = strstr(name, ".ISE");
                                                                                                                       if (index == NULL)
                        return 2;
                                                                                                                               return 1;
                strcpy(key, temp);
                for (size_t i = length; i < MIN_KEY_LENGTH; i++)</pre>
                        key[i] = '0';
                key[MIN_KEY_LENGTH] = ' \ 0';
                                                                                               // delete any previous input file information
                                                                                              if (input_file_name != NULL)
        else
                                                                                                       delete [] input_file_name;
                key = new (nothrow) char[length + 1];
                if (key == NULL)
                                                                                              // set the input file name
                                                                                              length = strlen(name);
                        return 2;
                                                                                              input_file_name = new (nothrow) char[length + 1];
                strcpy(key, temp);
                                                                                              if (input_file_name == NULL)
                                                                                                      return 2;
       delete [] name_copy;
delete [] temp;
                                                                                              strcpy(input_file_name, name);
        return 0;
                                                                                              return 0;
// Pre-conditions:
                        The name must be a pointer to a valid jpeg or ise file
                                                                                      // Pre-conditions:
                                                                                                               The name must be a pointer to a valid jpeg or ise file
                    type.
                                                                                                           type.
// Post-conditions:
                        The input_file_name will be set using the new string
                                                                                      // Post-conditions:
                                                                                                               The output_file_name will be set using the new string
                    specified. Any previous data in input_file_name will
                                                                                                           specified. Any previous data in output_file_name will
1
11
                    be lost.
                                                                                      11
                                                                                                          be lost.
// Parameters:
                    The only argument to this method is a pointer to a
                                                                                      // Parameters:
                                                                                                          The only argument to this method is a pointer to a
11
                    character string containing the input_file_name,
                                                                                                           character string containing the output_file_name,
                                                                                      //
                    specifying the input file to encryption or decryption.
                                                                                                           specifying the output file to encryption or decryption.
// Return values: An integer is returned indicating a success or failure.
                                                                                      // Return values:
                                                                                                          An integer is returned indicating a success or failure.
11
                    A zero will indicate a success.
                                                                                      11
                                                                                                          A zero will indicate a success.
11
                    A one will indicate an invalid input file name.
                                                                                      11
                                                                                                          A one will indicate an invalid output file name.
                                        A two will indicate a memory allocation
                                                                                                                               A two will indicate a memory allocation
11
                                                                                      11
error.
                                                                                      error.
// Description:
                    This method is used to set the input_file_name.
                                                                                      // Description:
                                                                                                        This method is used to set the output_file_name.
                    The method must be called prior to the encryption
                                                                                      //--
11
                    or decryption methods if they were not specified
11
                    in the constructor.
                                                                                      int ise::set_output_file_name(char* name)
11
11-
                                                                                              size_t length;
int ise::set_input_file_name(char* name)
                                                                                              // check that the name is not NULL
        size_t length;
                                                                                              if (name == NULL)
        // check that the name is not NULL
                                                                                                      return 1;
        if (name == NULL)
                return 1;
                                                                                              // check that the name is a jpeg or ise file type
                                                                                              char * index;
                                                                                              index = strstr(name, ".jp");
        // check that the name is a jpeg or ise file type
                                                                                              if (index == NULL)
        char * index;
        index = strstr(name, ".jp");
                                                                                                       index = strstr(name, ".JP");
        if (index == NULL)
                                                                                                      if (index == NULL)
```

Apr 21, 04 17:48	ise.cpp	Page 9/23	Apr 21, 04 17:48	ise.cpp	Page 10/23
{	nden strate(neme light).		, ret	turn NULL;	
i	f (index == NULL)		return outr	<pre>put_file_name;</pre>	
1	<pre>index = strstr(name, ".ISE");</pre>		}		
	<pre>if (index == NULL) {</pre>		//		
	return 1;		// Pre-conditions:	The name must be a pointer to a va	lid ISE file.
}	}		// Parameters:	The only argument for this method is a	pointer
}				to a character string indicating the n valid ISE file.	ame of a
<pre>// delete any pre if (output_file_n</pre>	evious output file information name != <b>NULL</b> )		<pre>// Return values: // //</pre>	The function will return an integer in the type of the original file from whi ISE file was created.	dicating ch the specified
{ delete []	output_file_name;			0 will indicate an unknown or unimplem 1 will indicate a jpeg file.	ented file type.
}				2 will indicate a mp3 file. 3 will indicate a zip file	
// set the output	file name		11	The return values may be extended to a	ccommodate other file
length = strlen(n output_file_name	<pre>name); = new (nothrow) char[length + 1];</pre>		types. // Description:	This method will return an integer cor	responding to
if (output_file_n	name == <b>NULL</b> )			the original file type of an encrypted	ISE file.
return 2;			//		
, strcpy(output_fil	e_name, name);		{ {	IIe_type(chai* name)	
return 0;			char the_ty	ype;	
}			ifstream is	<pre>se_infs(name, ios::binary);</pre>	
//			// check th if (ise_int	hat the file can be opened fs.good() == false)	
// Pre-conditions: N // Post-conditions: N	None. None.		ret	turn 0;	
// Return values: The m // If th // NULL	nethod will return the input_file_nam ne input_file_name is not set, the me	ne character string. ethod will return	// read the ise_infs.re	<pre>e first byte from the ise file ead(&amp;the_type, sizeof(the_type));</pre>	
// Description: This //	is the accessor method for the input	file name.	// check i: if (the_typ	f the file is a jpeg ise pe == 'l')	
<pre>char* ise::get_input_file </pre>	e_name()		ret	turn 1;	
// check that the if (input_file_na	e input file is not NULL ame == <b>NULL</b> )		// check i: if (the_typ	f the file is a mp3 ise pe == '2')	
return NU	JLL;		ret	turn 2;	
<pre>} return input_file }</pre>	e_name;		} // check i: if (the_typ	f the file is a zip ise pe == '3')	
//			{ ret	turn 3;	
// // Pre-conditions: N	Jone		}		
// Post-conditions: N	None.		ise_infs.c	lose();	
// Parameters: None. // Return values: The m	method will return the output_file_na	nme character string	// otherwis	se the file is unknown	
// If th	ne output_file_name is not set, the m	nethod will return	}		
// Description: This	is the accessor method for the output	at file name.	//		
<pre>//char* ise::get_output_fil</pre>	.e_name()		// Pre-conditions: name.	The user of the class has previous	ly set the input_file_
// check that the <b>if</b> (output_file_n	e output file is not NULL name == <b>NULL</b> )		<pre>// Post-conditions // // // // Parameters•</pre>	: The output_file_name data member point a file name and file path, based upon to by the input_file_name. None	s to a string with the string pointed

Apr 21, 04 17:48	ise.cpp	Page 11/23 Apr	21,
// Return values: // // Description: // // // // // //	An integer is returned indicating a succes A zero will indicate a success. A one will indicate a failure. The file name and path created will be the string pointed to by the input_file_name except that the extension of the file will to .ise. If this file already exists, the added on to the end of the file name, just extension. If this file already exists, un incrementing this number and checking, un name does not previously exist.	ss or failure. e same as the data member, l be changed en a 0 will be t before the we will keep til the new file	
int ise::make_ise_f	ile_name()		
<pre>char* index // size equ size_t leng // used to int number, char letter // stores n stack<int> ifstream In // set an i number = 0; length = st</int></pre>	; als length of extention number th, size; find the name extention number temp, remainder, count, digit; = '0'; ame extention number file_index; File; se file name from the input file name rlen(input_file_name);		
output_file <b>if</b> (output_	_name = <b>new</b> (nothrow) <i>char</i> [length + 1]; file_name == <b>NULL</b> )		
ret	urn 1;		
<pre>} strcpy(outp // check th index = str // if file if (index =</pre>	<pre>ut_file_name, input_file_name); e jpeg file extention str(output_file_name, ".jp"); extention is ".JPG" = NULL)</pre>		
{ ind	<pre>ex = strstr(output_file_name, ".JP");</pre>		
} // check if <b>if</b> (index = {	not a jpeg file = <b>NULL</b> )		
ret	urn 1;		
// add ise *(index+1) *(index+2) *(index+3) *(index+4)	<pre>extention = 'i'; = 's'; = 'c'; = '\0';</pre>		
InFile.open	(output_file_name);		
// if file while (InFi	<pre>name already exists, make a new file name le.good())</pre>		
' InF num tem // whi {	<pre>ile.close(); ber++; p = number; calculate name extention number le (temp != 0)</pre>		
}	<pre>remainder = temp % 10; file_index.push(remainder); temp = temp / 10;</pre>		

```
04 17:48
                                                               Page 12/23
                               ise.cpp
        // create output file name
        if (output_file_name != NULL)
                delete [] output_file_name;
        size = file_index.size();
       output_file_name = new (nothrow) char[length + size + 1];
       if (output_file_name == NULL)
        {
                return 1;
        strcpy(output_file_name, input_file_name);
       index = strstr(output_file_name, ".jp");
        // if file extention is ".JPG"
       if (index == NULL)
                index = strstr(output_file_name, ".JP");
        1
        count = 0;
        // convert top of stack to a character
       while (!file_index.empty())
                digit = file_index.top();
                file_index.pop();
                switch (digit)
                case 0:
                        letter = '0';
                        break;
                case 1:
                        letter = '1';
                        break;
                case 2:
                         letter = '2';
                        break;
                case 3:
                        letter = '3';
                        break;
                case 4:
                        letter = '4';
                        break;
                case 5:
                        letter = '5';
                        break;
                case 6:
                        letter = '6';
                        break;
                case 7:
                        letter = '7';
                        break;
                case 8:
                        letter = '8';
                        break;
                case 9:
                        letter = '9';
                        break;
                // add extention number
                *(index + count) = letter;
                count++;
        // add ise file extention
        *(index + size) = '.';
        *(index + size + 1) = 'i';
        *(index + size + 2) = 's';
        *(index + size + 3) = 'e';
        *(index + size + 4) = ' \setminus 0';
```

ise.cpp

Page 14/23

```
Apr 21, 04 17:48
                                       ise.cpp
                                                                      Page 13/23
                                                                                        Apr 21, 04 17:48
                                                                                                                             ise.cpp
                InFile.open(output_file_name);
                                                                                                       InFile.close();
                                                                                                      number++;
        return 0;
                                                                                                       temp = number;
                                                                                                       // calculate name extention number
                                                                                                      while (temp != 0)
// Pre-conditions:
                        The user of the class has previously set the input_file_
                                                                                                               remainder = temp % 10;
name.
                                                                                                               file_index.push(remainder);
// Post-conditions: The output_file_name data member points to a string with
                                                                                                               temp = temp / 10;
                    a file name and file path, based upon the string pointed
11
                    to by the input_file_name.
// Parameters:
                    None.
                                                                                                       // create output file name
// Return values:
                    An integer is returned indicating a success or failure.
                                                                                                       if (output_file_name != NULL)
                    A zero will indicate a success.
11
                                                                                                       {
                    A one will indicate a failure.
11
                                                                                                               delete [] output_file_name;
// Description:
                    The file name and path created will be the same as the
                    string pointed to by the input_file_name data member,
                                                                                                       size = file_index.size();
11
                    except that the extension of the file will be changed
                                                                                                      output_file_name = new (nothrow) char[length + size + 1];
11
                    to .jpg. If this file already exists, then a 0 will be
                                                                                                       if (output_file_name == NULL)
                    added on to the end of the file name, just before the
                    extension. If this file already exists, we will keep
                                                                                                               return 1;
                    incrementing this number and checking, until the new file
                    name does not previously exist.
                                                                                                       strcpy(output_file_name, input_file_name);
11
                                                                                                      index = strstr(output_file_name, ".is");
11
                                                                                                       // check if file extention is ".ISE"
                                                                                                      if (index == NULL)
int ise::make_output_file_name()
{
                                                                                                       {
        char* index;
                                                                                                               index = strstr(input_file_name, ".IS");
        // size equals length of extention number
        size_t length, size;
        // used to find the name extention number
                                                                                                       // index offset
        int number, temp, remainder, count, digit;
                                                                                                       count = 0;
        char letter = '0';
        // stores name extention number
                                                                                                       // convert top of stack to a character
        stack<int> file_index;
                                                                                                      while (!file_index.empty())
        ifstream InFile;
                                                                                                       {
                                                                                                               digit = file_index.top();
        // set an output file name from the ise file name
                                                                                                               file_index.pop();
        number = 0;
                                                                                                               switch (digit)
        length = strlen(input_file_name);
        output_file_name = new (nothrow) char[length + 1];
                                                                                                               case 0:
        if (output_file_name == NULL)
                                                                                                                       letter = '0';
                                                                                                                       break;
        {
                return 1;
                                                                                                               case 1:
                                                                                                                       letter = '1';
        strcpy(output_file_name, input_file_name);
                                                                                                                       break;
        // check the ise file extention
                                                                                                               case 2:
        index = strstr(output_file_name, ".is");
                                                                                                                       letter = '2';
        // check if the extention is .ISE
                                                                                                                       break;
        if (index == NULL)
                                                                                                               case 3:
                                                                                                                       letter = '3';
        {
                index = strstr(input_file_name, ".IS");
                                                                                                                       break;
                                                                                                               case 4:
        // check if not a valid ise file
                                                                                                                       letter = '4';
        if (index == NULL)
                                                                                                                       break;
                                                                                                               case 5:
        {
                return 1;
                                                                                                                       letter = '5';
                                                                                                                       break;
        // add jpeg extention
                                                                                                               case 6:
        *(index+1) = 'j';
                                                                                                                       letter = '6';
        *(index+2) = 'p';
                                                                                                                       break;
        *(index+3) = 'g';
                                                                                                               case 7:
        *(index+4) = ' \ 0';
                                                                                                                       letter = '7';
                                                                                                                       break;
        InFile.open(output_file_name);
                                                                                                               case 8:
                                                                                                                       letter = '8';
        // if file name already exists, make a new file name
                                                                                                                       break;
        while (InFile.good())
                                                                                                               case 9:
```

Apr 21, 04 17:48	ise.cpp	Page 15/23	Apr 21, 04 17:48	ise.cpp	Page 16/23
}	<pre>letter = '9'; break;</pre>		    	be provided. The second and third argum and will be set to a default value of NU	nents are optional JLL.
/// * ( <u>:</u> cou	<pre>add extention number index + count) = letter; int++;</pre>		<pre>// jpeg_ise::jpeg_ise(     ise(key, input_fi </pre>	<pre>char* key, char* input_file_name, char* c le_name, output_file_name)</pre>	<pre>&gt;utput_file_name)</pre>
// add jpeg *(index + s *(index + s *(index + s *(index + s *(index + s	<pre>g extetion size) = '.'; size + 1) = 'j'; size + 2) = 'p'; size + 3) = 'g'; size + 4) = '\0';</pre>		<pre>{ jpeg_ise::~jpeg_ise { }</pre>	()	
InFile.oper	n(output_file_name);		//		
} return 0; } //			// Pre-conditions: // // //	The input_file_name and key must be the overloaded constructor or the set_input_file_name(char* name) and set_ functions prior to calling this method. This code requires that the input and ou	set using either _key(char* key) uput file pointers
// // Pre-conditions: Non // Post-conditions: None. // Parameters: None. // Return values: The med // If the // NULL. // Description: This is	ne. ne. thod will return the key characte key is not set, the method will s the accessor method for the key	r string. return	// // Post-conditions: // // // // Parameters: // Return values: //	are at the head of the file. An encrypted file will be created with t specified by the output_file_name data member. If this data member is NULL, th name will be created based upon the input data member. None. An integer is returned indicating a succ A zero will indicate a success.	the name and path then a default file t_file_name cess or failure.
<pre>// char* ise::get_key() {     // check that the i     if (here - NULL)</pre>	key is not NULL		// // // //	A one will indicate could not open input A two will indicate could not create ise A three will indicate could not open ise A four will indicate	: file name 9 file name 9 file 9 the jpeg file is no
<pre> {     return NULI } </pre>	L;		// Description: //	The encrypt_file method will take a star compression JPEG file and selectively er Huffman Table frames found within the fi	ndard baseline ncrypt the ile.
<pre> return key; } //</pre>			     	be overwritten. A new, encrypted file w created for the selectively encrypted JE	vill be PEG image.
// // Default Constructor // Pre-conditions: Nou // Post-conditions: Nou // Parameters: Nou // Return values: Cou // Description: Default //	ne. ne. nstructor, no return type. t constructor is not used by user	s.	<pre>// int jpeg_ise::encry {     // check if     ifstream in     if (infs.good()     {         ret</pre>	<pre>pt_file() the input file exists fs(jpeg_ise::get_input_file_name(), ios::     == false) urn 1;</pre>	:binary);
<pre>// ipeq ise::ipeq ise() : ise</pre>			}		
{ }			<pre>// Check if if (jpeg_is {</pre>	<pre>ise_file_name is empty e::get_output_file_name() == NULL)</pre>	
// // Overloaded Constructor // Pre-conditions: The // Post-conditions: An JPE	e key must be a pointer to a char G ISE object is created containin	acter string.	/// jpe <b>if</b> {	<pre>create the ise output file g_ise::make_ise_file_name(); (jpeg_ise::get_output_file_name() == NULI return 2:</pre>	L)
// Parameters: The fir // Parameters: The fir // The sec to be e to be e the fir // the fir // the fir encrypt // Return values: Co // Description: An ISE // constri	embers. rst argument is a pointer to the cond argument is the name and pat encrypted or decrypted. The thir le name and path for the output f tion or decryption. nstructor, no return type. object is constructed with the d t or decrypt a file. This overlo uctor only requires that the firs	key. h of the input file d argument is ile generated by lata necessary to aded t argument	<pre>} } // check if ofstream outfs( if (outfs.good( {     ret }</pre>	<pre>output file can open jpeg_ise::get_output_file_name(), ios::bi ) == false) urn 3;</pre>	inary);



Apr 21, 04 17:48 <b>ise.cp</b>	<b>p</b> Page 19/23	Apr 21, 04 17:48	ise.cpp	Page 20/23	
<pre>memset(cipher_text_output pt_counter = 0;</pre>	,0,BUFFER_LENGTH);	{	blockEncrypt(&cipherinst,&keyinst,plain_text,keyLength,ciphe		
<pre>plain_text[pt_counter++] }</pre>	= b;	r_text);	<pre>for (int i = 0; i &lt; BUFFER LENGTH; i++)</pre>		
// continue f <b>if</b> (pt_counter < BUFFER_LENGT {	illing buffer H)		{ // send to o cipher text output[i]= (char)cipher	utput file text[i];	
plain_text[pt_counter+] // en if(pt_	= c; crypt if the buffer is full _counter == BUFFER_LENGTH)	<pre>tput[i]));</pre>	<pre>outfs.write(&amp;cipher_text_output[i], s. }</pre>	<pre>izeof(cipher_text_ou</pre>	
{	blockEncrypt(&cipherinst,&keyins		<pre>// reset the buffer memset(plain_text,0,BUFFER_LENGTH);</pre>		
<pre>t,plain_text,keyLength,cipher_text);</pre>	<pre>for (int i = 0; i &lt; BUFFER_LENGT</pre>		<pre>memset(cipher_text,0,BUFFER_LENGTH); memset(cipher_text_output,0,BUFFER_LENGT)</pre>	H);	
H; i++)	{		<pre>pt_counter = 0;</pre>	er	
	<pre>// send to output file cipher_text_output[i]=(c</pre>	}	<pre>plain_text[pt_counter++] = b;</pre>		
<pre>nar)cipner_text[1];</pre>	outfs.write(&cipher_text	}			
_output[i], <b>sizeof</b> (cipher_text_output[i]));	}	}			
	<pre>// reset the buffer memset(plain_text,0,BUFFER_LENGT</pre>	if (!is_baselin {	e    !is_ffda)		
H);	<pre>memset(cipher_text,0,BUFFER_LENG</pre>	return 4;			
TH);	<pre>memset(cipher_text_output,0,BUFF</pre>	infs.close();			
ER_LENGTH);	<pre>pt_counter = 0; atom encounter = false;</pre>	<pre>outfs.close(); noturn 0;</pre>			
}	stop_encrypt - farse,	}			
}		//			
uffer // 11 the bur	fer is full, encrypt and add c to b	// Pre-conditions:	The input_file_name and key must be	set using either	
else {			the overloaded constructor or the set_input_file_name(char* name) and set	key(char* key)	
// en blockEncrypt(&cipherinst,	<pre>crypt &amp;keyinst,plain_text,keyLength,ciphe</pre>		functions prior to calling this method. This code requires that the input and ou, are at the head of the file.	put file pointers	
<pre>for (int i = 0; i &lt; BUFFE {</pre>	<pre>for (int i = 0; i &lt; BUFFER_LENGTH; i++) {</pre>		s: An decrypted file will be created with the name and pat specified by the output file name data	he name and path	
cipher_text_output[i]	<pre>// send to output file =(char)cipher_text[i];</pre>		member. If this data member is NULL, th name will be created based upon the inpu	en a default file t_file_name	
<pre>outIs.write(&amp;cipher_t tput[i]));</pre>	ext_output[1] <b>, sizeof</b> (cipner_text_ou	// Parameters:	None.		
} // re	} // reset the buffer		An integer is returned indicating a succ A zero will indicate a success.	ess or failure.	
memset (cipher_text, 0, BUFF memset (cipher_text, 0, BUFF	ER_LENGTH);		A two will indicate could not open ise f	ile	
pt_counter = 0;	, 0, BOFFER_LENGIN);		A four will indicate could not open output	ut jpeg file	
er // aa	a secona nali oi marker to new bull	// Description:	selectively decrypt the Huffman Table fr	ames found	
<pre>plain_text[pt_counter++] + } </pre>	= c;		Within the file. If the file already exists, the existing be overwritten. A new, encrypted file w	file will ill be	
// if no jpeg marker,	fill a buffer and encrypt		created for the selectively new decrypte	d JPEG image.	
ELSE {	o fill buffor	int jpeg_ise::decry	<pre>pt_file()</pre>		
// continue to fill buffer if (pt_counter < BUFFER_LENGTH)		// check if	input file is not a jpeg ise file	file neme())   TDDG	
plain_text[pt_counter++]	= b;	<pre>if (jpeg_ise::get_ise_file_type(jpeg_ise::get_input_file_name()) != JPEG _TYPE)</pre>			
// encrypt if	the buffer is full		urn 1;		
ise.cpp

```
Apr 21, 04 17:48
                                      ise.cpp
                                                                      Page 21/23
                                                                                        Apr 21, 04 17:48
                                                                                                                              ise.cpp
                                                                                                                                                               Page 22/23
      ifstream infs(jpeg ise::get input file name(), ios::binary);
                                                                                                           inhuff = true;
                                                                                                       }
       // check if input file could not open
  if (infs.good() == false)
                                                                                                   outfs.write(&b, sizeof(b));
                                                                                                       // if half of a jpeg marker was found
               return 2;
                                                                                                       // split block case
                                                                                               else if (inhuff == true && split_block == true)
  // check if ise_file_name is NULL
  if (jpeg_ise::get_output_file_name() == NULL)
                                                                                                               // fill buffer to be decrypted
                                                                                                   while (ct_counter < BUFFER_LENGTH)</pre>
               // create output jpeg file
               jpeg_ise::make_output_file_name();
                                                                                                       cipher text[ct counter++] = b;
               if (jpeg_ise::get_output_file_name() == NULL)
                                                                                                       if(ct_counter < BUFFER_LENGTH) infs.read(&b, sizeof(b));</pre>
                       return 3;
                                                                                                               // decrypt buffer
                                                                                                   blockDecrypt(&cipherinst,&keyinst,cipher_text,keyLength,plain_text);
                                                                                                               // if first byte is not second half of huffman marker
      // check if output file could not open
                                                                                                               if (plain_text[0] == 0xDA)
  ofstream outfs(jpeg_ise::get_output_file_name(), ios::binary);
  if (outfs.good() == false)
                                                                                                                        // stop decryption
                                                                                                       inhuff = false;
               return 4;
                                                                                                   split_block = false;
  //output jpeg identifier to head of file
                                                                                                               // send decrypted data to output file
  char file_type;
                                                                                                               for (int i = 0; i < BUFFER LENGTH; i++)</pre>
  infs.read(&file_type,sizeof(file_type));
    // check if file type of ise is
                                                                                                       plain_text_output[i]=(char)plain_text[i];
  /*if (file_type != '1')
                                                                                                       outfs.write(&plain_text_output[i], sizeof(plain_text_output[i]));
  ł
      return 1;
                                                                                                               // reset the buffer
  1*/
                                                                                                   memset(plain_text, 0, BUFFER_LENGTH);
                                                                                                   memset(plain text output, 0, BUFFER LENGTH);
  bool decrypt_huffman_table, decrypt_encoded_data;
                                                                                                   memset(cipher_text, 0, BUFFER_LENGTH);
  decrypt_huffman_table = decrypt_encoded_data = false;
                                                                                                   ct counter = 0;
  bool ff, inhuff, split_block;
  ff = inhuff = split_block = false;
                                                                                                       // in the huffman table
  int keyLength = 128;
                                                                                               else if(inhuff == true)
  unsigned char plain_text[BUFFER_LENGTH];
  memset(plain_text, 0, BUFFER_LENGTH);
                                                                                                               // fill the buffer to be decrypted
  unsigned char cipher_text[BUFFER_LENGTH];
                                                                                                   while (ct_counter < BUFFER_LENGTH)</pre>
  memset(cipher_text, 0, BUFFER_LENGTH);
                                                                                                   {
  char plain_text_output[BUFFER_LENGTH];
                                                                                                       cipher_text[ct_counter++] = b;
  memset(plain_text_output, 0, BUFFER_LENGTH);
                                                                                                                        if(ct_counter < BUFFER_LENGTH) infs.read(&b, size</pre>
  keyInstance keyinst;
                                                                                      of(b));
  cipherInstance cipherinst;
                                                                                                   }
      makeKey(&keyinst, DIR_DECRYPT, keyLength, jpeq_ise::get_key());
  char iv[BUFFER_LENGTH];
                                                                                                               // decrypt the buffer
  memset(iv,0,BUFFER_LENGTH);
                                                                                                   blockDecrypt(&cipherinst, &keyinst, cipher_text, keyLength, plain_text);
  cipherInit(&cipherinst, MODE_ECB, iv);
                                                                                                               // search through decrypted data
  int ct_counter = 0;
                                                                                                   for (int i = 0; i < BUFFER_LENGTH; i++)</pre>
                                                                                                                        // if marker found
  char b:
      // begin ise selective decryption algorithm
                                                                                                       if (plain_text[i] == 0xFF && i != 15)
  while (infs.read(&b, sizeof(b)))
                                                                                                                                // if not huffman marker
               // send unencrypted data to output file
                                                                                                           if (plain_text[i+1] == 0xDA)
      if (inhuff == false && split_block == false)
                                                                                                                                         // stop decryption
           if ((byte)b == 0xFF)
                                                                                                               inhuff = false;
                                                                                                               break;
               outfs.write(&b, sizeof(b));
               infs.read(&b, sizeof(b));
               if ((byte)b == 0xC4 || (byte)b == 0xC0)
                                                                                                                        // if half of jpeg marker found
                                                                                                       else if (plain_text[i] == 0xFF && i == 15)
```

```
Apr 21, 04 17:48
                                                ise.cpp
                                                                                        Page 23/23
                                                  // go to split block case
                        split_block = true;
                   }
              }
              // send decrypted data to output file
for (int i = 0; i < BUFFER_LENGTH; i++)</pre>
              {
                   plain_text_output[i]=(char)plain_text[i];
outfs.write(&plain_text_output[i],sizeof(plain_text_output[i]));
              }
                              // reset the buffer
             memset(plain_text,0,BUFFER_LENGTH);
memset(plain_text_output,0,BUFFER_LENGTH);
              memset(cipher_text, 0, BUFFER_LENGTH);
              ct_counter = 0;
         }
    }
   infs.close();
   outfs.close();
    return 0;
```

## **ISE Manipulator Code Files**

SfrmAbout.cs Page 2/2

М	ay 02, 04 0:32	frmAbout.cs	Page 1/2	May 02, 04 0:32	frmAbout.cs	Page 2/2
111						
///	File Name:	frmAbout.cs		/// <summary> /// Clean up any re</summary>	esources being used.	
         	File Description:	This file implements all of the functionali ISE Manipulator's about form. This file co only the code for the about form and nothin	ty of the ntains g else.	/// protected override	void Dispose( bool disposing )	
       		This code has been developed to assist Team working with JPEG images and testing techni to develop our Selective Encryption algorit	ISE in ques used hm for ISO	if( disposing ) { if(components !	= null)	
	Project Name:	Standard Baseline JPEG Image files. Selective Encryption for JPEG Images		{ components.Di }	spose();	
	2	CSCI 4308-4318: Senior Project August 2003 to May 2004 Department of Computer Science University of Colorado at Boulder		} base.Dispose( dis }	sposing );	
	Project Sponsor:	Tom Lookabaugh Assistant Professor of Computer Science		<pre>#region Windows For /// <summary> /// Required method</summary></pre>	rm Designer generated code d for Designer support - do not modify	
	Project Manager:	University of Colorado at Boulder Bruce Sanders University of Colorado at Boulder		/// the contents of /// private void Initia	t this method with the code editor. alizeComponent()	
	Team ISE Members:	Shinya Daigaku Geoffrey Griffith Joe Jarchow		System.Resources. new S this.picAbout = r this.SuspendLayou	<pre>ResourceManager resources = System.Resources.ResourceManager(typeof lew System.Windows.Forms.PictureBox(); t();</pre>	(frmAbout));
 		Joseph Kadhim Andrew Pouzeshi		// picAbout //		
	This code is open	source and may be used with no cost.		this.picAbout.Ima ((Sys	age = stem.Drawing.Image)(resources.GetObject	("picAbout.Image")
	The authors are in from the usage of no warranties, pr	n no way responsible for any effects this code. It is provided as is with otections, promises or any form of		this.picAbout.Loo this.picAbout.Nan this.picAbout.Siz	<pre>cation = new System.Drawing.Point(8, 8) ne = "picAbout"; ze = new System.Drawing.Size(448, 608);</pre>	;
	support. The aut for good purposes	hors would hope it would only be used . Thank you.		this.picAbout.Siz Syste	<pre>zeMode = em.Windows.Forms.PictureBoxSizeMode.Str Dindows = 0:</pre>	etchImage;
/// usi	ng System;			this.picAbout.Tak this.picAbout.Cli	<pre>officer = 0; oStop = false; ick += new System.EventHandler(this.pic</pre>	About_Click);
usi usi	ng System.Collection ng System.Component	ns; Model;		// frmAbout //		
usi	ng System.Windows.F	orms;		this.AutoScaleBas this.ClientSize = this.Controls.Add	seSize = new System.Drawing.Size(5, 13) = new System.Drawing.Size(464, 621); l(this.picAbout);	;
nam { /	espace JPEG_Manipul	ator		this.Icon = ((Sys this.Name = "frm# this.StartPositic	stem.Drawing.Icon)(resources.GetObject( About"; on =	"\$this.Icon")));
/	<pre>// Summary descript //  ublic class frmabour</pre>	ion for frmAbout. t · System Windows Forms Form		Syste this.Text = "About this.TopMost = the	em.Windows.Forms.FormStartPosition.Cent ut the ISE JPEG Manipulator";	erScreen;
{	private System.Win	dows.Forms.PictureBox picAbout;		this.ResumeLayout	(false);	
	/// Required design ///	ner variable.		} #endregion		
	private System.Com	<pre>ponentModel.Container components = null;</pre>		<pre>private void picAbo {    this.Close();</pre>	<pre>out_Click(object sender, System.EventAr</pre>	gs e)
	<pre>/// <summary> /// This is the frm /// </summary> public frmAbout()</pre>	mAbout() constructor.		}		
	<pre>InitializeCompone }</pre>	ent();				

SfrmLoad.cs Page 2/4

Ma	ay 02, 04 0:32	frmLoad.cs	Page 1/4	Ma	y 02, 04 0:32	frmLoad.cs
1//				1	public frmLoad()	
	File Name:	frmLoad.cs			InitializeComponen LoadFormConstructo	t(); r();
                	File Description:	This file implements all of the functionalit ISE Manipulator's loading form. This file of only the code for the loading form and nothi This code has been developed to assist Team working with JPEG images and testing technic to develop our Selective Encryption algorith Standard Baseline JPEG Image files.	cy of the contains ing else. ISE in ques used nm for ISO		<pre>} /// <summary> /// Clean up any res /// </summary> protected override v {     if( disposing )</pre>	purces being used. Did Dispose( bool disposing )
	Project Name:	Selective Encryption for JPEG Images CSCI 4308-4318: Senior Project August 2003 to May 2004 Department of Computer Science University of Colorado at Boulder			<pre>{     if(components !=         {             components.Dis         } }</pre>	<pre>null) pose();</pre>
	Project Sponsor:	Tom Lookabaugh Assistant Professor of Computer Science University of Colorado at Boulder			base.Dispose( disp } #endregion	osing );
	Project Manager:	Bruce Sanders University of Colorado at Boulder				
	Team ISE Members:	Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi			#region Windows Form /// <summary> /// Required method /// the contents of /// </summary> private void Initial	Designer generated code for Designer support - do not modify this method with the code editor. izeComponent()
	This code is open The authors are i from the usage of no warranties, pr support. The aut for good purposes	source and may be used with no cost. n no way responsible for any effects this code. It is provided as is with otections, promises or any form of hors would hope it would only be used . Thank you.			System.Resources.R new Sy this.barLoadProgre this.lblLoad = new this.btnCancelLoad this.SuspendLayout // barLoadProgress //	<pre>esourceManager resources = stem.Resources.ResourceManager(typeof(frmLoad) ss = new System.Windows.Forms.ProgressBar(); System.Windows.Forms.Label(); = new System.Windows.Forms.Button(); (); </pre>
usin usin usin usin usin	ng System; ng System.Drawing; ng System.Collection ng System.Component ng System.Windows.F	ns; Model; orms;			this.barLoadProgre this.barLoadProgre this.barLoadProgre // // lblLoad	<pre>ss.location = new System.brawing.Foint(8, 32); ss.Name = "barLoadProgress"; ss.Size = new System.Drawing.Size(272, 23); ss.TabIndex = 0;</pre>
nam { /	espace JPEG_Manipul	ator			this.lblLoad.Locat this.lblLoad.Name this.lblLoad.Size this.lblLoad.TabIn	ion = new System.Drawing.Point(16, 8); = "lblLoad"; = new System.Drawing.Size(256, 16); dex = 1;
/ /	// Summary descript // ublic class frmLoad	ion for frmLoadMessage. : System.Windows.Forms.Form			// // // btnCancelLoad	= "Data Loading, Please Wait";
ť	private System.Win private System.Win private System.Win	dows.Forms.ProgressBar barLoadProgress; dows.Forms.Label lblLoad; dows.Forms.Button btnCancelLoad;			// this.btnCancelLoad this.btnCancelLoad this.btnCancelLoad this.btnCancelLoad	.Cursor = System.Windows.Forms.Cursors.Arrow; .Location = new System.Drawing.Point(88, 64); .Name = "btnCancelLoad"; .Size = new System.Drawing.Size(112, 24);
	private bool cance	led;			this.btnCancelLoad this.btnCancelLoad this.btnCancelLoad	.TabIndex = 0; .Text = "&Cancel Load"; .Click += new
	<pre>#region Form Requi /// <summary></summary></pre>	red Code			System.Event // // frmLoad	Handler(this.btnCancelLoad_Click);
	<pre>/// Required design ///  private System.Complexity</pre>	ner variable. ponentModel.Container components = null;			<pre>// this.AutoScaleBase this.ClientSize = this.Controls.Add(</pre>	Size = new System.Drawing.Size(5, 13); new System.Drawing.Size(292, 93); this.btnCancelLoad);

```
frmLoad.cs
                                                                                                                         frmLoad.cs
May 02, 04 0:32
                                                                       Page 3/4
                                                                                      May 02, 04 0:32
                                                                                                                                                             Page 4/4
    this.Controls.Add(this.lblLoad);
                                                                                         /// <param name="StepSize">Step size for the Load Bar.</param>
                                                                                        public void StartLoading(int MinValue, int MaxValue, int StepSize)
    this.Controls.Add(this.barLoadProgress);
    this.Cursor = System.Windows.Forms.Cursors.WaitCursor;
    this.Icon = ((System.Drawing.Icon)(resources.GetObject("$this.Icon")));
this.Name = "frmLoad";
                                                                                           int i = 0;
                                                                                          this.barLoadProgress.Maximum = MaxValue;
    this.StartPosition =
                                                                                          this.barLoadProgress.Minimum = MinValue;
            System.Windows.Forms.FormStartPosition.CenterScreen;
                                                                                          this.barLoadProgress.Step = StepSize;
    this.Text = "Loading Data";
    this.TopMost = true;
                                                                                          if(i < MinValue) i = MinValue;
    this.ResumeLayout(false);
                                                                                          this.barLoadProgress.Value = i;
                                                                                          this.barLoadProgress.Update();
                                                                                          this.Show();
  #endregion
                                                                                          this.Activate();
                                                                                          this.btnCancelLoad.Focus();
  /// <summary>
  /// If this button is clicked, the Cancelled property on this form
  /// will be set to true. This property will remain true until the
                                                                                        /// <summary>
                                                                                        /// This function updates the progress bar. If the been cancel button
  /// is destroyed.
  /// </summary>
                                                                                        /// has been clicked, then this function will return false, but form will
  /// <param name="sender">The sender parameter is a pointer to the
                                                                                        /// STILL be updated.
  /// function calling this function. </param>
                                                                                        /// </summary>
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                        /// <returns>Returns true if cancel button has NOT been pressed.</returns>
  /// data.</param>
                                                                                        public bool UpdateForm()
  private void btnCancelLoad_Click(object sender, System.EventArgs e)
                                                                                          this.Update();
                                                                                          if(canceled)
    canceled = true;
                                                                                            if (MessageBox.Show(
                                                                                               "Are you sure you want to CANCEL this operation?\n" +
  /// <summary>
                                                                                               "Clicking \"OK\" will cancel this operation.\n" +
  /// This is the constructor that ISE will initialize all our variables
                                                                                               "Clicking \"CANCEL\" will continue this operation.\n",
                                                                                               "Operation Aborted!",
  /// for this form and then this method will be called by this Load form
  /// constructor, in this file.
                                                                                              MessageBoxButtons.OKCancel,
  /// </summary>
                                                                                              MessageBoxIcon.Error) == DialogResult.OK)
  private void LoadFormConstructor()
                                                                                              canceled = true;
    canceled = false;
    StartLoading(0, 100, 1);
                                                                                            else canceled = false;
    this.barLoadProgress.Value = 0;
    this.ShowInTaskbar = true;
  }
                                                                                          return !canceled;
                                                                                        }
  /// <summary>
  /// True if the Cancel Button has been hit.
                                                                                        /// <summarv>
  /// </summarv>
                                                                                        /// This function updates and increments the progress bar. If the been
  public bool Canceled
                                                                                        /// cancel button has been clicked, then this function will return false,
                                                                                        /// but form will STILL be updated and incremented.
    get { return canceled; }
                                                                                        /// </summarv>
    set { canceled = value; }
                                                                                        /// <returns>Returns true if cancel button has NOT been pressed.</returns>
  }
                                                                                        public bool UpdateAndIncrement()
                                                                                          this.barLoadProgress.PerformStep();
  /// <summary>
                                                                                          this.Update();
  /// Gets or Sets the value of the Progress Bar.
                                                                                          return !canceled;
  /// </summary>
  public int LoadProgressValue
    get { return barLoadProgress.Value; }
                                                                                      }
    set { barLoadProgress.Value = value; }
  }
  /// <summary>
   /// This resets and prepares the Load form.
  /// </summary>
  /// <param name="MinValue">Minimum value for the Load Bar.</param>
  /// <param name="MaxValue">Maximum value for the Load Bar.</param>
```

May 02, 04 2:03	frmMain.cs	Page 1/186	May 02, 04 2:03	frmMain.cs	Page 2/186
			#region ISE Coded Funct	tions	
//////////////////////////////////////	frmMain.cs		#region ISE JPEG Manip	ulator Variables and Constructor	
/// File Description: /// /// /// /// /// ///	This file implements all of the funct ISE Manipulator's main form. This fi all of the code for the ISE Manipulat code for the "About Form" (frmAbout.c for the "Loading Form" (frmLoading.cs has been developed to assist Team ISE JPEG images and testing techniques us our Selective Encryption algorithm for Baseline JPEG Image files.	tionality of the le contains the cor, except the cs) and the code ). This code S in working with sed to develop or ISO Standard	<pre>// Data member for the private System.Windows private frmLoad Loading private frmSplash Splas // Data memebers for L private System.Drawing private System.Drawing private System.Drawing private System.Drawing</pre>	about form .Forms.Form MainAbout; g; shScreen; oaded JPEG images .Image JPEG; .Image ISE; .Image JPEGsmall; .Image ISEsmall;	
/// Project Name: /// /// ///	Selective Encryption for JPEG Images CSCI 4308-4318: Senior Project August 2003 to May 2004 Department of Computer Science University of Colorado at Boulder		<pre>// Data member to story private Queue FileOrde: // Data members for the // // The May file size it</pre>	e the JPEG image file order r; e raw JPEG image data	
/// Project Sponsor: /// ///	Tom Lookabaugh Assistant Professor of Computer Scier University of Colorado at Boulder	nce	private const int MAX_] private const int MAX_] private const int AVE_]	BYTES = 10485760; // 10 meg FILE_SIZE = 20971520; // 20 meg (2x FILE_SIZE = 10485760; // 10 meg	: MAX_BYTES)
/// Project Manager: /// ///	Bruce Sanders Assistant Professor of Computer Scier University of Colorado at Boulder	nce	<pre>// Assumes no more tab private const int MAX_l private const int MAX_ private const int MAX_l</pre>	les than the Baseline Compression HUFFMAN = 8; QUANTIZER = 4; APPDATA = 10;	
/// Team ISE Members: /// /// /// ///	Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi		<pre>// Data members for the private string Origina private StringBuilder of private StringBuilder of private byte[] NewData</pre>	e original and new raw data stream lEncodedData; OriginalDataStream; EncodedData; ;	
/// This code is open /// This code is open /// The authors are i /// from the usage of /// no warranties, pr /// support. The aut /// for good purposes	n source and may be used with no cost. .n no way responsible for any effects : this code. It is provided as is with cotections, promises or any form of thors would hope it would only be used . Thenk you	1	<pre>// Fixed size variable: private int NumberOfLi private int RestartInto private int FileSize; private int ExpandImage private int RestartMod</pre>	s nes; erval; e; 8;	
<pre>/// lof good purposes /// /// using System;</pre>			private int SizeOfScan private int SizeOfProg private int SizeOfComm	Header; ression; ents;	
using System.Drawing; using System.Collectic using System.Component using System.Windows.F using System.Data; using System.IO;	ns; :Model; ?orms;		private int[] SizeOfHu: private int[] SizeOfQua private int[] SizeOfApp	<pre>ffman = new int[MAX_HUFFMAN]; antizer = new int[MAX_QUANTIZER]; pData = new int[MAX_APPDATA];</pre>	
using System.Text; namespace JPEG_Manipul {	Lator		<pre>// Temporary Variables private int FrameSize; private int Count; private int Temp; private int Value; private int Value;</pre>		
<pre>/// <summary> /// This is the JPEG /// System.Windows.F /// functionality re /// into the interfa /// a new image here</summary></pre>	; Manipulator's main form inhereited fr 'orms.Form class. This form provides r equired for breaking down JPEG images, ace, allowing the user to alter values,	rom the most of the loading them and recreate	private int Low; private int temp; private string Program	Directory;	
/// public class frmMain { public const strin	<pre>i : System.Windows.Forms.Form ig VERSION = "1.0.7";</pre>		// Others private FileStream Ori private FileStream New] private string Manipula	ginalFile; File; atedFileName;	

May 02, 04 2:03	frmMain.cs	Page 3/186	May 02, 04 2:03	frmMain.cs	Page 4/186
private bool Loading]	Interface;				
// Random Number Gene private System.Random	erator n RandomNumber;		/// <summary> /// Pre-conditions</summary>	:	
<pre>// Data members to de private bool PicOrigi private bool PicOrigi private bool PicManip</pre>	etermine if the image is stretched inalStretched; inalSmallStretched; pulatedStretched;		/// The menuOpe /// Post-conditio /// A new origi /// within the /// controls.	n menu object has generated a Click eve ns: nal JPEG image has been loaded and disp picOriginal and the picOrignalSmall Pic	nt. Jayed StureBox
private System.Window private System.Window private System.Window private System.Window	<pre>ws.Forms.Timer timerSplash; ws.Forms.MenuItem menuItem2; ws.Forms.MenuItem menuTutorial; ws.Forms.MenuItem menuManual;</pre>		/// Description: /// This method /// the menuOpe /// is to allow	is used to resolve a Click event gener n menu object. The purpose of this men the user to open a new original JPEG i	ated by u object mage file
private System.Window private System.Window private bool PicManip	vs.Forms.Menultem menultemo; vs.Forms.Menultem menuAbout; pulatedSmallStretched;		/// within the /// LoadNewPict /// this docume ///	application. Inis function will simply ure() function described in section 4.2 nt.	.3.2 of
/// <summary> /// Pre-conditions: /// Post-conditions:</summary>	None.		<pre>/// <param data.<="" e="" event="" name="s /// function calli /// &lt;param name=" p=""/></pre>	ender">The sender parameter is a refere ng this function. ">The parameter is for the base class aram>	nce to the
/// ISE variables /// Parameters:	and initialization routines have } None.	been executed.	<pre>private void menu0 {     LoadNovDicture()</pre>	<pre>pen_Click(object sender, System.EventAr .</pre>	gs e)
/// Function return /// Description:	rns void.		} // End of: menu	, Open_Click(object sender, System.EventA	.rgs e);
/// This function /// logic. This i /// and setting de	is used to execute all ISE initial includes initialization routines for afaults.	lization or variables	/// <summary> /// Pre-conditions</summary>	:	
/// private void ISEConst	cructor()		/// Ine menuExi /// Post-conditio /// The applica	t menu object has generated a CLICK even ns: tion is terminated and exited successfu	nt.
if(FileOrder != nul FileOrder = new Que	ll) FileOrder = null; eue();		/// Description: /// This method	is used to resolve a Click event gener	ated by the
if(OriginalDataStre if(EncodedData != r OriginalDataStream StripaRuildor(AM	eam != null) OriginalDataStream = n null) EncodedData = null; = new E FILE SIZE MAY FILE SIZE).	null;	/// menuExit me /// allow the u /// finished. /// unsaved dat	nu object. The purpose of this menu object. The purpose of this menu object to exit the application when they h This function should check to see if th a before exiting and if so, should ask t to cave the current information. The	ject is to ave where is any the user
EncodedData = new S	StringBuilder(AVE_FILE_SIZE, MAX_F)	ILE_SIZE);	/// function wi /// successfull	<pre>11 call the Application.Exit() method t y exit the Windows application.</pre>	.0
LoadingInterface =	false;		/// /// <param name="s&lt;/td&gt;&lt;td&gt;ender"/> The sender parameter is a refere	nce to the	
RestartInterval = 0; FileSize = 0;	);		/// function calli /// <param name="e&lt;br&gt;/// event data.&lt;/p&lt;/td&gt;&lt;td&gt;ng this function. &lt;/param&gt;&lt;br&gt;"/> The e parameter is for the base class aram>	to pass	
<pre>ExpandImage = 0; RestartMod8 = 0;</pre>			private void menuE	xit_Click(object sender, System.EventAr	gs e)
RandomNumber = new System.Random(uno RandomNumber.Next(5	checked((int)DateTime.Now.Ticks)); 5000);		Application.Exit }	();	
PicOriginalStretche PicOriginalSmallStr	ed = false;		/// <summary> /// Pre-conditions</summary>	: ut menu object has generated a Click eu	zont
PicManipulatedStret PicManipulatedSmall	cched = false; lStretched = false;		/// Post-conditio /// The frmAbou	t Form has been displayed for the user	to view.
<pre>ProgramDirectory =</pre>	Environment.CurrentDirectory;		/// Description: /// This method	is used to resolve a Click event gener	ated by
<pre>// Update frmMain 1 this.Text = "ISE JE }</pre>	Text PEG Manipulator - Version " + VERS:	ION;	<pre>/// is to allow /// details abo /// instance of /// user.</pre>	the user to view the about window to f ut the system. This function creates a the frmAbout form and then displays it	ind out new for the
<pre>#endregion ISE JPEG N</pre>	Manipulator Variables		<pre>///  /// <param e"="" name="s /// function calli /// &lt;param name="/>/// function calli</pre>	ender">The sender parameter is a refere ng this function. ">The e parameter is for the base class	nce to the
#region Interface Met	chods		/// event data. <td>aram&gt;</td> <td>co pubb</td>	aram>	co pubb

May 02, 04 2:03	frmMain.cs	Page 5/186	May 02, 04 2:03	frmMain.cs	Page 6/186
<pre>private void menuAbo {     MainAbout = new fr     MainAbout.Show(); }</pre>	ut_Click(object sender, System.EventArgs	e)	/// Description: /// This method /// menuOpenProj /// to allow the	is used to resolve a Click event generated ject menu object. The purpose of this menu e user to save the current project file, inc	by the object is cluding the
<pre>/// <summary> /// Pre-conditions: /// The menuNewPr /// Post-conditions /// A new project /// Description: /// This method i /// menuNewProjec /// is to allow them to /// allow them to /// of original i /// there is any /// if so, should /// information. /// CreateNewProj /// this document /// <param e"="" name="sen /// function calling /// &lt;param name="/> /// event data.</summary></pre>	oject menu object has generated a Click of file has been created by the application s used to resolve a Click event generated t menu object. The purpose of this menu he user to create a new project file that o store picture, note data and manipulated mages. This function should check to see unsaved data before creating a new project ask the user if they want to save the cu This function should simply call the ect() method outlined in section 4.2.3.1:  der">The sender parameter is a reference this function. The e parameter is for the base class to am> Project_Click(object sender, System.Event a();	event. h. d by the object t will d data if ct and urrent l of to the pass tArgs e)	<pre>/// the project. /// the project. /// function des ///  /// <param data.<="" e"="" name="se /// function callin /// &lt;param name=" param=""/> private void menuSa {     SaveNewProject(); } /// Summary&gt; /// Pre-conditions: /// The txtManig /// event. /// Post-condition /// A warning is /// that alreadg /// Description: /// This method /// the txtManig /// TextBox is t /// file that w: /// manipulated /// and path aln /// function (de /// f</pre>	<pre>current of the changed text reflects a regevent solution of the changed text reflects a reference for aveProject_Click(object sender, System.Event ; pulatedFile TextBox object has generated a reflects a reflects. is used to resolve a TextChanged event generated a reflect a reflect of the reflects a reflect of the reflect of</pre>	<pre>subject () swProject () to the pass event tArgs e)  FextChanged file path erated by this ath of the te a file name ng() a warning</pre>
<pre>/// <summary> /// Pre-conditions: /// The menuOpenP /// Post-conditions /// A previously /// application a /// have been rel /// Description:</summary></pre>	Project menu object has generated a Click created project file has been opened by P ind all values previously saved within the coaded into the application interface.	event. the e project	<pre>/// to the users ///  /// <param data.<="" e"="" name="se /// function callin /// &lt;param name=" param=""/> private void txtMar System. {</pre>	s. ender">The sender parameter is a reference f ng this function. ">The e parameter is for the base class to p nipulatedFile_TextChanged(object sender, .EventArgs e)	to the pass event
<pre>/// This method i menuOpenProje /// to allow the /// This function /// before creati /// they want to /// the project f /// This function /// outlined in s</pre>	s used to resolve a Click event generated of menu object. The purpose of this menu user to open a previously created project i should check to see if there is any unsi- ing a new project and if so, should ask the save the current information. The values is will be reloaded into the application i should simply call the LoadNewProject() section 4.2.3.9 of this document.	d by the a object is t file. aved data he user if s stored in h interface. method	<pre>bool Check; if(File.Exists(t) (txtManipulated { Check = ShowWan "File name: ' "\nALREADY E2 "File Exists'</pre>	<pre>xtManipulatedFile.Text) &amp;&amp; dFile.Text != txtOriginalFile.Text)) rning( " + txtManipulatedFile.Text + XISTS!\nAre you sure you want to overwrite t ");</pre>	this file?",
<pre>///  /// <param e"="" name="sen /// function calling /// &lt;param name="/> /// data. private void menuOpe { LoadNewProject(); }</pre>	<pre>der"&gt;The sender parameter is a reference this function.  The e parameter is for the base class to nProject_Click(object sender, System.Even</pre>	to the pass event ntArgs e)	<pre>if(Check) Maniy else txtManipu } else if(txtManipu { // Create a nar ManipulatedFile string tt = Ma ManipulatedFile Count = ttt.Ind</pre>	<pre>oulatedFileName = txtManipulatedFile.Text; latedFile.Text = ManipulatedFileName; ulatedFile.Text == txtOriginalFile.Text) me for the changed file eName = openFileDialog.FileName; anipulatedFileName.ToLower(); eName = ManipulatedFileName.ToLower(); dexOf(".jpq");</pre>	
<pre>/// <summary> /// Pre-conditions: /// The menuSaveP /// Post-conditions /// This function /// project notes /// an SEP file.</summary></pre>	roject menu object has generated a Click :: 1 saves the current values loaded in the P 2 and any manipulate data values and stor	event. Manipulator, es them in	<pre>// Manipulated ManipulatedFild Temp = 0; string num_leng while(File.Exis {</pre>	<pre>the file name if it already exists eName = ManipulatedFileName.Insert(Count, "_ gth; sts(ManipulatedFileName))</pre>	_changed0");

May 02, 04 2:03	frmMain.cs	Page 7/186	May 02, 04 2:03	frmMain.cs	Page 8/186
Count = Manipula	atedFileName.IndexOf(Temp.ToString(	+ ".jpg");	lblQuantizerOrigi	.nalMarker2.Text = lblQuantizerMarker	c2.Text;
num_length = Ten ManipulatedFileN	mp.ToString(); Name =		}		
nanipataccariter	ManipulatedFileName.Remove(Count, )	<pre>uum_length.Length);</pre>	,		
Temp++;	Nome				
ManipulatedFiler	Name = ManipulatedFileName.Insert(Count, '	<pre>femp.ToString());</pre>	/// <summary> /// Pre-conditions:</summary>		
}		- I · · · · · · · · · · · · · · · · · ·	/// The txtQuantiz	er3 TextBox object has generated a (	Click event.
tytManinulatedFile	e Text - ManipulatedFileName.		/// Post-conditions:	first time the data has been alter	ad the data is
this.Update();	e.iekt – Manipulateur Hendme,		/// copied into th	ie txtQuantizerOriginal3 TextBox.	, the data is
}			/// Description:		
}	eName = txtManipulatedFile.lext;		/// Inis method is /// txtQuantizer3	TextBox object. The purpose of this	s TextBox is to
			/// allow the user	to manipulate the values in the thi	ird Quantizer
/// <summarv></summarv>			/// table containe /// this data has	d within the JPEG image. If this is been altered, this function copies t	3 the first time the data from the
/// Pre-conditions:			/// txtQuantizer3	TextBox (before it has been changed)	) into the
/// The txtQuantize	er1 TextBox object has generated a (	lick event.	/// txtQuantizerOr	iginal3 TextBox.	
/// If this is the	first time the data has been altered	ed, the data is	/// <param name="send&lt;/td&gt;&lt;td&gt;ler"/> The sender parameter is a refer€	ence to the	
/// copied into the	e txtQuantizerOriginall TextBox.		/// function calling	this function.	
/// Description: /// This method is	used to resolve a Click event gene	rated by the	/// <param name="e"/> 1 /// data.	ne e parameter is for the base class	s to pass event
/// txtQuantizer1 1	TextBox object. The purpose of this	3 TextBox is to	private void txtQuant	.izer3_Click(object sender, System.Ev	ventArgs e)
/// allow the user	to manipulate the values in the fire	st Quantizer	{ if(!LoadingInterfac	e && this tytOuantizerOriginal3 Text	⊢ "")
/// this data has h	been altered, this function copies	the data from	{		, )
/// the txtQuantize	erl TextBox (before it has been chan	iged) into the	txtQuantizerOrigi	.nal3.Text = txtQuantizer3.Text;	m2 Tout
///	Iginali Textbox.		}	naimarkers.lext = ibiQuantizerMarker	.s.lext;
/// <param name="sende&lt;/td&gt;&lt;td&gt;er"/> The sender parameter is a refere	ence to the	}			
/// function calling t /// <param name="e"/> Th	this function. he e parameter is for the base class	a to pass event			
/// data.			/// <summary></summary>		
private void txtQuanti	izer1_Click(object sender, System.E <sup>,</sup>	ventArgs e)	/// Pre-conditions:	ver4 TextBox object has generated a (	lick event
if(!LoadingInterface	e && this.txtQuantizerOriginal1.Text	; == "")	/// Post-conditions:	ert rexchox object has generated a c	JIEK EVENC.
{ twtOuantigorOrigin	nall Toyt - tytOyantigorl Toyt.		/// If this is the	first time the data has been altere	ed, the data is
lblQuantizerOrigin	nalMarker1.Text = lblQuantizerMarke:	fl.Text;	/// Description:	e txtQuantizeroriginary rextbox.	
}			/// This method is	used to resolve a Click event gener	cated by the
}			/// txtQuantizer4 /// allow the user	to manipulate the values in the four	arth Ouantizer
			/// table containe	d within the JPEG image. If this is	s the first time
/// <summary> /// Pre-conditions.</summary>			/// this data has	been altered, this function copies t	the data from the
/// The txtQuantize	er2 TextBox object has generated a (	Click event.	/// txtQuantizerOr	iginal4 TextBox.	11100 0110
/// Post-conditions:	first time the data has been alter	ad the data is	///	lor">The conder parameter is a refer	onco to the
/// copied into the	e txtQuantizerOriginal2 TextBox.	a, the data 15	/// function calling	this function.	
/// Description:	wood to possible a Click event conc	ested by the	/// <param name="e"/> T	he e parameter is for the base class	s to pass event
/// inis method is /// txtQuantizer2 1	TextBox object. The purpose of this	s TextBox is to	private void txtOuant	izer4_Click(object sender, System.Ev	ventArgs e)
/// allow the user	to manipulate the values in the sec	cond Quantizer	{		
/// table contained /// this data has h	a within the JPEG image. If this is been altered, this function copies (	the first time the	<pre>if(!LoadingInterfac {</pre>	<pre>.e &amp;&amp; this.txtQuantizerOriginal4.Text</pre>	; == "")
/// txtQuantizer2 1	TextBox (before it has been changed	into the	` txtQuantizerOrigi	.nal4.Text = txtQuantizer4.Text;	
/// txtQuantizerOri	iginal2 TextBox.		lblQuantizerOrigi	.nalMarker4.Text = lblQuantizerMarker	:4.Text;
/// <param name="sende&lt;/td&gt;&lt;td&gt;er"/> The sender parameter is a refere	ence to the	}			
/// function calling t	this function.				
/// <param name="e"/> Th /// data.	ne e parameter is for the base class	; to pass event	/// <summarv></summarv>		
private void txtQuanti	izer2_Click(object sender, System.E	/entArgs e)	/// Pre-conditions:		
{ if(lloadingThtorford	o ii thig tytoughtigororigingly Tour	"")	/// The txtHuffman	.1 TextBox object has generated a Cli	ick event.
<pre>if (:LoadingInterface {</pre>	e αα this.txtQuantizerOriginal2.Tex	```)	/// Fost-conditions: /// If this is the	e first time the data has been altere	ed, the data is
txtQuantizerOrigir	nal2.Text = txtQuantizer2.Text;		/// copied into th	e txtHuffmanOriginall TextBox.	

May 02, 04 2:03	frmMain.cs	Page 9/186	May 02, 04 2:03	frmMain.cs	Page 10/186
/// Descript	ion:		/// <param name="e"/> The	e e parameter is for the base cl	lass to pass event
/// This m /// txtHuf	ethod is used to resolve a Click event generated fmanl TextBox object. The purpose of this Text?	d by the Box is to	/// data. private void txtHuffmar	n3_GotFocus(object sender, Syste	em.EventArgs e)
/// allow /// contai	the user to manipulate the values in the first ned within the JPEG image. If this is the first	time this	if(!LoadingInterface	&& this.txtHuffmanOriginal3.Tex	<t )<="" =="" td=""></t>
/// txtHuf	fmanl TextBox (before it has been changed) into	the	txtHuffmanOriginal	3.Text = txtHuffman3.Text; Marker3.Text = lblHuffmanMarker3	3.Text:
/// /// <param na<="" td=""/> <th>&gt; &gt; me="sender"&gt;The sender parameter is a reference</th> <th>to the</th> <td>}</td> <td></td> <td></td>	> > me="sender">The sender parameter is a reference	to the	}		
/// function /// <param na<="" td=""/> <th>calling this function.  me="e"&gt;The e parameter is for the base class to</th> <th>pass event</th> <td></td> <td></td> <td></td>	calling this function. me="e">The e parameter is for the base class to	pass event			
/// data.private void	ram> txtHuffman1_GotFocus(object sender, System.Even <sup>4</sup>	tArgs e)	<pre>/// <summary> /// Pre-conditions: /// The bot Weffmen()</summary></pre>	Test Des shirt has seen at a	
if(!Loading	<pre>Interface &amp;&amp; this.txtHuffmanOriginal1.Text == "</pre>	")	/// Ine txtHuirman4 /// Post-conditions:	lextBox object has generated a	cred the data is
txtHuffma lblHuffma	nOriginal1.Text = txtHuffman1.Text; nOriginalMarker1.Text = lblHuffmanMarker1.Text;		/// copied into the /// Description:	txtHuffmanOriginal4 TextBox.	Lereu, the data is
}			/// This method is u /// txtHuffman4 Text /// allow the user t	used to resolve a Click event ge Box object. The purpose of thi to manipulate the values in the	enerated by the is TextBox is to fourth Huffman table
/// <summary></summary>			/// contained within /// data has been al	n the JPEG image. If this is th ltered, this function copies the	ne first time this e data from the
/// Pre-condi /// The tx	tions: tHuffman2 TextBox object has generated a Click (	event.	/// txtHuffman4 Text /// txtHuffmanOrigin	zBox (before it has been changed hal4 TextBox.	d) into the
/// Post-con /// If thi /// copied	s is the first time the data has been altered, f into the txtHuffmanOriginal2 TextBox.	the data is	/// /// <param name="sender&lt;br&gt;/// function calling th&lt;/td&gt;&lt;td&gt;r"/> The sender parameter is a ref	ference to the	
/// Descript /// This m	ion: ethod is used to resolve a Click event generate	d by the	<pre>/// <param name="e"/>The /// data.</pre>	e e parameter is for the base cl	lass to pass event
/// txtHuf /// allow	fman2 TextBox object. The purpose of this TextI the user to manipulate the values in the second	Box is to Huffman	private void txtHuffmar	n4_GotFocus(object sender, Syste	em.EventArgs e)
/// table /// this d	ata has been altered, this function copies the ( fman2 TextBox (before it has been changed) into	data from the	<pre>ir(!LoadingInterrace {     txtHuffmanOriginal</pre>	<pre>&amp;&amp; this.txtHuffmanOriginal4.lex 4 Text = txtHuffman4 Text.</pre>	(t == "")
/// txtHuf /// <th><pre>fmanOriginal2 TextBox. &gt;</pre></th> <th></th> <td>lblHuffmanOriginalN }</td> <td>Marker4.Text = lblHuffmanMarker4</td> <td>1.Text;</td>	<pre>fmanOriginal2 TextBox. &gt;</pre>		lblHuffmanOriginalN }	Marker4.Text = lblHuffmanMarker4	1.Text;
/// <param na<br=""/> /// function	<pre>me="sender"&gt;The sender parameter is a reference calling this function. </pre>	to the	}		
/// <param na<br=""/> /// data. <th>me="e"&gt;The e parameter is for the base class to ram&gt;</th> <th>pass event</th> <td>/// <summary></summary></td> <td></td> <td></td>	me="e">The e parameter is for the base class to ram>	pass event	/// <summary></summary>		
{ if(!Loading	<pre>Interface &amp;&amp; this.txtHuffmanOriginal2.Text == "</pre>	")	/// The txtHuffman5 /// Post-conditions:	TextBox object has generated a	Click event.
{ txtHuffma	nOriginal2.Text = txtHuffman2.Text;		/// If this is the f /// copied into the	First time the data has been alt txtHuffmanOriginal5 TextBox.	tered, the data is
lblHuffma }	<pre>nOriginalMarker2.Text = lblHuffmanMarker2.Text;</pre>		/// Description: /// This method is u	used to resolve a Click event ge	enerated by the
}			/// allow the user t	to manipulate the values in the the JPEG image. If this is the	fifth Huffman table
/// <summary> /// Pre-condi</summary>	tions:		/// data has been al /// txtHuffman5 Text	ltered, this function copies the Box (before it has been changed	data from the division of the data from the division of the di
/// The tx /// Post-con	tHuffman3 TextBox object has generated a Click of ditions:	event.	/// txtHuffmanOrigin ///	nal5 TextBox.	
/// II thi /// copied	s is the first time the data has been altered, t into the txtHuffmanOriginal3 TextBox.	the data is	/// <param e"="" name="sender&lt;br&gt;/// function calling th&lt;br&gt;/// &lt;param name="/> The	r">The sender parameter is a rep his function. > e parameter is for the base of	lerence to the
/// This m /// txtHuf	ethod is used to resolve a Click event generated fman3 TextBox object. The purpose of this Text	d by the Box is to	/// data. private void txtHuffmar	n5_GotFocus(object sender, Syste	em.EventArgs e)
/// allow /// contai	the user to manipulate the values in the third ${\tt I}$ ned within the JPEG image. If this is the first	Huffman table time this	{ if(!LoadingInterface	&& this.txtHuffmanOriginal5.Te>	<t )<="" =="" td=""></t>
/// data h /// txtHuf /// tytPuf	as been altered, this function copies the data is fman3 TextBox (before it has been changed) into fmanOriginal3 TextBox	trom the the	txtHuffmanOriginal5	5.Text = txtHuffman5.Text;	Text.
/// <th><pre>immortgingid textbox. &gt; me="conder"\The conder parameter is a reference.</pre></th> <th>to the</th> <td>}</td> <td>Harkers.iekt - ipinulimanmalkelt</td> <td>,,</td>	<pre>immortgingid textbox. &gt; me="conder"\The conder parameter is a reference.</pre>	to the	}	Harkers.iekt - ipinulimanmalkelt	,,
/// function	calling this function <th>co che</th> <td>ſ</td> <td></td> <td></td>	co che	ſ		

frmMain.cs frmMain.cs May 02, 04 2:03 May 02, 04 2:03 Page 11/186 Page 12/186 111 contained within the JPEG image. If this is the first time this data has been altered, this function copies the data from the /// <summary> 111 /// Pre-conditions: 111 txtHuffman8 TextBox (before it has been changed) into the 111 The txtHuffman6 TextBox object has generated a Click event. 111 txtHuffmanOriginal8 TextBox. /// Post-conditions: /// </summary> 111 If this is the first time the data has been altered, the data is /// <param name="sender">The sender parameter is a reference to the 111 copied into the txtHuffmanOriginal6 TextBox. /// function calling this function. </param> /// Description: /// <param name="e">The e parameter is for the base class to pass event 111 This method is used to resolve a Click event generated by the /// data.</param> 111 txtHuffman6 TextBox object. The purpose of this TextBox is to private void txtHuffman8\_GotFocus(object sender, System.EventArgs e) 111 allow the user to manipulate the values in the sixth Huffman table 111 contained within the JPEG image. If this is the first time this if(!LoadingInterface && this.txtHuffmanOriginal8.Text == "") 111 data has been altered, this function copies the data from the 111 txtHuffman6 TextBox (before it has been changed) into the txtHuffmanOriginal8.Text = txtHuffman8.Text; 111 txtHuffmanOriginal6 TextBox. lblHuffmanOriginalMarker8.Text = lblHuffmanMarker8.Text; /// </summary> /// <param name="sender">The sender parameter is a reference to the } /// function calling this function. </param> /// <param name="e">The e parameter is for the base class to pass event /// data.</param> /// <summary> private void txtHuffman6\_GotFocus(object sender, System.EventArgs e) /// Pre-conditions: 111 The btnRestoreQuantizer1 Button object has generated a Click if(!LoadingInterface && this.txtHuffmanOriginal6.Text == "") 111 event. /// Post-conditions: txtHuffmanOriginal6.Text = txtHuffman6.Text; 111 The information stored within the txtQuantizerOriginal1 (the lblHuffmanOriginalMarker6.Text = lblHuffmanMarker6.Text; 111 original picture data) is copied back into the txtQuantizer1 111 TextBox object. } /// Description: 111 This method is used to resolve a Click event generated by the 111 btnRestoreQuantizer1 Button object. The purpose of this Button /// <summary> 111 is to allow the user to restore the original data for this /// Pre-conditions: 111 Quantizer table to the txtQuantizer1 TextBox. 111 The txtHuffman7 TextBox object has generated a Click event. /// </summary> /// Post-conditions: /// <param name="sender">The sender parameter is a reference to the 111 If this is the first time the data has been altered, the data is /// function calling this function. </param> 111 copied into the txtHuffmanOriginal7 TextBox. /// <param name="e">The e parameter is for the base class to pass event /// Description: /// data.</param> 111 This method is used to resolve a Click event generated by the private void btnRestoreQuantizer1\_Click(object sender, System.EventArgs e) 111 txtHuffman7 TextBox object. The purpose of this TextBox is to 111 allow the user to manipulate the values in the seventh Huffman if(lblOuantizerOriginalMarker1.Text != "") table contained within the JPEG image. If this is the first time 111 111 this data has been altered, this function copies the data from the txtQuantizer1.Text = txtQuantizerOriginal1.Text; 111 txtHuffman7 TextBox (before it has been changed) into the txtOuantizerOriginal1.Text = ""; 111 txtHuffmanOriginal7 TextBox. lblQuantizerMarker1.Text = lblQuantizerOriginalMarker1.Text; /// </summary> lblQuantizerOriginalMarker1.Text = ""; /// <param name="sender">The sender parameter is a reference to the /// function calling this function. </param> /// <param name="e">The e parameter is for the base class to pass event /// data.</param> private void txtHuffman7\_GotFocus(object sender, System.EventArgs e) /// <summarv> /// Pre-conditions: if (!LoadingInterface && this.txtHuffmanOriginal7.Text == "") 111 The btnRestoreQuantizer2 Button object has generated a Click 111 event. txtHuffmanOriginal7.Text = txtHuffman7.Text; /// Post-conditions: lblHuffmanOriginalMarker7.Text = lblHuffmanMarker7.Text; 111 The information stored within the txtQuantizerOriginal2 (the 111 original picture data) is copied back into the txtQuantizer2 } 111 TextBox object. /// Description: 111 This method is used to resolve a Click event generated by the /// <summarv> 111 btnRestoreQuantizer2 Button object. The purpose of this Button /// Pre-conditions: 111 is to allow the user to restore the original data for this 111 The txtHuffman8 TextBox object has generated a Click event. 111 Quantizer table to the txtQuantizer2 TextBox. /// </summarv> /// Post-conditions: 111 If this is the first time the data has been altered, the data is /// <param name="sender">The sender parameter is a reference to the 111 copied into the txtHuffmanOriginal8 TextBox. /// function calling this function. </param> /// Description: /// <param name="e">The e parameter is for the base class to pass event 111 This method is used to resolve a Click event generated by the /// data.</param> 111 txtHuffman8 TextBox object. The purpose of this TextBox is to private void btnRestoreQuantizer2\_Click(object sender, System.EventArgs e) 111 allow the user to manipulate the values in the eighth Huffman table

May 02, 04 2:03	frmMain.cs	Page 13/186 N	ay 02, 04 2:03	frmMain.cs	Page 14/186
if(lblQuantizerOrig	inalMarker2.Text != "")				
<pre>{     txtQuantizer2.Tex     txtQuantizerOrigi     lblQuantizerMarke:     lblQuantizerOrigi;     } }</pre>	<pre>t = txtQuantizerOriginal2.Text; nal2.Text = ""; r2.Text = lblQuantizerOriginalMar} nalMarker2.Text = "";</pre>	xer2.Text;	<pre>/// <summary> /// Pre-conditions: /// The btnRestor; /// Post-conditions /// The informati; /// TextBox objec /// Description:</summary></pre>	eHuffmanl Button object has generate : on stored within the txtHuffmanOrigi ure data) is copied back into the tx t.	d a Click event. nall (the tHuffman1
<pre>/// <summary> /// Pre-conditions: /// The btnRestored /// event. /// Post-conditions: /// The information /// original pictu: /// Description: /// Description: /// This method is /// btnRestoreQuan: /// is to allow th /// Quantizer table /// </summary> /// <param name="send /// function calling; /// data.&lt;/param&gt;&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;Quantizer3 Button object has gener&lt;br&gt;n stored within the txtQuantizerOu&lt;br&gt;re data) is copied back into the t&lt;br&gt;used to resolve a Click event gen&lt;br&gt;tizer3 Button object. The purpose&lt;br&gt;e user to restore the original dat&lt;br&gt;e to the txtQuantizer3 TextBox.&lt;br&gt;er"/>The sender parameter is a refe this function.  he e parameter is for the base cla</pre>	rated a Click riginal3 (the extQuantizer3 herated by the e of this Button ta for this erence to the ass to pass event	<pre>/// This method i, /// btnRestoreHuf: /// to allow the y /// table to the /// cyaram name="send /// function calling /// <param name="e"/>' /// data. private void btnRestore { if(lblHuffmanOrigi: { txtHuffman1.Text txtHuffmanOrigin lblHuffmanOrigin } }</pre>	<pre>s used to resolve a Click event gene fmanl Button object. The purpose of user to restore the original data fo txtHuffmanl TextBox. der"&gt;The sender parameter is a refer this function.  The e parameter is for the base clas oreHuffmanl_Click(object sender, Sys nalMarker1.Text != "") . = txtHuffmanOriginall.Text; tall.Text = ""; '1.Text = lblHuffmanOriginalMarker1.T talMarker1.Text = "";</pre>	rated by the this Button is r this Huffman ence to the s to pass event tem.EventArgs e) ext;	
private void btnResto:	reQuantizer3_Click(object sender,	System.EventArgs e)			
<pre>if(lblQuantizerOrig. {     txtQuantizer3.Tex:     txtQuantizerOrigin     lblQuantizerMarke:     lblQuantizerOrigin   } } /// <summary> /// Pre-conditions: /// The btnRestore( /// event. /// Post-conditions: /// The information /// original pictu: /// TextBox object /// Description:</summary></pre>	<pre>inalMarker3.Text != "") t = txtQuantizerOriginal3.Text; nal3.Text = ""; r3.Text = lblQuantizerOriginalMar] nalMarker3.Text = ""; Quantizer4 Button object has gener n stored within the txtQuantizerOr re data) is copied back into the t .</pre>	cer3.Text; cated a Click riginal4 (the cxtQuantizer4	<pre>/// <summary> /// Pre-conditions: /// The btnRestor. /// Post-conditions /// The informati. /// TextBox object /// Description: /// This method i /// btnRestoreHuf /// to allow the /// table to the /// <summary> /// <param name="sen /// function calling /// data.&lt;/param&gt; private void btnRest {&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;eHuffman2 Button object has generate:&lt;br&gt;on stored within the txtHuffmanOrigi&lt;br&gt;ure data) is copied back into the tx&lt;br&gt;t.&lt;br&gt;s used to resolve a Click event gene&lt;br&gt;fman2 Button object. The purpose of&lt;br&gt;user to restore the original data fo&lt;br&gt;txtHuffman2 TextBox.&lt;br&gt;der"/>The sender parameter is a refer this function.  The e parameter is for the base clas</summary></summary></pre>	d a Click event. nal2 (the tHuffman2 rated by the this Button is r this Huffman ence to the s to pass event tem.EventArgs e)	
<pre>/// This method is /// btnRestoreQuan /// is to allow th /// Quantizer table ///  /// <param calling="" data.<="" function="" name="send" param=""/></pre>	used to resolve a Click event gen tizer4 Button object. The purpose e user to restore the original dat e to the txtQuantizer4 TextBox. er">The sender parameter is a refe this function. he e parameter is for the base cla	herated by the e of this Button ta for this erence to the ass to pass event	<pre>if(lblHuffmanOrigin {     txtHuffman2.Text     txtHuffmanOrigin     lblHuffmanMarker     lblHuffmanOrigin } }</pre>	<pre>nalMarker2.Text != "") = txtHuffmanOriginal2.Text; al2.Text = ""; 2.Text = lblHuffmanOriginalMarker2.T alMarker2.Text = "";</pre>	ext;
<pre>private void btnResto: {     if(lblQuantizerOrig:         {         txtQuantizer4.Tex:         txtQuantizerOrigin         lblQuantizerMarke:         lblQuantizerOrigin     } }</pre>	<pre>reQuantizer4_Click(object sender, inalMarker4.Text != "") t = txtQuantizerOriginal4.Text; nal4.Text = ""; r4.Text = lblQuantizerOriginalMarMarMarMarMarker4.Text = "";</pre>	System.EventArgs e) cer4.Text;	<pre>/// <summary> /// Pre-conditions: /// The btnRestore /// Post-conditions /// The information /// original pict /// TextBox objec /// Description: /// This method i /// btnRestoreHuf</summary></pre>	eHuffman3 Button object has generate : .on stored within the txtHuffmanOrigi .ure data) is copied back into the tx .t. .s used to resolve a Click event gene fman3 Button object. The purpose of	d a Click event. nal3 (the tHuffman3 rated by the this Button is

```
frmMain.cs
                                                                                                                     frmMain.cs
May 02, 04 2:03
                                                                  Page 15/186
                                                                                   May 02, 04 2:03
                                                                                                                                                      Page 16/186
  111
         to allow the user to restore the original data for this Huffman
                                                                                          txtHuffman5.Text = txtHuffmanOriginal5.Text;
  111
         table to the txtHuffman3 TextBox.
                                                                                          txtHuffmanOriginal5.Text = "";
  /// </summary>
                                                                                          lblHuffmanMarker5.Text = lblHuffmanOriginalMarker5.Text;
  /// <param name="sender">The sender parameter is a reference to the
                                                                                          lblHuffmanOriginalMarker5.Text = "";
  /// function calling this function. </param>
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                      }
  /// data.</param>
  private void btnRestoreHuffman3_Click(object sender, System.EventArgs e)
                                                                                      /// <summary>
    if(lblHuffmanOriginalMarker3.Text != "")
                                                                                      /// Pre-conditions:
                                                                                           The btnRestoreHuffman6 Button object has generated a Click event.
                                                                                      111
      txtHuffman3.Text = txtHuffmanOriginal3.Text;
                                                                                      /// Post-conditions:
      txtHuffmanOriginal3.Text = "";
                                                                                           The information stored within the txtHuffmanOriginal6 (the
                                                                                      111
      lblHuffmanMarker3.Text = lblHuffmanOriginalMarker3.Text;
                                                                                      111
                                                                                          original picture data) is copied back into the txtHuffman6
      lblHuffmanOriginalMarker3.Text = "";
                                                                                      111
                                                                                           TextBox object.
                                                                                      /// Description:
  }
                                                                                      111
                                                                                            This method is used to resolve a Click event generated by the
                                                                                      111
                                                                                           btnRestoreHuffman7 Button object. The purpose of this Button is
                                                                                      111
                                                                                           to allow the user to restore the original data for this Huffman
                                                                                            table to the txtHuffman7 TextBox.
  /// <summary>
                                                                                      111
  /// Pre-conditions:
                                                                                      /// </summary>
  111
        The btnRestoreHuffman4 Button object has generated a Click event.
                                                                                      /// <param name="sender">The sender parameter is a reference to the
                                                                                      /// function calling this function. </param>
  /// Post-conditions:
  111
      The information stored within the txtHuffmanOriginal4 (the
                                                                                      /// <param name="e">The e parameter is for the base class to pass event
  /// original picture data) is copied back into the txtHuffman4
                                                                                      /// data.</param>
  111
       TextBox object.
                                                                                      private void btnRestoreHuffman6_Click(object sender, System.EventArgs e)
  /// Description:
  111
        This method is used to resolve a Click event generated by the
                                                                                        if(lblHuffmanOriginalMarker6.Text != "")
  111
         btnRestoreHuffman4 Button object. The purpose of this Button is
  111
        to allow the user to restore the original data for this Huffman
                                                                                          txtHuffman6.Text = txtHuffmanOriginal6.Text;
  /// table to the txtHuffman4 TextBox.
                                                                                          txtHuffmanOriginal6.Text = "";
  /// </summary>
                                                                                          lblHuffmanMarker6.Text = lblHuffmanOriginalMarker6.Text;
  /// <param name="sender">The sender parameter is a reference to the
                                                                                          lblHuffmanOriginalMarker6.Text = "";
  /// function calling this function. </param>
  /// <param name="e">The e parameter is for the base class to pass event
  /// data.</param>
  private void btnRestoreHuffman4_Click(object sender, System.EventArgs e)
                                                                                      /// <summary>
    if(lblHuffmanOriginalMarker4.Text != "")
                                                                                      /// Pre-conditions:
                                                                                             The btnRestoreHuffman7 Button object has generated a Click event.
      txtHuffman4.Text = txtHuffmanOriginal4.Text;
                                                                                      /// Post-conditions:
      txtHuffmanOriginal4.Text = "";
                                                                                      111
                                                                                            The information stored within the txtHuffmanOriginal7 (the
      lblHuffmanMarker4.Text = lblHuffmanOriginalMarker4.Text;
                                                                                      111
                                                                                           original picture data) is copied back into the txtHuffman7
      lblHuffmanOriginalMarker4.Text = "";
                                                                                      111
                                                                                           TextBox object.
                                                                                      /// Description:
  }
                                                                                      111
                                                                                            This method is used to resolve a Click event generated by the
                                                                                            btnRestoreHuffman7 Button object. The purpose of this Button is
                                                                                      111
                                                                                            to allow the user to restore the original data for this Huffman
                                                                                      111
  /// <summarv>
                                                                                      111
                                                                                            table to the txtHuffman7 TextBox.
  /// Pre-conditions:
                                                                                      /// </summary>
  111
         The btnRestoreHuffman5 Button object has generated a Click event.
                                                                                      /// <param name="sender">The sender parameter is a reference to the
  /// Post-conditions:
                                                                                      /// function calling this function. </param>
  111
        The information stored within the txtHuffmanOriginal5 (the
                                                                                      /// <param name="e">The e parameter is for the base class to pass event
                                                                                      /// data.</param>
  111
        original picture data) is copied back into the txtHuffman5
  111
       TextBox object.
                                                                                      private void btnRestoreHuffman7_Click(object sender, System.EventArgs e)
  /// Description:
  111
        This method is used to resolve a Click event generated by the
                                                                                        if(lblHuffmanOriginalMarker7.Text != "")
  111
        btnRestoreHuffman5 Button object. The purpose of this Button is
  111
        to allow the user to restore the original data for this Huffman
                                                                                          txtHuffman7.Text = txtHuffmanOriginal7.Text;
  111
        table to the txtHuffman5 TextBox.
                                                                                          txtHuffmanOriginal7.Text = "";
  /// </summary>
                                                                                          lblHuffmanMarker7.Text = lblHuffmanOriginalMarker7.Text;
  /// <param name="sender">The sender parameter is a reference to the
                                                                                          lblHuffmanOriginalMarker7.Text = "";
  /// function calling this function. </param>
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                      }
  /// data.</param>
  private void btnRestoreHuffman5_Click(object sender, System.EventArgs e)
                                                                                      /// <summary>
    if(lblHuffmanOriginalMarker5.Text != "")
                                                                                      /// Pre-conditions:
                                                                                      111
                                                                                             The btnRestoreHuffman8 Button object has generated a Click event.
```

frmMain.cs frmMain.cs May 02, 04 2:03 Page 17/186 May 02, 04 2:03 Page 18/186 /// Post-conditions: /// Pre-conditions: 111 The information stored within the txtHuffmanOriginal8 (the 111 The btnLoad Menu Button object has generated a Click event. 111 original picture data) is copied back into the txtHuffman8 /// Post-conditions: 111 TextBox object. 111 A previously created project file has been loaded by the /// Description: 111 application. 111 This method is used to resolve a Click event generated by the /// Description: 111 btnRestoreHuffman8 Button object. The purpose of this button is 111 This method is used to resolve a Click event generated by the 111 to allow the user to restore the original data for this Huffman 111 btnLoad Menu Button object. The purpose of this Button object is to allow the user to open a previously created project file. The values stored in the project file will be reloaded into the 111 table to the txtHuffman8 TextBox. 111 /// </summary> 111 application interface. This function will simply call the /// <param name="sender">The sender parameter is a reference to the 111 /// function calling this function. </param> 111 LoadNewProject() function described later in this document. /// <param name="e">The e parameter is for the base class to pass event /// </summary> /// data.</param> /// <param name="sender">The sender parameter is a reference to the private void btnRestoreHuffman8\_Click(object sender, System.EventArgs e) /// function calling this function. </param> /// <param name="e">The e parameter is for the base class to pass event if(lblHuffmanOriginalMarker8.Text != "") /// data.</param> private void btnLoad\_Click(object sender, System.EventArgs e) txtHuffman8.Text = txtHuffmanOriginal8.Text; txtHuffmanOriginal8.Text = ""; LoadNewProject(); lblHuffmanMarker8.Text = lblHuffmanOriginalMarker8.Text; lblHuffmanOriginalMarker8.Text = ""; } /// <summary> /// Pre-conditions: 111 The btnSave Menu Button object has generated a Click event. /// <summary> /// Post-conditions: /// Pre-conditions: 111 This function saves the current values loaded in the Manipulator 111 The btnUpdate Menu Button object has generated a Click event. 111 and any project notes, if included. /// Post-conditions: /// Description: 111 A changed picture has been updated within the application. 111 This method is used to resolve a Click event generated by the /// Description: 111 btnSave Menu Button object. The purpose of this Button object is /// to allow the user to save a project file and all current 111 This method is used to resolve a Click event generated by the 111 btnUpdate Menu Button object. The purpose of this Button object 111 information in the application. The values stored in the project 111 is to allow the user to create a new manipulated image for the 111 file will be reloaded into the application interface. This 111 user to see. 111 function will simply call the SaveNewProject() function described /// </summary> 111 later in this document. /// <param name="sender">The sender parameter is a reference to the /// </summary> /// function calling this function. </param> /// <param name="sender">The sender parameter is a reference to the /// <param name="e">The e parameter is for the base class to pass event /// function calling this function. </param> /// data.</param> /// <param name="e">The e parameter is for the base class to pass event private void btnUpdate\_Click(object sender, System.EventArgs e) /// data.</param> private void btnSave\_Click(object sender, System.EventArgs e) CreateISEImage(); SaveNewProject(); 1 /// <summary> /// Pre-conditions: /// <summarv> 111 The btnNew Menu Button object has generated a Click event. /// Pre-conditions: /// Post-conditions: 111 The btnLoadPicture Menu Button object has generated a Click event. 111 This function clears out all data for pictures. /// Post-conditions: /// Description: 111 An image file has been loaded by the application. 111 This method is used to resolve a Click event generated by the /// Description: 111 btnNew Menu Button object. The purpose of this Button object is 111 This method is used to resolve a Click event generated by the 111 to allow the user to create a new project file that will allow 111 btnLoadPicture Menu Button object. The purpose of this Button 111 them to store picture and note data about different images. 111 object is to allow the user to open an image file. The values stored in the project file will be reloaded into the application /// </summary> 111 /// <param name="sender">The sender parameter is a reference to the 111 interface. This function will simply call the LoadNewProject() /// function calling this function. </param> 111 function described later in this document. /// <param name="e">The e parameter is for the base class to pass event /// </summary> /// data.</param> /// <param name="sender">The sender parameter is a reference to the private void btnNew\_Click(object sender, System.EventArgs e) /// function calling this function. </param> /// <param name="e">The e parameter is for the base class to pass event ClearInterfaceData(); /// data.</param> } private void btnLoadPicture\_Click(object sender, System.EventArgs e) LoadNewPicture(); /// <summary>

frmMain.cs frmMain.cs May 02, 04 2:03 Page 19/186 May 02, 04 2:03 Page 20/186 /// <summary> /// Pre-conditions: 111 The menuPaste menu object has generated a Click event. /// <summary> /// Pre-conditions: /// Post-conditions: 111 The btnUpdatePicture Menu Button object has generated a Click 111 Most recent text on the system clipboard has been pasted to the 111 111 selected TextBox within the Manipulator. event. /// Post-conditions: /// Description: A changed picture has been updated within the application. 111 This method is used to resolve a Click event generated by the 111 /// Description: 111 menuPaste menu object. The purpose of this menu object is to This method is used to resolve a Click event generated by the btnUpdatePicture Button object. The purpose of this Button allow the user to copy the most recent text from the clipboard to a selected Manipulator TextBox. 111 111 111 111 111 object is to allow the user to create a manipulated image based /// </summary> 111 upon the data changed by user. /// <param name="sender">The sender parameter is a reference to the /// </summary> /// function calling this function.</param> /// <param name="sender">The sender parameter is a reference to the /// <param name="e">The e parameter is for the base class to pass event /// function calling this function. </param> /// data.</param> /// <param name="e">The e parameter is for the base class to pass event private void menuPaste\_Click(object sender, System.EventArgs e) /// data.</param> private void btnUpdatePicture\_Click(object sender, System.EventArgs e) SendKeys.Send("^v"); CreateISEImage(); /// <summary> /// Pre-conditions: /// <summary> /// The btnClearHuffman1 button object has generated a Click event. /// Pre-conditions: /// Post-conditions: /// The menuCut menu object has generated a Click event. The corresponding txtHuffman1 text box has been cleared. 111 /// Post-conditions: /// Description: 111 Selected text has been cut from the text box and copied to the 111 This method is used to resolve a Click event generated by the 111 111 btnClearHuffman1 button object. The purpose of this button is to system clipboard. /// Description: 111 allow the user to quickly clear out the corresponding txtHuffman1 111 This method is used to resolve a Click event generated by the 111 text box control. 111 menuCut menu object. The purpose of this menu object is to allow /// </summary> 111 the user to cut selected text from any TextBox field within the /// <param name="sender">The sender parameter is a reference to the 111 Manipulator. The cut text is copied to the system clipboard for /// function calling this function.</param> 111 /// <param name="e">The e parameter is for the base class to pass event future retrieval. /// </summary> /// data.</param> /// <param name="sender">The sender parameter is a reference to the private void btnClearHuffman1\_Click(object sender, System.EventArgs e) /// function calling this function.</param> /// <param name="e">The e parameter is for the base class to pass event if(txtHuffmanOriginal1.Text.Trim() == "") /// data.</param> private void menuCopy\_Click(object sender, System.EventArgs e) txtHuffmanOriginal1.Text = txtHuffman1.Text; lblHuffmanOriginalMarker1.Text = lblHuffmanMarker1.Text; SendKeys.Send("^c"); txtHuffman1.Text = ""; /// <summary> /// Pre-conditions: 111 The menuCopy menu object has generated a Click event. /// <summary> /// Post-conditions: /// Pre-conditions: 111 Selected text has been copied to the system clipboard. /// The btnClearHuffman2 button object has generated a Click event. /// Description: /// Post-conditions: 111 This method is used to resolve a Click event generated by the 111 The corresponding txtHuffman2 text box has been cleared. 111 menuCopy menu object. The purpose of this menu object is to /// Description: 111 allow the user to copy selected text from any TextBox field 111 This method is used to resolve a Click event generated by the 111 within the Manipulator. The text is copied to the system 111 btnClearHuffman2 button object. The purpose of this button is to 111 clipboard for future retrieval. 111 allow the user to quickly clear out the corresponding txtHuffman2 /// </summary> 111 text box control. /// <param name="sender">The sender parameter is a reference to the /// </summarv> /// function calling this function.</param> /// <param name="sender">The sender parameter is a reference to the /// <param name="e">The e parameter is for the base class to pass event /// function calling this function.</param> /// data.</param> /// <param name="e">The e parameter is for the base class to pass event private void menuCut\_Click(object sender, System.EventArgs e) /// data.</param> private void btnClearHuffman2\_Click(object sender, System.EventArgs e) SendKeys.Send("^x"); } if(txtHuffmanOriginal2.Text.Trim() == "") txtHuffmanOriginal2.Text = txtHuffman2.Text;

```
frmMain.cs
                                                                                                                      frmMain.cs
May 02, 04 2:03
                                                                  Page 21/186
                                                                                    May 02, 04 2:03
                                                                                                                                                      Page 22/186
      lblHuffmanOriginalMarker2.Text = lblHuffmanMarker2.Text;
                                                                                       111
                                                                                             allow the user to quickly clear out the corresponding txtHuffman5
                                                                                      111
                                                                                             text box control.
    }
                                                                                      /// </summary>
    txtHuffman2.Text = "";
                                                                                      /// <param name="sender">The sender parameter is a reference to the
  }
                                                                                      /// function calling this function.</param>
                                                                                      /// <param name="e">The e parameter is for the base class to pass event
                                                                                       /// data.</param>
                                                                                      private void btnClearHuffman5_Click(object sender, System.EventArgs e)
  /// <summary>
  /// Pre-conditions:
      The btnClearHuffman3 button object has generated a Click event.
                                                                                         if(txtHuffmanOriginal5.Text.Trim() == "")
  111
  /// Post-conditions:
  /// The corresponding txtHuffman3 text box has been cleared.
                                                                                          txtHuffmanOriginal5.Text = txtHuffman5.Text;
  /// Description:
                                                                                          lblHuffmanOriginalMarker5.Text = lblHuffmanMarker5.Text;
        This method is used to resolve a Click event generated by the
  111
  111
         btnClearHuffman3 button object. The purpose of this button is to
  111
         allow the user to quickly clear out the corresponding txtHuffman3
                                                                                        txtHuffman5.Text = "";
  111
         text box control.
  /// </summary>
  /// <param name="sender">The sender parameter is a reference to the
                                                                                      /// <summary>
  /// function calling this function.</param>
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                      /// Pre-conditions:
  /// data.</param>
                                                                                      111
                                                                                           The btnClearHuffman6 button object has generated a Click event.
  private void btnClearHuffman3_Click(object sender, System.EventArgs e)
                                                                                      /// Post-conditions:
                                                                                      /// The corresponding txtHuffman6 text box has been cleared.
    if(txtHuffmanOriginal3.Text.Trim() == "")
                                                                                      /// Description:
                                                                                      111
                                                                                             This method is used to resolve a Click event generated by the
      txtHuffmanOriginal3.Text = txtHuffman3.Text;
                                                                                      111
                                                                                             btnClearHuffman6 button object. The purpose of this button is to
      lblHuffmanOriginalMarker3.Text = lblHuffmanMarker3.Text;
                                                                                      111
                                                                                             allow the user to quickly clear out the corresponding txtHuffman6
                                                                                      111
                                                                                             text box control.
                                                                                      /// </summary>
    txtHuffman3.Text = "";
                                                                                      /// <param name="sender">The sender parameter is a reference to the
                                                                                      /// function calling this function.</param>
                                                                                      /// <param name="e">The e parameter is for the base class to pass event
                                                                                      /// data.</param>
  /// <summary>
                                                                                      private void btnClearHuffman6_Click(object sender, System.EventArgs e)
  /// Pre-conditions:
  /// The btnClearHuffman4 button object has generated a Click event.
                                                                                        if(txtHuffmanOriginal6.Text.Trim() == "")
  /// Post-conditions:
        The corresponding txtHuffman4 text box has been cleared.
                                                                                          txtHuffmanOriginal6.Text = txtHuffman6.Text;
  111
  /// Description:
                                                                                          lblHuffmanOriginalMarker6.Text = lblHuffmanMarker6.Text;
  111
        This method is used to resolve a Click event generated by the
        btnClearHuffman4 button object. The purpose of this button is to
  111
  111
       allow the user to quickly clear out the corresponding txtHuffman4
                                                                                        txtHuffman6.Text = "";
  111
      text box control.
                                                                                      }
  /// </summary>
  /// <param name="sender">The sender parameter is a reference to the
  /// function calling this function.</param>
                                                                                       /// <summarv>
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                      /// Pre-conditions:
  /// data.</param>
                                                                                       /// The btnClearHuffman7 button object has generated a Click event.
  private void btnClearHuffman4_Click(object sender, System.EventArgs e)
                                                                                       /// Post-conditions:
                                                                                      111
                                                                                            The corresponding txtHuffman7 text box has been cleared.
    if(txtHuffmanOriginal4.Text.Trim() == "")
                                                                                      /// Description:
                                                                                      111
                                                                                            This method is used to resolve a Click event generated by the
                                                                                           btnClearHuffman7 button object. The purpose of this button is to
      txtHuffmanOriginal4.Text = txtHuffman4.Text;
                                                                                      111
      lblHuffmanOriginalMarker4.Text = lblHuffmanMarker4.Text;
                                                                                      111
                                                                                            allow the user to quickly clear out the corresponding txtHuffman7
                                                                                       111
                                                                                            text box control.
                                                                                      /// </summarv>
    txtHuffman4.Text = "";
                                                                                      /// <param name="sender">The sender parameter is a reference to the
  }
                                                                                       /// function calling this function.</param>
                                                                                      /// <param name="e">The e parameter is for the base class to pass event
                                                                                       /// data.</param>
  /// <summary>
                                                                                      private void btnClearHuffman7_Click(object sender, System.EventArgs e)
  /// Pre-conditions:
  111
        The btnClearHuffman5 button object has generated a Click event.
                                                                                        if(txtHuffmanOriginal7.Text.Trim() == "")
  /// Post-conditions:
  111
        The corresponding txtHuffman5 text box has been cleared.
                                                                                          txtHuffmanOriginal7.Text = txtHuffman7.Text;
  /// Description:
                                                                                          lblHuffmanOriginalMarker7.Text = lblHuffmanMarker7.Text;
  111
         This method is used to resolve a Click event generated by the
  111
         btnClearHuffman5 button object. The purpose of this button is to
```

May 02, 04 2:03	frmMain.cs	Page 23/186	May 02, 04 2:03	frmMain.cs	Page 24/186
<pre>May 02, 04 2:03     txtHuffman7.Text = ""; } /// <summary> /// Pre-conditions: /// The btnClearHuffma /// Description: /// This method is use /// Description: /// This method is use /// allow the user to /// text box control. /// </summary> /// <param name="sender"/> /// function calling this /// <param name="sender"/> /// function calling this /// <param name="sender"/> /// function calling this /// <param name="sender"/> /// data. private void btnClearHuffman8. {     txtHuffmanOriginal8.     txtHuffmanOriginal8.T     lbHuffmanOriginalMar }     txtHuffman8.Text = ""; }</pre>	<pre>frmMain.cs n8 button object has generate txtHuffman8 text box has beer d to resolve a Click event ge utton object. The purpose of quickly clear out the corresp The sender parameter is a ref function. parameter is for the base cl man8_Click(object sender, Sys Text.Trim() == "") ext = txtHuffman8.Text; ker8.Text = lblHuffmanMarker8</pre>	Page 23/186 ed a Click event. In cleared. Enerated by the Entries button is to bonding txtHuffman8 Enerence to the Lass to pass event stem.EventArgs e) 3.Text;	<pre>May 02, 04 2:03 /// <param e"="" name="send /// function calling /// &lt;param name="/>' /// data. private void btnClear {     if(txtQuantizerOrig)     lblQuantizerOrig)     lblQuantizerOrig)     txtQuantizerOrig)     // Pre-conditions:     /// Pre-conditions:     /// Pre-conditions:     /// The btnClearQuant     /// Description:     /// This method i:     /// to allow the y     /// to allow the y     /// <param name="send     /// &lt;param name=" send="" summary=""/>     /// <param name="e"/>'     /// data.     /// callow the y     /// caram name="e"&gt;'     /// data.     /// data. </pre>	<pre>frmMain.cs der"&gt;The sender parameter is a refet this function. The e parameter is for the base clu rQuantizer2_Click(object sender, Sy ginal2.Text.Trim() == "") inal2.Text = txtQuantizer2.Text; inalMarker2.Text = lblQuantizerMark = ""; uantizer3 button object has generated ding txtQuantizer3 text box has been s used to resolve a Click event generated izer3 button object. The purpose of user to quickly clear out the correct text box control. der"&gt;The sender parameter is a refet this function. The e parameter is for the base clue counting a click is the clue text box control.</pre>	Page 24/186 erence to the ass to pass event ystem.EventArgs e) ker2.Text; ted a Click event. en cleared. nerated by the of this button is esponding erence to the ass to pass event
<pre>/// <summary> /// Pre-conditions: /// The btnClearQuanti /// Post-conditions: /// Description: /// This method is use /// btnClearQuantizer1 /// to allow the user /// txtQuantizer1 text /// </summary> /// <param name="sender"/> /// function calling this /// data. </pre>	zerl button object has genera txtQuantizerl text box has be d to resolve a Click event ge button object. The purpose to quickly clear out the corr box control. The sender parameter is a ref function. parameter is for the base cl	ented a Click event. en cleared. enerated by the of this button is responding ference to the lass to pass event	<pre>private void btnClea: { if(txtQuantizerOrig txtQuantizerOrig lblQuantizerOrig } txtQuantizer3.Text } /// <summary> /// Pre-conditions: /// The btnClearQuartic /// The btnClearQuartic // The bt</summary></pre>	<pre>rQuantizer3_Click(object sender, Sy ginal3.Text.Trim() == "") inal3.Text = txtQuantizer3.Text; inalMarker3.Text = lblQuantizerMar; = ""; uantizer4 button object has genera;</pre>	ystem.EventArgs e) ker3.Text; ted a Click event.
<pre>private void btnClearQuan {     if(txtQuantizerOriginal     {         txtQuantizerOriginal1         lblQuantizerOriginalM     } }</pre>	<pre>tizer1_Click(object sender, \$ 1.Text.Trim()== "") .Text = txtQuantizer1.Text; arker1.Text = lblQuantizerMar</pre>	<pre>;ystem.EventArgs e) cker1.Text;</pre>	<pre>/// Post-conditions: /// The correspond /// Description: /// This method i: /// btnClearQuant /// to allow the y /// txtQuantizer4 /// </pre>	ding txtQuantizer4 text box has been sused to resolve a Click event genizer4 button object. The purpose suser to quickly clear out the correctext box control.	en cleared. nerated by the of this button is esponding
<pre>txtQuantizer1.Text = "" } /// <summary> /// Pre-conditions: /// The btnClearQuanti /// Post-conditions: /// The corresponding /// Description: /// This method is use /// btnClearQuantizer2 /// the oblew the way.</summary></pre>	; zer2 button object has genera txtQuantizer2 text box has be d to resolve a Click event ge button object. The purpose	ated a Click event. een cleared. enerated by the of this button is	<pre>/// <param "send<br="" name="send&lt;br&gt;/// function calling&lt;br&gt;/// &lt;param name="/>/// data. private void btnCleas { if(txtQuantizerOrig</pre>	<pre>der"&gt;The sender parameter is a refet this function. The e parameter is for the base clue rQuantizer4_Click(object sender, Syn ginal4.Text.Trim() == "") inal4.Text = txtQuantizer4.Text; inalMarker4.Text = lblQuantizerMart = "".</pre>	erence to the ass to pass event ystem.EventArgs e) ker4.Text;
<pre>/// to allow the user /// txtQuantizer2 text /// </pre>	to quickly clear out the corr box control.	esponding	<pre>txtQuantizer4.Text }</pre>	= "";	

May 02, 04 2:03	frmMain.cs	Page 25/186	May 02, 04 2:03	frmMain.cs	Page 26/186
/// <summary> /// Pre-conditions: /// The btnAddRand /// event.</summary>	domHuffmanl button object has gene:	rated a Click	<pre>t = RandomNumber.1 a += Convert(t).To txtHuffman2.Text - }</pre>	Next(16); pString() + " "; = a;	
<pre>/// Post-conditions: /// The correspond /// concatenated t /// the control. /// Description: /// This method is /// btnAddRandomHu /// is to allow th /// of the existin /// be represent t /// addition this /// of data. ///  ///  ///  ///  ///  ///  private void btnAddRa { int t = RandomNumbe string a = Convert( if(txtHuffmanOrigina lblHuffmanOrigina } </pre>	<pre>ling txtHuffman1 text box has a ran to the end of any text that was all s used to resolve a Click event gen iffman1 button object. The purpose the user to simulate adding a random ing text in the txtHuffman1 control the hexadecimal value of one byte of method will also add a space (" " der"&gt;The sender parameter is a refer this function. The e parameter is for the base cla undomHuffman1_Click(object sender, er.Next(16); t).ToString(); hall.Text = txtHuffman1.Text; uMarker1.Text = lblHuffmanMarker1</pre>	ndom byte ready existing in nerated by the e of this button m byte to the end . This data will of data. In ) after the byte erence to the ass to pass event System.EventArgs e) .Text;	<pre>/// <summary> /// Pre-conditions: /// The btnAddRar /// event. /// Post-conditions: /// The correspon /// concatenated /// the control. /// Description: /// This method : /// btnAddRandom /// to allow the /// the existing /// represent the /// this method v /// </summary> /// <param data.<="" e":="" name="ser /// function calling /// cparam name=" param=""/> private void btnAddI {     int t = RandomNumM     string a = Convert } /// Summary /// Summary </pre>	ndomHuffman3 button object has gener inding txtHuffman3 text box has a ran to the end of any text that was alr lis used to resolve a Click event gen Huffman3 button object. The purpose user to simulate adding a random by text in the txtHuffman3 control. The e hexadecimal value of one byte of d will also add a space (" ") after the nder">The sender parameter is a refe g this function. The e parameter is for the base cla RandomHuffman3_Click(object sender, ber.Next(16); t(t).ToString();	ated a Click dom byte eady existing in erated by the of this button is te to the end of is data will be ata. In addition e byte of data. rence to the ss to pass event System.EventArgs e)
<pre>t = RandomNumber.Ne a += Convert(t).ToS txtHuffmanl.Text += }</pre>	<pre>sxt(16); string() + " "; a;</pre>		if(txtHuffmanOrig: { txtHuffmanOrigin ] }	inal3.Text == "") nal3.Text = txtHuffman3.Text; nalMarker3.Text = lblHuffmanMarker3.	Text;
/// <summary> /// Pre-conditions: /// The btnAddRand /// event.</summary>	lomHuffman2 button object has gene:	rated a Click	<pre>t = RandomNumber.1 a += Convert(t).To txtHuffman3.Text - }</pre>	Next(16); OString() + " "; += a;	
<pre>/// Post-conditions: /// The correspond /// concatenated t /// bescription: /// Description: /// Dismethod is /// btnAddRandomHu /// to allow the u /// the existing t /// cyaram name="send /// function calling /// cyaram name="send /// function calling /// cyaram name="e"&gt;T /// data. private void btnAddRa { int t = RandomNumbe string a = Convert( if(txtHuffmanOrigin //</pre>	<pre>ling txtHuffman2 text box has a ran to the end of any text that was all so the end of any text that was all iffman2 button object. The purpose iser to simulate adding a random by text in the txtHuffman2 control. The hexadecimal value of one byte of of all also add a space (" ") after the er"&gt;The sender parameter is a refer this function. The e parameter is for the base claim undomHuffman2_Click(object sender, er.Next(16); t).ToString(); mal2.Text == "")</pre>	ndom byte ready existing in nerated by the e of this button is yte to the end of his data will be data. In addition he byte of data. erence to the ass to pass event System.EventArgs e)	<pre>/// <summary> /// Pre-conditions: /// The btnAddRam /// event. /// Post-conditions: /// The correspon /// concatenated /// the control. /// Description: /// This method : /// btnAddRandom /// to allow the /// the existing /// this method w /// cyaram name="sem /// function calling /// sparam name="e" /// data. private void btnAddRaddm ///</summary></pre>	ndomHuffman4 button object has gener inding txtHuffman4 text box has a ran to the end of any text that was alr is used to resolve a Click event gen Huffman4 button object. The purpose user to simulate adding a random by text in the txtHuffman4 control. The e hexadecimal value of one byte of d will also add a space (" ") after the nder">The sender parameter is a refe g this function. >The e parameter is for the base cla RandomHuffman4_Click(object sender,	ated a Click dom byte eady existing in erated by the of this button is te to the end of is data will be ata. In addition e byte of data. rence to the ss to pass event System.EventArgs e)
{ txtHuffmanOrigina lblHuffmanOrigina }	ul2.Text = txtHuffman2.Text; ulMarker2.Text = lblHuffmanMarker2	.Text;	<pre>int t = RandomNuml string a = Convert</pre>	<pre>per.Next(16); c(t).ToString();</pre>	

```
frmMain.cs
                                                                                                                      frmMain.cs
                                                                  Page 27/186
May 02, 04 2:03
                                                                                    May 02, 04 2:03
                                                                                                                                                      Page 28/186
    if(txtHuffmanOriginal4.Text == "")
                                                                                       private void btnAddRandomHuffman6 Click (object sender, System.EventArgs e)
      txtHuffmanOriginal4.Text = txtHuffman4.Text;
                                                                                         int t = RandomNumber.Next(16);
      lblHuffmanOriginalMarker4.Text = lblHuffmanMarker4.Text;
                                                                                         string a = Convert(t).ToString();
                                                                                         if(txtHuffmanOriginal6.Text == "")
    t = RandomNumber.Next(16);
    a += Convert(t).ToString() + " ";
                                                                                           txtHuffmanOriginal6.Text = txtHuffman6.Text;
    txtHuffman4.Text += a;
                                                                                          lblHuffmanOriginalMarker6.Text = lblHuffmanMarker6.Text;
                                                                                        t = RandomNumber.Next(16);
  /// <summary>
                                                                                        a += Convert(t).ToString() + " ";
  /// Pre-conditions:
                                                                                        txtHuffman6.Text += a;
  111
         The btnAddRandomHuffman5 button object has generated a Click
  111
         event.
  /// Post-conditions:
  111
        The corresponding txtHuffman5 text box has a random byte
                                                                                       /// <summary>
  111
         concatenated to the end of any text that was already existing in
                                                                                       /// Pre-conditions:
                                                                                             The btnAddRandomHuffman7 button object has generated a Click
  111
         the control.
                                                                                       111
  /// Description:
                                                                                       111
                                                                                             event.
  111
         This method is used to resolve a Click event generated by the
                                                                                       /// Post-conditions:
  111
         btnAddRandomHuffman5 button object. The purpose of this button is
                                                                                             The corresponding txtHuffman7 text box has a random byte
                                                                                       111
  111
        to allow the user to simulate adding a random byte to the end of
                                                                                       111
                                                                                             concatenated to the end of any text that was already existing in
  111
        the existing text in the txtHuffman5 control. This data will be
                                                                                       111
                                                                                             the control.
                                                                                       /// Description:
  111
        represent the hexadecimal value of one byte of data. In addition
  111
         this method will also add a space (" ") after the byte of data.
                                                                                       ///
                                                                                             This method is used to resolve a Click event generated by the
  /// </summary>
                                                                                       111
                                                                                             btnAddRandomHuffman7 button object. The purpose of this button is
  /// <param name="sender">The sender parameter is a reference to the
                                                                                       111
                                                                                             to allow the user to simulate adding a random byte to the end of
  /// function calling this function.</param>
                                                                                       111
                                                                                            the existing text in the txtHuffman7 control. This data will be
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                       111
                                                                                            represent the hexadecimal value of one byte of data. In addition
  /// data.</param>
                                                                                       111
                                                                                           this method will also add a space (" ") after the byte of data.
  private void btnAddRandomHuffman5_Click(object sender, System.EventArgs e)
                                                                                       /// </summary>
                                                                                       /// <param name="sender">The sender parameter is a reference to the
    int t = RandomNumber.Next(16);
                                                                                       /// function calling this function.</param>
                                                                                       /// <param name="e">The e parameter is for the base class to pass event
    string a = Convert(t).ToString();
                                                                                       /// data.</param>
    if(txtHuffmanOriginal5.Text == "")
                                                                                       private void btnAddRandomHuffman7_Click(object sender, System.EventArgs e)
      txtHuffmanOriginal5.Text = txtHuffman5.Text;
                                                                                         int t = RandomNumber.Next(16);
      lblHuffmanOriginalMarker5.Text = lblHuffmanMarker5.Text;
                                                                                        string a = Convert(t).ToString();
    }
                                                                                         if(txtHuffmanOriginal7.Text == "")
    t = RandomNumber.Next(16);
    a += Convert(t).ToString() + " ";
                                                                                           txtHuffmanOriginal7.Text = txtHuffman7.Text;
    txtHuffman5.Text += a;
                                                                                          lblHuffmanOriginalMarker7.Text = lblHuffmanMarker7.Text;
                                                                                        t = RandomNumber.Next(16);
  /// <summary>
                                                                                        a += Convert(t).ToString() + " ";
  /// Pre-conditions:
                                                                                        txtHuffman7.Text += a;
  111
         The btnAddRandomHuffman6 button object has generated a Click
  111
         event.
  /// Post-conditions:
  111
        The corresponding txtHuffman6 text box has a random byte
                                                                                       /// <summarv>
  111
         concatenated to the end of any text that was already existing in
                                                                                       /// Pre-conditions:
  111
        the control.
                                                                                       111
                                                                                             The btnAddRandomHuffman8 button object has generated a Click
  /// Description:
                                                                                       111
                                                                                             event.
  111
        This method is used to resolve a Click event generated by the
                                                                                       /// Post-conditions:
        btnAddRandomHuffman6 button object. The purpose of this button is
  111
                                                                                       111
                                                                                            The corresponding txtHuffman8 text box has a random byte
  111
        to allow the user to simulate adding a random byte to the end of
                                                                                       111
                                                                                             concatenated to the end of any text that was already existing in
  111
        the existing text in the txtHuffman6 control. This data will be
                                                                                       111
                                                                                             the control.
  111
         represent the hexadecimal value of one byte of data. In addition
                                                                                       /// Description:
         this method will also add a space (" ") after the byte of data.
  111
                                                                                       111
                                                                                             This method is used to resolve a Click event generated by the
  /// </summary>
                                                                                       111
                                                                                             btnAddRandomHuffman8 button object. The purpose of this button is
  /// <param name="sender">The sender parameter is a reference to the
                                                                                       111
                                                                                             to allow the user to simulate adding a random byte to the end of
  /// function calling this function.</param>
                                                                                       111
                                                                                             the existing text in the txtHuffman8 control. This data will be
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                       111
                                                                                             represent the hexadecimal value of one byte of data. In addition
  /// data.</param>
                                                                                       111
                                                                                             this method will also add a space (" ") after the byte of data.
```

```
frmMain.cs
                                                                                                                       frmMain.cs
May 02, 04 2:03
                                                                                     May 02, 04 2:03
                                                                   Page 29/186
                                                                                                                                                         Page 30/186
  /// </summary>
                                                                                        111
                                                                                               This method is used to resolve a Click event generated by the
  /// <param name="sender">The sender parameter is a reference to the
                                                                                               btnAddRandomQuantizer2 button object. The purpose of this button
                                                                                        ///
  /// function calling this function.</param>
                                                                                        111
                                                                                               is to allow the user to simulate adding a random byte to the end
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                        111
                                                                                               of the existing text in the txtQuantizer2 control. This data will
  /// data.</param>
                                                                                        111
                                                                                             be represent the hexadecimal value of one byte of data. In
  private void btnAddRandomHuffman8_Click(object sender, System.EventArgs e)
                                                                                        111
                                                                                             addition this method will also add a space (" ") after the byte
                                                                                        111
                                                                                               of data.
                                                                                        /// </summary>
    int t = RandomNumber.Next(16);
    string a = Convert(t).ToString();
                                                                                        /// <param name="sender">The sender parameter is a reference to the
                                                                                        /// function calling this function.</param>
                                                                                        /// <param name="e">The e parameter is for the base class to pass event
    if(txtHuffmanOriginal8.Text == "")
                                                                                        /// data.</param>
      txtHuffmanOriginal8.Text = txtHuffman8.Text;
                                                                                        private void btnAddRandomQuantizer2_Click(object sender,
      lblHuffmanOriginalMarker8.Text = lblHuffmanMarker8.Text;
                                                                                                    System.EventArgs e)
                                                                                          int t = RandomNumber.Next(16);
    t = RandomNumber.Next(16);
                                                                                          string a = Convert(t).ToString();
    a += Convert(t).ToString() + " ";
    txtHuffman8.Text += a;
                                                                                          if(txtQuantizerOriginal2.Text == "")
                                                                                            txtQuantizerOriginal2.Text = txtQuantizer2.Text;
                                                                                            lblQuantizerOriginalMarker2.Text = lblQuantizerMarker2.Text;
  /// <summary>
  /// Pre-conditions:
  /// The btnAddRandomQuantizer1 button object has generated a Click
                                                                                          t = RandomNumber.Next(16);
                                                                                          a += Convert(t).ToString() + " ";
  111
         event.
  /// Post-conditions:
                                                                                          txtQuantizer2.Text += a;
  111
        The corresponding txtQuantizer1 text box has a random byte
  111
         concatenated to the end of any text that was already existing in
  111
        the control.
  /// Description:
                                                                                        /// <summary>
  111
        This method is used to resolve a Click event generated by the
                                                                                        /// Pre-conditions:
         btnAddRandomQuantizer1 button object. The purpose of this button
                                                                                               The btnAddRandomQuantizer3 button object has generated a Click
  111
                                                                                        111
  111
         is to allow the user to simulate adding a random byte to the end
                                                                                        111
                                                                                               event.
  111
         of the existing text in the txtQuantizer1 control. This data will
                                                                                        /// Post-conditions:
  111
         be represent the hexadecimal value of one byte of data. In
                                                                                        111
                                                                                             The corresponding txtQuantizer3 text box has a random byte
  111
         addition this method will also add a space (" ") after the byte
                                                                                        111
                                                                                              concatenated to the end of any text that was already existing in
  111
         of data.
                                                                                        111
                                                                                               the control.
  /// </summary>
                                                                                        /// Description:
  /// <param name="sender">The sender parameter is a reference to the
                                                                                        111
                                                                                              This method is used to resolve a Click event generated by the btnAddRandomQuantizer3 button object. The purpose of this button
  /// function calling this function.</param>
                                                                                        111
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                        111
                                                                                              is to allow the user to simulate adding a random byte to the end
  /// data.</param>
                                                                                        111
                                                                                              of the existing text in the txtOuantizer3 control. This data will
  private void btnAddRandomQuantizer1_Click(object sender, System.EventArgs e)
                                                                                        111
                                                                                              be represent the hexadecimal value of one byte of data. In
                                                                                        111
                                                                                              addition this method will also add a space (" ") after the byte
    int t = RandomNumber.Next(16);
                                                                                        111
                                                                                              of data.
    string a = Convert(t).ToString();
                                                                                        /// </summarv>
                                                                                        /// <param name="sender">The sender parameter is a reference to the
    if(txtQuantizerOriginal1.Text == "")
                                                                                        /// function calling this function. </param>
                                                                                        /// <param name="e">The e parameter is for the base class to pass event
      txtQuantizerOriginal1.Text = txtQuantizer1.Text;
                                                                                        /// data.</param>
      lblQuantizerOriginalMarker1.Text = lblQuantizerMarker1.Text;
                                                                                        private void btnAddRandomQuantizer3_Click(object sender,
                                                                                                    System.EventArgs e)
    t = RandomNumber.Next(16);
                                                                                          int t = RandomNumber.Next(16);
    a += Convert(t).ToString() + " ";
                                                                                          string a = Convert(t).ToString();
    txtQuantizer1.Text += a;
  }
                                                                                          if(txtQuantizerOriginal3.Text == "")
                                                                                            txtQuantizerOriginal3.Text = txtQuantizer3.Text;
  /// <summary>
                                                                                            lblQuantizerOriginalMarker3.Text = lblQuantizerMarker3.Text;
  /// Pre-conditions:
  111
         The btnAddRandomQuantizer2 button object has generated a Click
  111
         event.
                                                                                          t = RandomNumber.Next(16);
  /// Post-conditions:
                                                                                          a += Convert(t).ToString() + " ";
  111
         The corresponding txtQuantizer2 text box has a random byte
                                                                                          txtQuantizer3.Text += a;
  111
         concatenated to the end of any text that was already existing in
  111
        the control.
  /// Description:
```

May 02, 04 2:03	frmMain.cs	Page 31/186 M	ay 02, 04 2:03	frmMain.cs	Page 32/186
<pre>/// <summary> /// Pre-conditions: /// The btnAddRand /// event. /// Post-conditions: /// The correspond /// concatenated t /// the control. /// Description: /// Description: /// Description: /// btnAddRandomQu /// is to allow th /// of the existin /// of the existin /// of the existin /// of data. ///  private void btnAddRa System.Ev {     int t = RandomNumbe     string a = Convert(     if(txtQuantizerOrigi     {         txtQuantizerOrigi         blQuantizerOrigi     } } /// Conditions // Conditions /// Conditions</summary></pre>	<pre>omQuantizer4 button object has get ing txtQuantizer4 text box has a ; o the end of any text that was al: used to resolve a Click event get antizer4 button object. The purp e user to simulate adding a randon g text in the txtQuantizer4 contron nt the hexadecimal value of one by his method will also add a space er"&gt;The sender parameter is a refet this function. he e parameter is for the base cl. ndomQuantizer4_Click(object sende: entArgs e) r.Next(16); t).ToString(); final4.Text == txtQuantizer4.Text; nalMarker4.Text = lblQuantizerMariet ind set in the text is the text in the text is the</pre>	herated a Click random byte ready existing in herated by the ose of this button n byte to the end ol. This data will yte of data. In (" ") after the byte erence to the ass to pass event r, ker4.Text;	<pre>/// menuLargeOrigir /// to allow the us /// modes for the of /// summary&gt; /// sparam name="sended /// function calling t /// sparam name="e"&gt;Th /// data.s/param&gt; private void menuLarge { if (PicOriginalStretch menuLargeOriginal. picOriginal.SizeMo if (menuSmallAnipp menuLargeOriginal. picOriginalStretch menuLargeOriginal. picOriginalStretch menuLargeOriginal. if (menuSmallManipp menuLargeManipul { menuAll.Checked } picOriginal.Update() } </pre>	<pre>nal Menu object. The purpose of t ser to toggle between "normal" and priginal picture on the tabOrigina er"&gt;The sender parameter is a refe this function. ne e parameter is for the base cla eOriginal_Click(object sender, Sys ched) ned = false; .Checked = false; ode = PictureBoxSizeMode.Normal; false; ned = true; .Checked = true; ode = PictureBoxSizeMode.StretchIm ulated.Checked == true &amp;&amp; al.Checked == true &amp;&amp; lated.Checked == true) = true; );</pre>	his Menu object is "stretch" size 1 Tab control. rence to the ss to pass event tem.EventArgs e)
<pre> } t = RandomNumber.Ne a += Convert(t).ToS txtQuantizer4.Text } /// Pre-conditions: /// Pre-conditions: /// Post-conditions: /// A changed pict /// Description: /// This method is /// menuUpdate Men /// allow the user /// changed by use ///  /// <param e"="" name="send /// function calling /// &lt;param name="/>T /// data. private void menuUpdat { CreateISEImage(); } </pre>	<pre>xt(16); tring() + " "; += a;</pre>	<pre>&lt; event. pplication. herated by the nu object is to sed upon the data erence to the ass to pass event entArgs e)</pre>	<pre>/// <summary> /// Pre-conditions: /// The menuLargeMa /// Post-conditions: /// If the picture /// will be changed /// switched to "nd /// Description: /// This method is /// This method is /// to allow the us /// to allow the us /// to allow the us /// for the changed ///  private void menuLarge {     if (PicManipulatedStre         menuLargeManipulated.Siz         picManipulated.Siz         picManipulated.Siz</summary></pre>	<pre>anipulated Menu object has generat in the picManipulated is in "norm i to "stretch" size mode, otherwis ormal" size mode. used to resolve a Click event gen lated Menu object. The purpose of ser to toggle between "normal" and i picture on the tabManipulated Ta er"&gt;The sender parameter is a refe this function. ne e parameter is for the base cla eManipulated_Click(object sender, retched) etched = false; ted.Checked = false; zeMode = PictureBoxSizeMode.Normal</pre>	<pre>ed a Click event. al" size mode, it e it will be .erated by the of this Menu object is i "stretch" size .b control. erence to the uss to pass event System.EventArgs e) ;;</pre>
<pre>/// <summary> /// Pre-conditions: /// The menuLarge0 /// Post-conditions: /// If the picture /// will be change /// switched to "n /// Description: /// This method is</summary></pre>	riginal Menu object has generated in the picOriginal is in "normal d to "stretch" size mode, otherwi ormal" size mode. used to resolve a Click event ge	a Click event. " size mode, it se it will be nerated by the	<pre>menuAll.Checked = } else {     PicManipulatedStre     picManipulated.Siz     if (menuSmallManipu         menuLargeOrigina         {         {         }         }</pre>	<pre>false; etched = true; ced.Checked = true; zeMode = PictureBoxSizeMode.Stretc ulated.Checked == true &amp;&amp; al.Checked == true &amp;&amp; al.Checked == true)</pre>	hImage;

```
frmMain.cs
                                                                                                                        frmMain.cs
May 02, 04 2:03
                                                                   Page 33/186
                                                                                     May 02, 04 2:03
                                                                                                                                                         Page 34/186
         menuAll.Checked = true;
                                                                                          if (PicManipulatedSmallStretched)
    picManipulated.Update();
                                                                                            PicManipulatedSmallStretched = false;
                                                                                            menuSmallManipulated.Checked = false;
                                                                                            picManipulatedSmall.SizeMode = PictureBoxSizeMode.Normal;
                                                                                            menuAll.Checked = false;
  /// <summary>
  /// Pre-conditions:
                                                                                          else
         The menuSmallOriginal Menu object has generated a Click event.
  111
  /// Post-conditions:
                                                                                            PicManipulatedSmallStretched = true;
  111
       If the picture in the picOriginalSmall is in "normal" size mode,
                                                                                            menuSmallManipulated.Checked = true;
  111
       it will be changed to "stretch" size mode, otherwise it will be
                                                                                            picManipulatedSmall.SizeMode = PictureBoxSizeMode.StretchImage;
         switched to "normal" size mode.
  111
                                                                                            if(menuSmallOriginal.Checked == true &&
  /// Description:
                                                                                              menuLargeManipulated.Checked == true &&
  111
         This method is used to resolve a Click event generated by the
                                                                                              menuLargeOriginal.Checked == true)
  111
         menuSmallOriginal Menu object. The purpose of this Menu object is
  111
         to allow the user to toggle between "normal" and "stretch" size
                                                                                              menuAll.Checked = true;
         modes for the original picture on the tabConsole Tab control.
  111
  /// </summary>
  /// <param name="sender">The sender parameter is a reference to the
                                                                                          picManipulatedSmall.Update();
  /// function calling this function.</param>
/// cparam name="e">The e parameter is for the base class to pass event
  /// data.</param>
  private void menuSmallOriginal_Click(object sender, System.EventArgs e)
                                                                                        /// <summary>
                                                                                        /// Pre-conditions:
    if (PicOriginalSmallStretched)
                                                                                               The menuAll Menu object has generated a Click event.
                                                                                        111
                                                                                        111
                                                                                            Post-conditions:
      PicOriginalSmallStretched = false;
                                                                                        111
                                                                                               The menuAll Menu control will become selected and all pictures
      menuSmallOriginal.Checked = false;
                                                                                        111
                                                                                               will be switched to "stretch" size mode. If this menu has been
      picOriginalSmall.SizeMode = PictureBoxSizeMode.Normal;
                                                                                        111
                                                                                              previously selected, all of the pictures will be switched to
      menuAll.Checked = false;
                                                                                        111
                                                                                               "normal" size mode instead.
                                                                                        /// Description:
                                                                                        111
                                                                                               This method is used to resolve a Click event generated by the
    else
                                                                                        111
                                                                                               menuAll Menu object. The purpose of this Menu object is to
                                                                                        111
                                                                                               allow the user to toggle between "normal" and "stretch" size
      PicOriginalSmallStretched = true;
      menuSmallOriginal.Checked = true;
                                                                                        111
                                                                                               modes for the all of the pictures on the all of the Tab control.
      picOriginalSmall.SizeMode = PictureBoxSizeMode.StretchImage;
                                                                                        /// </summary>
       if(menuSmallManipulated.Checked == true &&
                                                                                        /// <param name="sender">The sender parameter is a reference to the
         menuLargeManipulated.Checked == true &&
                                                                                        /// function calling this function.</param>
         menuLargeOriginal.Checked == true)
                                                                                        /// <param name="e">The e parameter is for the base class to pass event
                                                                                        /// data.</param>
         menuAll.Checked = true;
                                                                                        private void menuAll_Click(object sender, System.EventArgs e)
                                                                                          if (menuAll.Checked)
    picOriginalSmall.Update();
                                                                                            menuAll.Checked = false;
                                                                                            PicOriginalStretched = false;
                                                                                            menuLargeOriginal.Checked = false;
  /// <summary>
                                                                                            picOriginal.SizeMode = PictureBoxSizeMode.Normal;
  /// Pre-conditions:
                                                                                            PicManipulatedStretched = false;
  111
         The menuSmallManipulated Menu object has generated a Click event.
                                                                                            menuLargeManipulated.Checked = false;
  /// Post-conditions:
                                                                                            picManipulated.SizeMode = PictureBoxSizeMode.Normal;
  111
         If the picture in the picManipulatedSmall is in "normal" size mode, i
                                                                                            PicOriginalSmallStretched = false;
                                                                                            menuSmallOriginal.Checked = false;
  111
          will be changed to "stretch" size mode, otherwise it will be
                                                                                            picOriginalSmall.SizeMode = PictureBoxSizeMode.Normal;
          switched to "normal" size mode.
                                                                                            PicManipulatedSmallStretched = false;
  111
  111
       Description:
                                                                                            menuSmallManipulated.Checked = false;
  111
         This method is used to resolve a Click event generated by the
                                                                                            picManipulatedSmall.SizeMode = PictureBoxSizeMode.Normal;
  111
         menuSmallManipulated Menu object. The purpose of this Menu object is
                                                                                          else
  111
         to allow the user to toggle between "normal" and "stretch" size
  111
         modes for the original picture on the tabConsole Tab control.
                                                                                            menuAll.Checked = true;
  /// </summary>
                                                                                            PicOriginalStretched = true;
  /// <param name="sender">The sender parameter is a reference to the
                                                                                            menuLargeOriginal.Checked = true;
  /// function calling this function.</param>
                                                                                            picOriginal.SizeMode = PictureBoxSizeMode.StretchImage;
  /// <param name="e">The e parameter is for the base class to pass event
                                                                                            PicManipulatedStretched = true;
  /// data.</param>
                                                                                            menuLargeManipulated.Checked = true;
  private void menuSmallManipulated_Click(object sender, System.EventArgs e)
                                                                                            picManipulated.SizeMode = PictureBoxSizeMode.StretchImage;
```

```
frmMain.cs
                                  frmMain.cs
May 02, 04 2:03
                                                                                     May 02, 04 2:03
                                                                   Page 35/186
                                                                                                                                                        Page 36/186
      PicOriginalSmallStretched = true;
                                                                                       111
                                                                                              been closed within the picOriginal, picOriginalSmall,
                                                                                              picManipulated and picManipulatedSmall PictureBox controls before
      menuSmallOriginal.Checked = true;
                                                                                       ///
      picOriginalSmall.SizeMode = PictureBoxSizeMode.StretchImage;
                                                                                       111
                                                                                              trying to load the new images. This function should do some error
      PicManipulatedSmallStretched = true;
                                                                                       111
                                                                                              checking to make sure that these files actually exist before
      menuSmallManipulated.Checked = true;
                                                                                       111
                                                                                              trying to load them. If one (or both) of the parameters does not
      picManipulatedSmall.SizeMode = PictureBoxSizeMode.StretchImage;
                                                                                       111
                                                                                              contain a valid file name and path, then it should be ignored and
                                                                                       111
                                                                                              an error message should be displayed in the txtError. If an image
                                                                                       111
    picOriginal.Update();
                                                                                              exists, yet it is too far damaged to load into the PictureBox
    picManipulated.Update();
                                                                                       111
                                                                                              controls, then an error message should be displayed for the user
    picOriginalSmall.Update();
                                                                                       111
                                                                                              to see. If any errors occur during load time, the error should
    picManipulatedSmall.Update();
                                                                                       111
                                                                                              be displayed in the txtError TextBox for the user to see.
                                                                                       111
                                                                                       111
                                                                                              To perform this functionality, this function should call
                                                                                       111
                                                                                              ClearInterfaceData(), to clear the interface. It should call
                                                                                       111
                                                                                              UpdateManipulatedPicture() to load the picManipulated picture. If
  private void frmMain_Load(object sender, System.EventArgs e)
                                                                                       111
                                                                                              a valid file doesnM-^Rt exist in the ManipulatedFilePath parameter,
                                                                                       111
                                                                                              then it should just load the file in the OriginalFilePath
     // Create the new splash screen
    SplashScreen = new frmSplash();
                                                                                       111
                                                                                              parameter. If the OriginalFilePath parameter doesnM-^Rt contain a
    SplashScreen.Show();
                                                                                       111
                                                                                              valid file, this function should call one of the ShowWarning()
                                                                                       111
                                                                                              methods to let the user know that the OriginalFilePath is an
                                                                                       111
                                                                                              invalid file and in that case, no data should be loaded to the
    // Set the timer
    timerSplash.Enabled = true;
                                                                                       111
                                                                                              interface. This function should set the txtOrginalFile data
    timerSplash.Interval = 2000; // 2000 millisecs = 2 secs
                                                                                       111
                                                                                              member. It should also open the original file in the picOriginal
                                                                                       111
                                                                                              and picOriginalSmall PictureBox data members. Lastly, this
    timerSplash.Start();
                                                                                       111
                                                                                              function should call LoadPictureData() for the original file to
                                                                                       111
                                                                                              load all of the data into the TextBox fields of the Manipulator.
                                                                                       /// </summary>
  private void timerSplash_Tick(object sender, System.EventArgs e)
                                                                                       /// <param name="OriginalFilePath">The OriginalFilePath parameter is a
                                                                                       /// file path of the to the image to be loaded into the picOriginal and
     // Close the splash screen once the timer expires
                                                                                       /// picOriginalSmall.</param>
    SplashScreen.Close();
                                                                                       /// <param name="ManipulatedFilePath">The ManipulatedFilePath parameter
    SplashScreen.Dispose();
                                                                                       /// is a file path of the to the image to be loaded into the
                                                                                       /// picManipulated and picManipulatedSmall.</param>
    timerSplash.Dispose();
                                                                                       private void LoadPicture(string OriginalFilePath,
                                                                                                   string ManipulatedFilePath)
  private void menuTutorial_Click(object sender, System.EventArgs e)
                                                                                         // To solve the problem with controls not losing focus when
    System.Windows.Forms.Help.ShowHelp(
                                                                                         // a new picture is loaded.
      this, ProgramDirectory + @"\ISE Manipulator Tutorial.pdf");
                                                                                         this.tabFile.Focus();
                                                                                         this.Update();
                                                                                         try
  private void menuManual_Click(object sender, System.EventArgs e)
                                                                                           LoadingInterface = true;
    System.Windows.Forms.Help.ShowHelp(
      this, ProgramDirectory + @"\ISE Manipulator Manual.pdf");
                                                                                           if(txtOriginalFile.Text != "")
                                                                                             if(!ShowWarning(
                                                                                               "\nYou currently have a file open for editing.\n" +
                                                                                               "If you open a newfile, all unsaved data will be lost!\n" +
  #endregion Interface Methods
                                                                                                "Are you sure you want to open this new file?"))
                                                                                               LoadingInterface = false;
  #region Common Methods
                                                                                               return;
  /// <summary>
                                                                                             ClearInterfaceData();
  /// Pre-conditions:
                         None.
  /// Post-conditions:
                                                                                           } // End of: if(txtOriginalFile.Text != "")
  111
         An original JPEG image has been loaded into the picOriginal and
  111
         picOriginalSmall PictureBox data members and a manipulated JPEG
                                                                                           this.Update();
  111
         image has been loaded into the picManipulated and
  111
         picManipulatedSmall data members. Also, all of the data contained
                                                                                           // This is for the Original Picture
  111
         in the original file should be loaded into the interface to
                                                                                           // Clear out the old image
  111
         display for the user.
                                                                                           if(JPEG != null) JPEG.Dispose();
  111
       Description:
                                                                                           if (JPEGsmall != null) JPEGsmall.Dispose();
  111
         This method should be called if the Manipulator needs to be
  111
         completely reload. This method should be used by any other function
                                                                                           // Load the Original pic and resize to control size.
  111
         that needs to reload both images and the data into the interface.
                                                                                           JPEG = new Bitmap(OriginalFilePath);
  111
         This method should check to make sure that any previous image has
                                                                                           if (menuLargeOriginal.Checked)
```

May 02, 04 2:03	frmMain.cs	Page 37/186	May 02, 04 2:03	frmMain.cs	Page 38/186
{			LoadingInterface =	false;	
PicOriginalStreto picOriginal.SizeN	ched = true; Mode = PictureBoxSizeMode.Stretch	Image;	} // End of: private	void LoadPicture()	
} else { PicOriginalStret(	shed = false.		/// <summary></summary>	hod definition	
picOriginal.Size	Mode = PictureBoxSizeMode.Normal;		/// private void LoadNew	Picture()	
picOriginal.Image = picOriginal.Update	= (Image)JPEG; ();		{ try		
// Load the console JPEGsmall = new Bit	e tab picture too cmap(OriginalFilePath);		{ LoadingInterface	= true;	
{ PicOriginalSmalls	Stretched = true:		{	IE.IEXt :- )	
picOriginalSmall, } else	SizeMode = PictureBoxSizeMode.St	retchImage;	if(!ShowWarnin "\nYou curre "If you open	g( ntly have a file open for editing. a newfile, all unsaved data will	.\n" + be lost!\n" +
{ PicOriginalSmalls	Stretched = false;		"Are you sur {	e you want to open this new file?"	'))
picOriginalSmall.	.SizeMode = PictureBoxSizeMode.No	rmal;	LoadingInter return;	<pre>face = false;</pre>	
picOriginalSmall.Ir picOriginalSmall.Up	<pre>nage = (Image)JPEGsmall; odate();</pre>		} } // End of: if(	<pre>txtOriginalFile.Text != "")</pre>	
// Load the Manipul UpdateManipulatedP	lated pic from same picture. icture(ManipulatedFilePath);		else if(txtProje {	ctPath.Text != "")	
<pre>// Update File Info txtOriginalFile.Te;</pre>	o kt = OriginalFilePath;		if(!ShowWarnin "\nYou curre "If you open "Are you sur	g( ntly have a file open for editing. a newfile, all unsaved data will e you want to open this new file?"	\n" + be lost!\n" + '))
<pre>// Create a name fo ManipulatedFileName txtManipulatedFile this.Update():</pre>	<pre>br the changed file = ManipulatedFilePath; Text = ManipulatedFileName;</pre>		{ LoadingInter return; }	face = false;	
// Load all of the LoadPictureData(Or:	Data Values into the interface lginalFilePath);		} // End of: if( ClearInterfaceDa	<pre>txtOriginalFile.Text != "") ta();</pre>	
// Update frmMain T this.Text = "ISE JH + openFileDialog	Fext PEG Manipulator - Version " + VER. FileName;	SION + " - "	openFileDialog.S if(openFileDialo {	howHelp = false; g.ShowDialog() == DialogResult.OK)	1
LoadingInterface =	false;		<pre>this.Update();</pre>		
<pre>} // End of: try bloc catch(Exception ex) {</pre>	ck		<pre>// This is for // Clear out t if(JPEG != nul if(JPEGsmall '</pre>	<pre>the Original Picture he old image 1) JPEG.Dispose(); = null) JPEGsmall.Dispose();</pre>	
<pre>if(ex.Message == "1     ex.Message == "A     ex.Source == "Sys {</pre>	<pre>Invalid parameter used."    generic error occurred in GDI+." stem.Drawing")</pre>		// Load the Or JPEG = new Bit if(menuLargeOr	<pre>iginal pic and resize to control s map(openFileDialog.FileName); iginal.Checked)</pre>	size.
OriginalFilePath LoadPicture(Orig: }	= ProgramDirectory + @"\default inalFilePath, OriginalFilePath);	bad.jpg";	{ PicOriginalS picOriginal.	<pre>tretched = true; SizeMode = PictureBoxSizeMode.Stre</pre>	etchImage;
else {			} else		
ShowWarning( "Warning, an ex "Exception Erro	<pre>kception occured:\n\n" + pr:\n" +</pre>		{ PicOriginalS picOriginal.	tretched = false; SizeMode = PictureBoxSizeMode.Norm	nal;
ex.Message + " ex.Source + "\n\nNot all lo	<pre>\n\nwas throw by:\n" + pad operations completed.!", pation");</pre>		} picOriginal.Im picOriginal.Up	<pre>hage = (Image)JPEG; date();</pre>	
ClearInterfaceDat }	ta();		<pre>// Load the co JPEGsmall = ne if(menuSmallOr</pre>	nsole tab picture too w Bitmap(openFileDialog.FileName); iginal.Checked)	

```
frmMain.cs
                                                                                                                         frmMain.cs
 May 02, 04 2:03
                                                                     Page 39/186
                                                                                       May 02, 04 2:03
                                                                                                                                                           Page 40/186
                                                                                                  "Exception Error:\n" +
            PicOriginalSmallStretched = true;
                                                                                                  ex.Message + "\n\nWas throw by:\n" +
           picOriginalSmall.SizeMode = PictureBoxSizeMode.StretchImage;
                                                                                                  ex.Source +
                                                                                                  "\n\nNot all load operations completed.!",
                                                                                                  "Load File Exception");
          else
                                                                                                ClearInterfaceData();
            PicOriginalSmallStretched = false;
            picOriginalSmall.SizeMode = PictureBoxSizeMode.Normal;
                                                                                           LoadingInterface = false;
          picOriginalSmall.Image = (Image)JPEGsmall;
          picOriginalSmall.Update();
                                                                                          } // End of: private void LoadNewPicture()
          // Load the Manipulated pic from same picture.
          UpdateManipulatedPicture(openFileDialog.FileName);
                                                                                          /// <summary>
                                                                                          /// Pre-conditions:
          // Update File Info
                                                                                                 The data of an image has been previously loaded into the
          txtOriginalFile.Text = openFileDialog.FileName;
                                                                                          111
                                                                                                 Manipulator.
                                                                                          /// Post-conditions:
          // Create a name for the changed file
                                                                                               A new image based on the FileName parameter has been loaded into
                                                                                          111
          ManipulatedFileName = openFileDialog.FileName;
                                                                                                the picManipulated and the picManipulatedSmall data fields.
                                                                                          111
          string ttt = ManipulatedFileName.ToLower();
                                                                                          111
                                                                                              Description:
          ManipulatedFileName = ManipulatedFileName.ToLower();
Count = ttt.IndexOf(".jpg");
                                                                                          111
                                                                                                This function is used to update picManipulated and
                                                                                                picManipulatedSmall data members, by loading a pre-existing
                                                                                          111
                                                                                          111
                                                                                                image. If the FileName parameter is not a valid JPEG image, then
          // Manipulated the file name if it already exists
                                                                                          111
                                                                                                an error message should be displayed by calling the ShowWarning()
                                                                                                method. Lastly, this method should do some error checking to
          ManipulatedFileName =
                                                                                          111
                                                                                                make sure this function executes properly. If an error is
                          ManipulatedFileName.Insert(Count, "_changed0");
                                                                                          111
                                                                                          111
                                                                                                 encountered, then the ShowWarning() method should be called to
          Temp = 0;
          string num_length;
                                                                                          111
                                                                                                display the error to the user and the txtError TextBox control
                                                                                          111
                                                                                                 should be updated with this error information.
          while (File.Exists (ManipulatedFileName))
                                                                                          /// </summary>
            Count = ManipulatedFileName.IndexOf(Temp.ToString() + ".jpg");
                                                                                          /// <param name="FileName">The FileName parameter is the name and path
            num_length = Temp.ToString();
                                                                                          /// of a JPEG file to be loaded.</param>
            ManipulatedFileName =
                                                                                          private void UpdateManipulatedPicture(string FileName)
                                ManipulatedFileName.Remove(Count, num length.Len
gth);
                                                                                           try
            Temp++;
            ManipulatedFileName
                                                                                              // This is for the Manipulated Picture
                                ManipulatedFileName.Insert(Count, Temp.ToString(
));
                                                                                              // Clear out the old images
                                                                                              if(ISE != null) ISE.Dispose();
                                                                                              if(ISEsmall != null) ISEsmall.Dispose();
          txtManipulatedFile.Text = ManipulatedFileName;
          this.Update();
                                                                                              // Open the new file and resize to control size.
                                                                                              ISE = new Bitmap(FileName);
          // Load all of the Data Values
                                                                                              if (menuLargeManipulated.Checked)
          LoadPictureData(openFileDialog.FileName);
                                                                                                PicManipulatedStretched = true;
          // Update frmMain Text
                                                                                                picManipulated.SizeMode = PictureBoxSizeMode.StretchImage;
          this.Text = "ISE JPEG Manipulator - Version " + VERSION + "
            + openFileDialog.FileName;
                                                                                              else
          LoadingInterface = false;
                                                                                                PicManipulatedStretched = false;
                                                                                                picManipulated.SizeMode = PictureBoxSizeMode.Normal;
      } // End of: try block
                                                                                              picManipulated.Image = (Image)ISE;
      catch (Exception ex)
                                                                                              picManipulated.Update();
        if(ex.Message == "Invalid parameter used." ||
                                                                                              // Load the console tab picture too
          ex.Message == "A generic error occurred in GDI+." ||
                                                                                              ISEsmall = new Bitmap(FileName);
          ex.Source == "System.Drawing")
                                                                                              if (menuSmallManipulated.Checked)
          string x = ProgramDirectory + @"\default_bad.jpg";
                                                                                                PicManipulatedSmallStretched = true;
          LoadPicture(x, x);
                                                                                                picManipulatedSmall.SizeMode = PictureBoxSizeMode.StretchImage;
        else
                                                                                              else
          ShowWarning(
                                                                                                PicManipulatedSmallStretched = false;
            "Warning, an exception occured:\n\n" +
                                                                                                picManipulatedSmall.SizeMode = PictureBoxSizeMode.Normal;
```

May 02, 04 2:03	frmMain.cs	Page 41/186	May 02, 04 2:03	frmMain.cs	Page 42/186		
<pre>} picManipulatedSmall.Imag picManipulatedSmall.Upda</pre>	<pre>ge = (Image)ISEsmall; ate();</pre>		/// <summary></summary>	Nana			
catch(Exception ex)			/// Post-conditions:	None.			
<pre>{     if(ex.Message == "Invalid parameter used."        ex.Message == "A generic error occurred in GDI+."        ex.Source == "System.Drawing")     {         UpdateManipulatedPicture(ProgramDirectory + @"\default_bad.jpg");     } }</pre>			<pre>/// A warning message box is displayed for the user to see and decide /// how to proceed. This box will be shown until the user clicks /// either the Ok or Cancel Button control on this message box, at /// which point, this method will exit. /// Description: /// This function is a simpler version of the other ShowWarning /// This function is a simpler version of the other ShowWarning</pre>				
else			/// message box.	Then, this function will call the ot	ther		
{ if(ShowWarning( "An Exception Occure "\n\nThe "\n\nException Messa "\n\nDo you want to	ed!" + Manipulator Failed to Load age: " + ex.Message + "\n\n" reload the original picture	the File properly."	<pre>/// Showwarning(s /// message param ///  /// <param name="mes /// warning message. /// &lt;returns&gt;Functic /// if the user has&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;string message, string caption) method&lt;br&gt;neter and the default title created.&lt;br&gt;ssage"/>The message parameter is explan . on returns True if the user has clicke clicked Cancel.</pre>	ation of the d Ok and False			
"An Exception Occure	ed!"	. ,	private bool ShowWar	rning(string message)			
{			return ShowWarning	g(message, "Warning!");			
UpdateManipulatedPic	ture(txtoriginalFile.Text.1	rim());	}				
else {			/// <summary></summary>				
UpdateManipulatedPic ProgramDirectory - } }	:ture( - @"\default_bad.jpg");		/// Pre-conditions: /// Post-conditions: /// All of the Te /// Manipulator	None. : extBox controls for all of the data fi will be reinitialized to empty strings	elds within the		
<pre>} // End of: private void [ // courseway]</pre>	JpdateManipulatedPicture(str	ing FileName)	/// Description: /// This purpose /// that needs to /// interface. S	of this method is to be called by any o clear out all of the data fields wit Specifically, this method should set a	other method hin the user all of the		
/// Pre-conditions: None /// Post-conditions: /// A warning message boo /// how to proceed. This /// either the Ok or Cano	s is displayed for the user s box will be shown until the cel Button control on this m	to see and decide e user clicks essage box, at	<pre>/// stillings to ei /// sub-tabs of t /// It should als /// all of the Ta ///  private void ClearIn</pre>	the Console tab on the Manipulator frm so clear out all of the PictureBox con ab controls of the application. nterfaceData()	Main Form. trols within		
<pre>/// which point, this met /// Description: /// The purpose of this n /// to display a warning</pre>	hod will exit. method is to be used by any message to the user. In ac	method that wants ddition, this	<pre>{     // Text fields to     this.txtApplicatio     this.txtApplicatio</pre>	<pre>clear. onData1.Text = ""; onData2.Text = "";</pre>			
<pre>/// method should return /// response given by the /// should call the stand /// </pre>	a frue or false value, depe : user receiving this messag lard MessageBox control to s	nding on the e. This method how the message.	this.txtApplicatic this.txtApplicatic this.txtApplicatic this.txtApplicatic	<pre>&gt;nData3.lext = ""; &gt;nData4.Text = ""; &gt;nData5.Text = ""; onData6.Text = "";</pre>			
<pre>/// <param name="message"/>Tl /// warning message.</pre>	1e message parameter is expl >	anation of the	this.txtApplication this.txtApplication	<pre>&gt;nData7.Text = ""; &gt;nData8.Text = "";</pre>			
<pre>/// <param name="caption"/>Th /// warning message box.Function return</pre>	e caption parameter is Wind aram> is True if the user has clic	ow title of ked Ok and False	this.txtApplication this.txtApplication this.txtApplication the second s	<pre>onData9.Text = ""; onData10.Text = "";</pre>			
/// if the user has clicked private bool ShowWarning(st	Cancel. cing message, string captior	ι)	this.txtManipulate this.txtComments.1	edFile.Text = ""; Text = "";			
<pre>{     string t = message.ToStrin </pre>	ng();		this.txtEncodedDat this.txtError.Text	<pre>:a.Text = ""; c = ""; </pre>			
if(!(t.Length > 0)) t = " if(MessageBox.Show(	;		this.txtExpand.Tex this.txtFileSize.T	<t "";<br="" =="">Iext = "0";</t>			
"Warning:\n" + t, caption,			this.txtHierarchia this.txtNumberLine	al.Text = ""; es.Text = "";			
MessageBoxButtons.OKCan	pielogRegult (K)		this.txtOriginalEr	<pre>ncodedData.Text = ""; ile Text = "";</pre>			
<pre>ressayeboxicon.filof) == { </pre>	- Dratognesurc.ON)		this.txtOriginalHe	eader.Text = "";			
return true; }			this.txtRestart.Te this.txtRestartMod	∍xt = ""; d8.Text = "";			
else return false;			this.txtScanHeader	c.Text = "";			

May 02, 04 2:03	frmMain.cs	Page 43/186	May 02, 04 2:03	frmMain.cs	Page 44/186
this.txtHuffman1.Tex	xt = "";		this.lblHuffmanOrigir	nalMarker3.Text = "";	
this.txtHuffman2.Tex	xt = "":		this lblHuffmanOrigin	nalMarker4.Text = "":	
this txtHuffman3 Tex	xt = ""·		this lblHuffmanOrigin	nalMarker5 Text = "":	
this tytHuffman4 Tex	vt = "".		this lblHuffmanOrigin	nalMarker6 Text - "":	
this tytHuffman5 Tex	vt = "".		this lblHuffmanOrigin	nalMarker7 Text - "":	
thic tytHuffman6 Tox	xc - , xt _ "".		this lblHuffmanOrigin	nalMarkorg Toxt = "":	
this tytusfman7 To	x = r		ciiis.ibiiluiillalloiigii	lainaikeio.iekt – ,	
this tutluffman Tak	x = i		this lbloughting Manh	Lond Tout III.	
this test Use for a Original	$KL = \cdots;$		this lbloughtizerMark	<pre>keri.lext = "";</pre>	
this tet Us for a Origi	inali.lext = "";		this lbloughtizerMark	<pre>kerz.lext = "";</pre>	
this txtHuffmanOrigi	inal2.lext = "";		this.lblQuantizerMark	(ers.lext = "";	
this.txtHurrmanOrigi	inal3.lext = "";		this.lbiQuantizerMark	<pre>(er4.lext = "";</pre>	
this.txtHuffmanOrigi	inal4.Text = "";				
this.txtHuffmanOrigi	inal5.Text = "";		this.lblQuantizerOric	<pre>jinalMarker1.Text = "";</pre>	
this.txtHuffmanOrigi	inal6.Text = "";		this.lblQuantizerOric	<pre>jinalMarker2.Text = "";</pre>	
this.txtHuffmanOrigi	inal/.Text = "";		this.lblQuantizerOric	<pre>jinalMarker3.Text = "";</pre>	
this.txtHuffmanOrigi	inal8.Text = "";		this.lblQuantizerOrio	jinalMarker4.Text = "";	
this tytowastigor1 7	Foxt _ "".		// Dicture components	a to alcor	
this tytousstiger? 7	$\Gamma_{OXT} = i$		nicOriginal Trage	s co crear	
this tutQuantizer2.	Text = ";		picOriginal.Image = 1	1011;	
this twtQuantizers.	Text = "";		picoriginal.update();	<i>i</i>	
this.txtQuantizer4.1	lext = "";		picOriginalSmall.Imag	je = null;	
this.txtQuantizerOri	iginall.lext = "";		picOriginalSmall.Upda	ate();	
this.txtQuantizerOri	iginal2.Text = "";		picManipulated.Image	= null;	
this.txtQuantizerOri	iginal3.Text = "";		picManipulated.Update	e();	
this.txtQuantizerOri	iginal4.Text = "";		picManipulatedSmall.1	<pre>image = null;</pre>	
this.txtQuantizerTak	pleNuml.Text = "";		picManipulatedSmall.U	Jpdate();	
this.txtQuantizerTak	<pre>pleNum2.Text = "";</pre>		}		
this.txtQuantizerTak	pleNum3.Text = "";				
this.txtQuantizerTak	oleNum4.Text = "";				
			/// <summary></summary>		
this.txtProjectPath.	.Text = "";		/// Pre-conditions:	None.	
this.txtNotes.Text =	= "";		/// Post-conditions:		
			/// A new file with	the data contained in the ByteDa	ataToWrite array
txtStartHuffman.Text	t = "";		/// has been created	1.	
txtStartHuffmanSize.	.Text = "";		/// Description:		
txtPrecision.Text =	"";		/// The Purpose of t	chis function is to allow the cal	ller to create a
txtNumberHuffmanLine	es.Text = "";		/// new file based u	upon the data in the byte array p	passed in. This
txtNumberHuffmanSamp	ples.Text = "";		/// file created sho	ould be the binary value of the B	oyte array and
txtNumberImageCompor	nents.Text = "";		/// nothing more. 1	If the byte array is null then ar	n empty file
<pre>txtComponents.Text =</pre>	= "";		/// should be create	ed. The name of this file will b	pe based upon file
			/// name in the txtM	ManipulatedFile TextBox control.	Lastly, this
// Label fields to c	clear		/// method should do	o some error checking to make su:	re this function
this.lblApplicationM	Marker1.Text = "";		<pre>/// executes properl</pre>	ly. If an error is encountered,	then the
this.lblApplicationM	Marker2.Text = "";		/// ShowWarning() me	ethod should be called to display	y the error to the
this.lblApplicationN	Marker3.Text = "";		<pre>/// user and the txt</pre>	Error TextBox control should be	updated with this
this.lblApplicationN	Marker4.Text = "";		/// error informatio	on.	-
this.lblApplicationN	Marker5.Text = "";		///		
this.lblApplicationN	Marker6.Text = "";		/// <param name="ByteDa&lt;/td&gt;&lt;td&gt;ataToWrite"/> The ByteDataToWrite	parameter is byte	
this.lblApplicationN	Marker7.Text = "";		/// array of data to be	e written to file.	
this.lblApplicationN	Marker8.Text = "";		private void WriteFile	(ref bvte[] BvteDataToWrite)	
this.lblApplicationN	Marker9.Text = "";		{	, , , , , , , , , , , , , , , , , , , ,	
this.lblApplicationM	Marker10.Text = "";		trv		
± ±	·		{		
this.lblExpandMarker	r.Text = "";		int c = FileSize;		
this.lblHierarchialN	Marker.Text = "";				
this.lblNumberLinesN	Marker.Text = "";		// Open the Origina	al File to Setup Data	
this.lblRestartMarke	er.Text = "";		if (NewFile != null)	) NewFile.Close();	
			if (File.Exists(txtM	ManipulatedFile.Text))	
this.lblHuffmanMarke	er1.Text = "";		Fil	le.Delete(txtManipulatedFile.Tex'	z);
this.lblHuffmanMarke	er2.Text = "";		NewFile = File.Oper	nWrite(this.txtManipulatedFile.T	ext);
this.lblHuffmanMarke	er3.Text = "";			, <u>F</u> ======================	
this.lblHuffmanMarke	er4.Text = "";		if (c >= BvteDataTo	oWrite.Length) c = BvteDataToWrig	te.Length;
this.lblHuffmanMarke	er5.Text = "":		NewFile.Write(Byter	DataToWrite, 0, c):	,
this.lblHuffmanMarke	er6.Text = "":				
this.lblHuffmanMarke	er7.Text = "":		// Close the file w	when complete	
this.lb]HuffmanMarke	er8.Text = "";		NewFile.Close():		
	· · · · · · · · · · · · · · · · · · ·		}		
this.lblHuffmanOrigi	inalMarker1.Text = "";		catch(Exception EX)		
this.lblHuffmanOrigi	inalMarker2.Text = "";		{		
. ,	•				

```
frmMain.cs
                                                                                                                       frmMain.cs
May 02, 04 2:03
                                                                   Page 45/186
                                                                                     May 02, 04 2:03
                                                                                                                                                         Page 46/186
      // Catch some exceptions
                                                                                        111
                                                                                               error checking to make sure all of the images load and that this
      if(!ShowWarning(
                                                                                        ///
                                                                                               method executes properly. If there is an error, the
         "An EXCEPTION occured!! Exception: \n" +
                                                                                        111
                                                                                               ShowWarning() method should be called and the txtError TextBox
         EX.Message + "\n\nThrown by: \n" + EX.Source +
                                                                                        111
                                                                                               control should be updated with this error information.
         "\n\nWould you like to TRY to continue? \n" +
                                                                                        /// </summary>
         "(If you choose OK, unexpected results may occur!)",
                                                                                        private void LoadNewProject()
         "An Exception Occured!"))
                                                                                          this.tabProject.Focus();
                                                                                          this.Update();
        ClearInterfaceData();
                                                                                          try
  }
                                                                                            openFileDialog1.ShowHelp = false;
                                                                                            if(openFileDialog1.ShowDialog() != DialogResult.OK) return;
  /// <summary>
  /// Pre-conditions:
                                                                                            if(txtProjectPath.Text != "")
                         None.
  /// Post-conditions:
  111
         All of the data members used to store information about the file
                                                                                              if(!ShowWarning(
  111
         structure of the current JPEG image are reinitialized to zero.
                                                                                                "\nYou currently have a file open for editing.\n" +
                                                                                                "If you open a newfile, all unsaved data will be lost!\n" +
  111
       Description:
  111
          The purpose of this method is to allow the caller to reinitialize
                                                                                                "Are you sure you want to open this new file?"))
  111
         all of the data members that store information about the structure
  111
         of the previous JPEG image loaded. This function should set the
                                                                                                return:
  111
         following data members to zero: NumberOfLines, RestartInterval,
  111
         FrameSize, ExpandImage, RestartMod8, SizeOfHuffman (all 8 array
                                                                                            } // End of: if(txtProjectPath.Text != "")
  111
         members), SizeOfQuantizer (all 4 array members), SizeOfAppData
  111
          (all 10 array members), SizeOfScanHeader, SizeOfProgression and
                                                                                            if(txtProjectPath.Text.Trim() != "")
  111
          SizeOfComments. Also, the FileOrder Queue should be cleared.
  /// </summary>
                                                                                              if(!ShowWarning(
  private void ClearData()
                                                                                                "\nYou currently have a file open for editing.\n" +
                                                                                                "If you open a newfile, all unsaved data will be lost!\n" +
    int i = 0;
                                                                                                "Are you sure you want to open this new file?"))
    NumberOfLines = 0;
                                                                                                return;
    RestartInterval = 0;
    FrameSize = 0;
                                                                                            } // End of: if(txtProjectPath.Text != "")
    ExpandImage = 0;
    RestartMod8 = 0;
                                                                                            else if(txtOriginalFile.Text.Trim() != "")
                                                                                              if(!ShowWarning(
    FileOrder.Clear();
                                                                                                "\nYou currently have a picture file open for editing.\n" +
                                                                                                "If you open a newfile, all unsaved data will be lost!\n" +
    for(i = 0; i < MAX HUFFMAN; i++) SizeOfHuffman[i] = 0;</pre>
                                                                                                "Are you sure you want to open this new file?"))
    for(i = 0; i < MAX_QUANTIZER; i++) SizeOfQuantizer[i] = 0;</pre>
    for(i = 0; i < MAX_APPDATA; i++) SizeOfAppData[i] = 0;</pre>
                                                                                                return;
    SizeOfScanHeader = 0:
    SizeOfProgression = 0;
    SizeOfComments = 0;
                                                                                            // Clear the interface
                                                                                            ClearInterfaceData();
                                                                                            txtProjectPath.Text = openFileDialog1.FileName;
  }
                                                                                            // Open the file to read from
  /// <summary>
                                                                                            StreamReader sr = new StreamReader(openFileDialog1.FileName);
  /// Pre-conditions:
                         None.
  /// Post-conditions:
                                                                                            string S, original_file_path, changed_file_path;
  111
         A previously existing SEP project file has been reloaded into the
                                                                                            char [] Data = null;
  111
         Manipulator.
                                                                                            int Size:
   111
       Description:
   111
        The purpose of the function is to allow the caller to load a
   111
         pre-existing SEP project file. This function should prompt the
                                                                                            // Read the data from SEP file
  111
         user to save the current project, if there is one currently
                                                                                            11
  111
          loaded. Then this function should call the ClearInterfaceData()
  111
         method and then should open the file and read all data, to reload
                                                                                            original_file_path = "";
  111
          all of the corresponding fields in the interface. This method
                                                                                            changed_file_path = "";
  111
          should load the project notes stored in the SEP file into the
  111
          txtNotes TextBox interface control. This method should also
                                                                                            // Get the Notes data
  111
          reload all of the PictureBox controls from the image file
                                                                                            S = sr.ReadLine();
  111
          information stored in the SEP file. This method should do some
                                                                                            Size = System.Convert.ToInt32(S.Trim());
```

May 02, 04 2:03	frmMain.cs	Page 47/186	May 02, 04 2:03	frmMain.cs	Page 48/186
if(Size > 0)			for(int $i = 0;$	i < Data.Length; i++)	
{ Data = new char	[Size]·		txtFileSize.	<pre>Text += Data[i].ToString();</pre>	
sr.Read(Data, 0,	Size);		}		
for(int i = 0; i	< Data.Length; i++)		// Cot the File	Commonto	
Data = null;	- Data[i].10Stillig();		S = sr.ReadLine(	);	
}			Size = System.Co	<pre>onvert.ToInt32(S.Trim());</pre>	
//			11(S1Ze > 0) {		
// File Tab Data			txtComments.Te	ext = "";	
//			Data = new cha sr.Read(Data,	ur [Size]; 0, Size);	
// Get the Origina	l File Path		for(int $i = 0;$	i < Data.Length; i++)	
S = sr.ReadLine(); Size = System.Conv	rert.ToInt32(S.Trim()):		txtComments. Data = null:	Text += Data[1].ToString();	
if(Size > 0)			}		
{ Data = new char	[Size]:		11		
sr.Read(Data, 0,	Size);		// Header Tab Da	ta	
for(int i = 0; i	<pre>&lt; Data.Length; i++) path += Data[i] ToString().</pre>		//		
Data = null;	pach - Data[1].100tring(),		// Get the Start	of Compression Marker	
}			S = sr.ReadLine(	);	
// Get the Manipul	ated File Path		if(Size > 0)	mvert.101mt32(5.111m());	
S = sr.ReadLine();			{		
if(Size > 0)	ert.101nt32(S.1r1m());		Data = new cha	n.lext = ""; r [Size];	
{			sr.Read(Data,	0, Size);	
Data = new char sr.Read(Data, 0,	[Size]; Size):		<pre>tor(int i = 0;     txtStartHuff</pre>	<pre>i &lt; Data.Length; i++) man.Text += Data[i].ToString():</pre>	
for(int $i = 0; i$	< Data.Length; i++)		Data = null;		
changed_file_p Data = null:	<pre>path += Data[i].ToString();</pre>		}		
}			// Get the Start	of Compression Header Size	
			S = sr.ReadLine( Size = System Co	); nvert ToInt32(S Trim());	
if(File.Exists(ori	ginal_file_path))		if(Size > 0)	mvere.ioines2(5.iiim()),	
{ LoadPicture(orig	inal file path. changed file path)		{ txtS	tartHuffmanSize.Text = "":	
}	pacn, changeapacn,	,	Data = new cha	r [Size];	
else			<pre>sr.Read(Data, for(int i = 0)</pre>	0, Size); i < Data Length: i++)	
if(ShowWarning(			txtStartHuff	<pre>manSize.Text += Data[i].ToString();</pre>	
"The Original	Picture file path:\n" + original_f	ile_path +	Data = null;		
"\n\nDo you wa	int to browse to the picture locati	on?",	J		
"Invalid File	Path!!"))		// Get the Preci	sion	
LoadNewPicture	e();		Size = System.Co	<pre>pnvert.ToInt32(S.Trim());</pre>	
}			if(Size > 0)		
{			txtPrecision.T	'ext = "";	
ShowWarning("L	oad Project operation has been can	nceled.",	Data = new cha	r [Size];	
ClearInterface	Data();		for (int $i = 0;$	i < Data.Length; i++)	
return;			txtPrecision	.Text += Data[i].ToString();	
}			Data = null; }		
// Cot the Till Of	ro data		// Cot the Thefe	an Lince	
<pre>// Get the File Si S = sr.ReadLine();</pre>	Ze uala		// Get the Hullm S = sr.ReadLine(	);	
Size = System.Conv	<pre>rert.ToInt32(S.Trim());</pre>		Size = System.Co	<pre>onvert.ToInt32(S.Trim());</pre>	
11(S1ZE > U) {			11(S1ZE > 0) {		
txtFileSize.Text	· = "";		txtNumberHuffm	anLines.Text = "";	
Data = new char sr.Read(Data, 0,	[Size]; Size);		Data = new cha sr.Read(Data,	0, Size);	

```
frmMain.cs
                                                                                                                              frmMain.cs
May 02, 04 2:03
                                                                       Page 49/186
                                                                                          May 02, 04 2:03
                                                                                                                                                                Page 50/186
         for(int i = 0; i < Data.Length; i++)</pre>
                                                                                                     txtHuffman1.Text += Data[i].ToString();
           txtNumberHuffmanLines.Text += Data[i].ToString();
                                                                                                   Data = null:
         Data = null;
                                                                                                 S = sr.ReadLine();
       // Get the Huffman Samples
                                                                                                 Size = System.Convert.ToInt32(S.Trim());
       S = sr.ReadLine();
                                                                                                 if(Size > 0)
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
                                                                                                   lblHuffmanOriginal1.Text = "";
                                                                                                   Data = new char [Size];
         txtNumberHuffmanSamples.Text = "";
                                                                                                   sr.Read(Data, 0, Size);
for(int i = 0; i < Data.Length; i++)</pre>
         Data = new char [Size];
         sr.Read(Data, 0, Size);
                                                                                                     lblHuffmanOriginal1.Text += Data[i].ToString();
         for(int i = 0; i < Data.Length; i++)</pre>
                                                                                                   Data = null;
           txtNumberHuffmanSamples.Text += Data[i].ToString();
         Data = null;
                                                                                                 S = sr.ReadLine();
                                                                                                 Size = System.Convert.ToInt32(S.Trim());
       // Get the Number of Image Components
                                                                                                 if(Size > 0)
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
                                                                                                   txtHuffmanOriginal1.Text = "";
       if(Size > 0)
                                                                                                   Data = new char [Size];
                                                                                                   sr.Read(Data, 0, Size);
for(int i = 0; i < Data.Length; i++)</pre>
         txtNumberImageComponents.Text = "";
         Data = new char [Size];
                                                                                                     txtHuffmanOriginal1.Text += Data[i].ToString();
         sr.Read(Data, 0, Size);
                                                                                                   Data = null;
         for(int i = 0; i < Data.Length; i++)</pre>
          txtNumberImageComponents.Text += Data[i].ToString();
         Data = null;
                                                                                                 // Get Compression Table 2 Data
                                                                                                 S = sr.ReadLine();
                                                                                                 Size = System.Convert.ToInt32(S.Trim());
       // Get the Number of Components
                                                                                                 if(Size > 0)
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
                                                                                                   lblHuffman2.Text = "";
       if(Size > 0)
                                                                                                   Data = new char [Size];
                                                                                                   sr.Read(Data, 0, Size);
for(int i = 0; i < Data.Length; i++)</pre>
         txtComponents.Text = "";
         Data = new char [Size];
                                                                                                     lblHuffman2.Text += Data[i].ToString();
         sr.Read(Data, 0, Size);
                                                                                                   Data = null;
         for(int i = 0; i < Data.Length; i++)</pre>
           txtComponents.Text += Data[i].ToString();
         Data = null;
                                                                                                 S = sr.ReadLine();
                                                                                                 Size = System.Convert.ToInt32(S.Trim());
       }
                                                                                                 if(Size > 0)
       // Huffman Table Data
                                                                                                   txtHuffman2.Text = "";
                                                                                                   Data = new char [Size];
                                                                                                   sr.Read(Data, 0, Size);
                                                                                                   for(int i = 0; i < Data.Length; i++)</pre>
       // Get Compression Table 1 Data
       S = sr.ReadLine();
                                                                                                     txtHuffman2.Text += Data[i].ToString();
       Size = System.Convert.ToInt32(S.Trim());
                                                                                                   Data = null;
       if(Size > 0)
         lblHuffman1.Text = "";
                                                                                                 S = sr.ReadLine();
                                                                                                 Size = System.Convert.ToInt32(S.Trim());
         Data = new char [Size];
         sr.Read(Data, 0, Size);
                                                                                                 if(Size > 0)
         for(int i = 0; i < Data.Length; i++)</pre>
           lblHuffman1.Text += Data[i].ToString();
                                                                                                   lblHuffmanOriginal2.Text = "";
         Data = null;
                                                                                                   Data = new char [Size];
                                                                                                   sr.Read(Data, 0, Size);
                                                                                                   for(int i = 0; i < Data.Length; i++)</pre>
       S = sr.ReadLine();
                                                                                                     lblHuffmanOriginal2.Text += Data[i].ToString();
       Size = System.Convert.ToInt32(S.Trim());
                                                                                                   Data = null;
       if(Size > 0)
         txtHuffman1.Text = "";
                                                                                                 S = sr.ReadLine();
         Data = new char [Size];
                                                                                                 Size = System.Convert.ToInt32(S.Trim());
         sr.Read(Data, 0, Size);
                                                                                                 if(Size > 0)
         for(int i = 0; i < Data.Length; i++)</pre>
```

May 02, 04 2:03	frmMain.cs	Page 51/186	May 02, 04 2:03	frmMain.cs	Page 52/186
<pre>txtHuffmanOrigin. Data = new char sr.Read(Data, 0, for(int i = 0; i txtHuffmanOrig: Data = null; }</pre>	al2.Text = ""; [Size]; Size); < Data.Length; i++) inal2.Text += Data[i].ToString();		<pre>S = sr.ReadLine() Size = System.Con if(Size &gt; 0) {    txtHuffman4.Tex Data = new char</pre>	; vvert.ToInt32(S.Trim()); :t = ""; : [Size];	
<pre>// Get Compression S = sr.ReadLine(); Size = System.Conve if(Size &gt; 0)</pre>	Table 3 Data ert.ToInt32(S.Trim());		<pre>sr.Read(Data, 0 for(int i = 0;     txtHuffman4.T     Data = null; }</pre>	), Size); i < Data.Length; i++) ?ext += Data[i].ToString();	
<pre>{     lblHuffman3.Text     Data = new char     sr.Read(Data, 0,     for(int i = 0; i         lblHuffman3.Tex     Data = null;     }     S = sr.ReadLine();     Size = System.Convolution </pre>	<pre>= ""; [Size]; Size); &lt; Data.Length; i++) xt += Data[i].ToString(); ert.ToInt32(S.Trim());</pre>		<pre>S = sr.ReadLine() Size = System.Con if(Size &gt; 0) {     lblHuffmanOrigi     Data = new char     sr.Read(Data, 0     for(int i = 0;     lblHuffmanOri     Data = null; </pre>	; nal4.Text = ""; [Size]; Size); i < Data.Length; i++) ginal4.Text += Data[i].ToString();	
<pre>if(Size &gt; 0) {     txtHuffman3.Text     Data = new char     sr.Read(Data, 0,     for(int i = 0; i         txtHuffman3.Te:     Data = null; } S = sr.ReadLine();</pre>	<pre>= ""; [Size]; Size); &lt; Data.Length; i++) xt += Data[i].ToString();</pre>		<pre>} S = sr.ReadLine() Size = System.Con if(Size &gt; 0) {     txtHuffmanOrigi     Data = new char     sr.Read(Data, 0     for(int i = 0;         txtHuffmanOri </pre>	; nvert.ToInt32(S.Trim()); nal4.Text = ""; [Size]; , Size); i < Data.Length; i++) ginal4.Text += Data[i].ToString();	
<pre>Size = System.Convo if(Size &gt; 0) { lblHuffmanOrigin. Data = new char sr.Read(Data, 0, for(int i = 0; i lblHuffmanOrig. Data = null; }</pre>	<pre>ert.ToInt32(S.Trim()); al3.Text = ""; [Size]; Size); &lt; Data.Length; i++) inal3.Text += Data[i].ToString();</pre>		<pre>Data = null; } // Get Compressic S = sr.ReadLine() Size = System.Com if(Size &gt; 0) {     lblHuffman5.Tex     Data = new char     sr.Read(Data, 0)</pre>	<pre>on Table 5 Data ; vert.ToInt32(S.Trim()); tt = ""; [Size]; , Size);</pre>	
<pre>S = sr.ReadLine(); Size = System.Conve if(Size &gt; 0) {</pre>	ert.ToInt32(S.Trim());		<pre>for(int i = 0; lblHuffman5.T Data = null; }</pre>	<pre>i &lt; Data.Length; i++) 'ext += Data[i].ToString();</pre>	
<pre>txtHurimanOrigin Data = new char sr.Read(Data, 0, for(int i = 0; i txtHuffmanOrig: Data = null; } // Get Compression</pre>	<pre>als.lext = ""; [Size]; Size); &lt; Data.Length; i++) inal3.Text += Data[i].ToString(); Table 4 Data</pre>		<pre>S = sr.ReadLine() Size = System.Com if(Size &gt; 0) {     txtHuffman5.Tex     Data = new char     sr.Read(Data, 0     for(int i = 0:</pre>	; ivert.ToInt32(S.Trim()); it = ""; : [Size]; ), Size); i < Data.Length; i++)	
S = sr.ReadLine(); Size = System.Conv if(Size > 0)	<pre>ert.ToInt32(S.Trim());</pre>		txtHuffman5.T Data = null; }	<pre>iext += Data[i].ToString();</pre>	
<pre>lblHuffman4.Text Data = new char sr.Read(Data, 0, for(int i = 0; i lblHuffman4.Te: Data = null; }</pre>	<pre>= ""; [Size]; Size); &lt; Data.Length; i++) xt += Data[i].ToString();</pre>		<pre>S = sr.ReadLine() Size = System.Con if(Size &gt; 0) {     lblHuffmanOrigi     Data = new char     sr.Read(Data, 0)</pre>	; wert.ToInt32(S.Trim()); nal5.Text = ""; [Size]; Size);	

```
frmMain.cs
May 02, 04 2:03
                                                                     Page 53/186
         for(int i = 0; i < Data.Length; i++)</pre>
           lblHuffmanOriginal5.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtHuffmanOriginal5.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
          txtHuffmanOriginal5.Text += Data[i].ToString();
         Data = null;
       // Get Compression Table 6 Data
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         lblHuffman6.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
          lblHuffman6.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtHuffman6.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
          txtHuffman6.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         lblHuffmanOriginal6.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
          lblHuffmanOriginal6.Text += Data[i].ToString();
         Data = null;
      S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtHuffmanOriginal6.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
          txtHuffmanOriginal6.Text += Data[i].ToString();
         Data = null;
       // Get Compression Table 7 Data
       S = sr.ReadLine();
      Size = System.Convert.ToInt32(S.Trim());
```

```
frmMain.cs
May 02, 04 2:03
                                                                       Page 54/186
       if(Size > 0)
         lblHuffman7.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           lblHuffman7.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtHuffman7.Text = "";
         Data = new char [Size];
        sr.Read(Data, 0, Size);
for(int i = 0; i < Data.Length; i++)</pre>
          txtHuffman7.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         lblHuffmanOriginal7.Text = "";
         Data = new char [Size];
        sr.Read(Data, 0, Size);
for(int i = 0; i < Data.Length; i++)</pre>
           lblHuffmanOriginal7.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtHuffmanOriginal7.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           txtHuffmanOriginal7.Text += Data[i].ToString();
         Data = null;
       1
       // Get Compression Table 8 Data
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         lblHuffman8.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           lblHuffman8.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtHuffman8.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           txtHuffman8.Text += Data[i].ToString();
         Data = null;
```

frmMain.cs May 02, 04 2:03 Page 55/186 S = sr.ReadLine(); Size = System.Convert.ToInt32(S.Trim()); if(Size > 0)lblHuffmanOriginal8.Text = ""; Data = new char [Size]; sr.Read(Data, 0, Size); for(int i = 0; i < Data.Length; i++)</pre> lblHuffmanOriginal8.Text += Data[i].ToString(); Data = null; S = sr.ReadLine(); Size = System.Convert.ToInt32(S.Trim()); if(Size > 0)txtHuffmanOriginal8.Text = ""; Data = new char [Size]; sr.Read(Data, 0, Size); for(int i = 0; i < Data.Length; i++)</pre> txtHuffmanOriginal8.Text += Data[i].ToString(); Data = null; // Quantizer Table Data 11 // Get the Quantizer Table 1 Data S = sr.ReadLine(); Size = System.Convert.ToInt32(S.Trim()); if(Size > 0)lblQuantizerMarker1.Text = ""; Data = new char [Size]; sr.Read(Data, 0, Size); for(int i = 0; i < Data.Length; i++)</pre> lblQuantizerMarker1.Text += Data[i].ToString(); Data = null; 1 S = sr.ReadLine(); Size = System.Convert.ToInt32(S.Trim()); if(Size > 0)txtQuantizerTableNum1.Text = ""; Data = new char [Size]; sr.Read(Data, 0, Size); for(int i = 0; i < Data.Length; i++)</pre> txtQuantizerTableNum1.Text += Data[i].ToString(); Data = null; } S = sr.ReadLine(); Size = System.Convert.ToInt32(S.Trim()); if(Size > 0)txtQuantizer1.Text = ""; Data = new char [Size]; sr.Read(Data, 0, Size); for(int i = 0; i < Data.Length; i++)</pre> txtQuantizer1.Text += Data[i].ToString(); Data = null; S = sr.ReadLine(); Size = System.Convert.ToInt32(S.Trim());

```
frmMain.cs
May 02, 04 2:03
                                                                       Page 56/186
       if(Size > 0)
         lblQuantizerOriginalMarker1.Text = "";
         Data = new char [Size];
        sr.Read(Data, 0, Size);
for(int i = 0; i < Data.Length; i++)</pre>
           lblQuantizerOriginalMarker1.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtQuantizerOriginal1.Text = "";
         Data = new char [Size];
        sr.Read(Data, 0, Size);
for(int i = 0; i < Data.Length; i++)</pre>
           txtQuantizerOriginal1.Text += Data[i].ToString();
         Data = null;
       // Get the Quantizer Table 2 Data
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         lblQuantizerMarker2.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           lblQuantizerMarker2.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtQuantizerTableNum2.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           txtQuantizerTableNum2.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         txtQuantizer2.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           txtQuantizer2.Text += Data[i].ToString();
         Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
         lblQuantizerOriginalMarker2.Text = "";
         Data = new char [Size];
         sr.Read(Data, 0, Size);
         for(int i = 0; i < Data.Length; i++)</pre>
           lblQuantizerOriginalMarker2.Text += Data[i].ToString();
         Data = null;
```
May 02, 04 2:03	frmMain.cs	Page 57/186	May 02, 04 2:03	frmMain.cs	Page 58/186
}			sr.Read(Data,	0, Size);	
S = sr.ReadLine() Size = System.Con if(Size > 0)	; vert.ToInt32(S.Trim());		for(int i = 0; txtQuantizer( Data = null; }	<pre>i &lt; Data.Length; i++) Original3.Text += Data[i].ToString();</pre>	
<pre>{    txtQuantizerOri    Data = new char    sr.Read(Data, 0    for(int i = 0;       txtQuantizerO </pre>	ginal2.Text = ""; [Size]; (, Size); i < Data.Length; i++) priginal2.Text += Data[i].ToString();		<pre>// Get the Quant. S = sr.ReadLine( Size = System.Con if(Size &gt; 0) {</pre>	<pre>izer Table 4 Data ); nvert.ToInt32(S.Trim());</pre>	
Data = null; }			lblQuantizerMa: Data = new cha: sr.Read(Data,	rker4.Text = ""; r [Size]; 0, Size);	
<pre>// Get the Quanti S = sr.ReadLine() Size = System.Con if(Size &gt; 0)</pre>	<pre>zer Table 3 Data ; vert.ToInt32(S.Trim());</pre>		<pre>for(int i = 0;</pre>	i < Data.Length; i++) Marker4.Text += Data[i].ToString();	
<pre>{     lblQuantizerMar     Data = new char     sr.Read(Data, 0     for(int i = 0;         lblQuantizerM     Data = null;     }     S = sr.ReadLine()     Size = System.Con </pre>	<pre>ker3.Text = ""; [Size]; , Size); i &lt; Data.Length; i++) larker3.Text += Data[i].ToString(); ; ; vert.ToInt32(S.Trim());</pre>		<pre>S = sr.ReadLine( Size = System.Con if(Size &gt; 0) { txtQuantizerTal Data = new chai sr.Read(Data, for for(int i = 0; txtQuantizer Data = null;</pre>	); nvert.ToInt32(S.Trim()); bleNum4.Text = ""; r [Size]; 0, Size); i < Data.Length; i++) TableNum4.Text += Data[i].ToString();	
if(Size > 0) {			}		
<pre>txtQuantizerTab Data = new char sr.Read(Data, 0 for(int i = 0; txtQuantizerT Data = null; } S = sr.ReadLine() Size = System.Con if(Size &gt; 0)</pre>	<pre>bleNum3.Text = ""; [Size]; (, Size); i &lt; Data.Length; i++) ableNum3.Text += Data[i].ToString(); ; ; vert.ToInt32(S.Trim());</pre>		<pre>S = sr.ReadLine( Size = System.Com if(Size &gt; 0) {     txtQuantizer4.'     Data = new chan     sr.Read(Data, for(int i = 0;         txtQuantizer-         Data = null;     } }</pre>	); nvert.ToInt32(S.Trim()); Text = ""; r [Size]; 0, Size); i < Data.Length; i++) 4.Text += Data[i].ToString();	
<pre>{     txtQuantizer3.T     Data = new char, 0     for(int i = 0;         txtQuantizer3     Data = null;     }     S = sr.ReadLine()     Size = System.Con     if(Size &gt; 0)</pre>	<pre>Text = ""; [Size]; i &lt; Data.Length; i++) .Text += Data[i].ToString(); ; vert.ToInt32(S.Trim());</pre>		<pre>S = sr.ReadLine( Size = System.Con if(Size &gt; 0) { lblQuantizerOr. Data = new cha: sr.Read(Data, - for(int i = 0; lblQuantizer( Data = null; } </pre>	); nvert.ToInt32(S.Trim()); iginalMarker4.Text = ""; r [Size]; 0, Size); i < Data.Length; i++) OriginalMarker4.Text += Data[i].ToString	(();
<pre>{     lblQuantizerOri     Data = new char     sr.Read(Data, 0     for(int i = 0;     lblQuantizerO     Data = null; } S = sr.ReadLine() Size = System.Con if(Size &gt; 0)</pre>	<pre>ginalMarker3.Text = ""; [Size]; , Size); i &lt; Data.Length; i++) riginalMarker3.Text += Data[i].ToString ; wert.ToInt32(S.Trim());</pre>	();	<pre>S = sr.ReadLine( Size = System.Con if(Size &gt; 0) { txtQuantizerOr. Data = new cha: sr.Read(Data, for(int i = 0; txtQuantizerO Data = null; } </pre>	); nvert.ToInt32(S.Trim()); iginal4.Text = ""; r [Size]; 0, Size); i < Data.Length; i++) Original4.Text += Data[i].ToString();	
{ txtQuantizerOri Data = new char	ginal3.Text = ""; [Size];		// // Application Da	ata	

02, 04 2:03	frmMain.cs	Page 59/186
//		
<pre>// Get the Application S = sr.ReadLine(); Size = System.Convert if(Size &gt; 0)</pre>	n Data 1 .ToInt32(S.Trim());	
<pre>lblApplicationMarker Data = new char [Si: sr.Read(Data, 0, Si: for(int i = 0; i &lt; 1 lblApplicationMar) Data = null; }</pre>	rl.Text = ""; ze]; ze); Data.Length; i++) kerl.Text += Data[i].ToString();	
<pre>S = sr.ReadLine(); Size = System.Convert if(Size &gt; 0)</pre>	.ToInt32(S.Trim());	
<pre>{ txtApplicationData1 Data = new char [Si: sr.Read(Data, 0, Si: for(int i = 0; i &lt; 1 txtApplicationData Data = null; } </pre>	.Text = ""; ze]; ze); Data.Length; i++) al.Text += Data[i].ToString();	
<pre>// Get the Application S = sr.ReadLine(); Size = System.Convert if(Size &gt; 0) </pre>	n Data 2 .ToInt32(S.Trim());	
<pre>lblApplicationMarkes Data = new char [Si: sr.Read(Data, 0, Si: for(int i = 0; i &lt; 1 lblApplicationMar) Data = null; }</pre>	r2.Text = ""; ze]; ze); Data.Length; i++) ker2.Text += Data[i].ToString();	
<pre>S = sr.ReadLine(); Size = System.Convert if(Size &gt; 0)</pre>	.ToInt32(S.Trim());	
<pre>txtApplicationData2 Data = new char [Si: sr.Read(Data, 0, Si: for(int i = 0; i &lt; 1 txtApplicationData Data = null; }</pre>	.Text = ""; ze]; ze); Data.Length; i++) a2.Text += Data[i].ToString();	
<pre>// Get the Application S = sr.ReadLine(); Size = System.Convert if(Size &gt; 0) </pre>	n Data 3 .ToInt32(S.Trim());	
<pre>lblApplicationMarkey Data = new char [Si: sr.Read(Data, 0, Si: for(int i = 0; i &lt; 1 lblApplicationMarkey Data = null;</pre>	r3.Text = ""; ze]; ze); Data.Length; i++) ker3.Text += Data[i].ToString();	
}		
<pre>S = sr.ReadLine(); Size = System.Convert if(Size &gt; 0)</pre>	.ToInt32(S.Trim());	

```
frmMain.cs
May 02, 04 2:03
                                                                    Page 60/186
        txtApplicationData3.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          txtApplicationData3.Text += Data[i].ToString();
        Data = null;
       // Get the Application Data 4
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        lblApplicationMarker4.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          lblApplicationMarker4.Text += Data[i].ToString();
        Data = null;
       }
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        txtApplicationData4.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          txtApplicationData4.Text += Data[i].ToString();
        Data = null;
       // Get the Application Data 5
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        lblApplicationMarker5.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          lblApplicationMarker5.Text += Data[i].ToString();
        Data = null;
       }
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        txtApplicationData5.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          txtApplicationData5.Text += Data[i].ToString();
        Data = null;
       }
       // Get the Application Data 6
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        lblApplicationMarker6.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          lblApplicationMarker6.Text += Data[i].ToString();
        Data = null;
```

M

2, 04 2:03	frmMain.cs	Page 61/186
}		
<pre>S = sr.ReadLine(); Size = System.Conver if(Size &gt; 0) {</pre>	rt.ToInt32(S.Trim());	
<pre>txtApplicationDat. Data = new char [ sr.Read(Data, 0, for(int i = 0; i txtApplicationD. Data = null; }</pre>	a6.Text = ""; Size]; Size); < Data.Length; i++) ata6.Text += Data[i].ToString();	
<pre>// Get the Applicat S = sr.ReadLine(); Size = System.Conver if(Size &gt; 0) </pre>	ion Data 7 rt.ToInt32(S.Trim());	
<pre>{     lblApplicationMar     Data = new char [     sr.Read(Data, 0,     for(int i = 0; i         lblApplicationM     Data = null; }</pre>	<pre>ker7.Text = ""; Size]; Size); &lt; Data.Length; i++) arker7.Text += Data[i].ToString();</pre>	
<pre>S = sr.ReadLine(); Size = System.Conver if(Size &gt; 0) {</pre>	rt.ToInt32(S.Trim());	
<pre>txtApplicationDat. Data = new char [ sr.Read(Data, 0, for(int i = 0; i txtApplicationData = null;</pre>	a7.Text = ""; Size]; Size); < Data.Length; i++) ata7.Text += Data[i].ToString();	
<pre>// Get the Applicat S = sr.ReadLine(); Size = System.Conver if(Size &gt; 0)</pre>	ion Data 8 rt.ToInt32(S.Trim());	
<pre>{     lblApplicationMar     Data = new char [     sr.Read(Data, 0,     for(int i = 0; i         lblApplicationM         Data = null; }</pre>	<pre>ker8.Text = ""; Size]; Size); &lt; Data.Length; i++) arker8.Text += Data[i].ToString();</pre>	
<pre>S = sr.ReadLine(); Size = System.Conver if(Size &gt; 0) {</pre>	rt.ToInt32(S.Trim());	
<pre>txtApplicationDat. Data = new char [ sr.Read(Data, 0, for(int i = 0; i txtApplicationD Data = null;</pre>	a8.Text = ""; Size]; Size); < Data.Length; i++) ata8.Text += Data[i].ToString();	
1		
<pre>// Get the Applicat S = sr.ReadLine(); Size = System.Conver if(Size &gt; 0)</pre>	ion Data 9 rt.ToInt32(S.Trim());	

```
frmMain.cs
May 02, 04 2:03
                                                                    Page 62/186
         lblApplicationMarker9.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          lblApplicationMarker9.Text += Data[i].ToString();
        Data = null;
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        txtApplicationData9.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          txtApplicationData9.Text += Data[i].ToString();
        Data = null;
       }
       // Get the Application Data 10
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        lblApplicationMarker10.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          lblApplicationMarker10.Text += Data[i].ToString();
        Data = null;
       }
      S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        txtApplicationData10.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          txtApplicationData10.Text += Data[i].ToString();
        Data = null;
       }
       // Misc Tab Data
       11
       // Get the Restart Marker Data
      S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        lblRestartMarker.Text = "";
        Data = new char [Size];
        sr.Read(Data, 0, Size);
        for(int i = 0; i < Data.Length; i++)</pre>
          lblRestartMarker.Text += Data[i].ToString();
        Data = null;
       }
       S = sr.ReadLine();
       Size = System.Convert.ToInt32(S.Trim());
       if(Size > 0)
        txtRestart.Text = "";
        Data = new char [Size];
```

sr.Read(Data, 0, Size);

May 02, 04 2:03	frmMain.cs	Page 63/186	May 02, 04 2:03	frmMain.cs	Page 64/186
for(int $i = 0$ ;	; i < Data.Length; i++)		S = sr.ReadLine()	);	
txtRestart.1	<pre>fext += Data[i].ToString();</pre>		Size = System.Cor	nvert.ToInt32(S.Trim());	
}			{		
			lblHierarchialN	<pre>farker.Text = "";</pre>	
// Get the Number S = sr.ReadLine(	er of Lines Marker Data ():		Data = new char sr.Read(Data, (	r [Size]; ). Size):	
Size = System.Co	<pre>onvert.ToInt32(S.Trim());</pre>		for (int $i = 0$ ;	i < Data.Length; i++)	
if(Size > 0)			lblHierarchia	alMarker.Text += Data[i].ToString();	
lblNumberLines	sMarker.Text = "";		}		
Data = new cha	ar [Size];				
sr.Read(Data, for(int i = 0)	U, Size); · i < Data Length· i++)		S = sr.ReadLine() Size = System Cor	); nvert ToInt32(S Trim()).	
lblNumberLir	hesMarker.Text += Data[i].ToString();		if(Size > 0)		
Data = null;			{ tytHierarchial	Text - "".	
J			Data = new char	[Size];	
S = sr.ReadLine	();		sr.Read(Data, (	), Size);	
size = System.Co	<pre>onvert.ToInt32(S.Trim());</pre>		<pre>ior(int i = 0;     txtHierarchia</pre>	1 < Data.Lengtn; 1++) al.Text += Data[i].ToString():	
{			Data = null;	baca(1).100011ng(//	
txtNumberLines	s.Text = "";		}		
sr.Read(Data,	0, Size);		// Get the Error	Data	
for(int $i = 0$ ;	; i < Data.Length; i++)		<pre>S = sr.ReadLine()</pre>	);	
txtNumberLir	<pre>nes.Text += Data[i].ToString();</pre>		Size = System.Cor	nvert.ToInt32(S.Trim());	
}			{		
	d Maulau Data		txtError.Text =	= "";	
S = sr.ReadLine	();		sr.Read(Data, (	(Size);	
Size = System.Co	<pre>onvert.ToInt32(S.Trim());</pre>		for (int $i = 0;$	i < Data.Length; i++)	
if(Size > 0)			txtError.Text	<pre>t += Data[i].ToString();</pre>	
lblExpandMarke	er.Text = "";		}		
Data = new cha	ar [Size];				
for (int $i = 0$ ;	; i < Data.Length; i++)		// Close the Stre	eam Reader	
lblExpandMar	<pre>rker.Text += Data[i].ToString();</pre>		<pre>sr.Close();</pre>		
Data = null;			} // End of: try b]	lock	
			catch (Exception ex)	)	
S = sr.ReadLine(	(); prvert ToInt32(S Trim());		{ if(ex Message	"Invalid parameter used "	
if(Size > 0)	5		ex.Message == '	"A generic error occurred in GDI+."	
{			ex.Source == "S	System.Drawing")	
Data = new cha	; ar [Size];		string x = Proc	gramDirectory + @"\default bad.jpg";	
sr.Read(Data,	0, Size);		LoadPicture(x,	x);	
for(int i = 0;	; 1 < Data.Length; i++)		} else		
Data = null;	2404(1).100011Hg(//		{		
}			ShowWarning(	exception accured.\n\n" +	
// Get the Resta	art Mod 8 Data		"Exception Er	cror:\n" +	
S = sr.ReadLine	();		ex.Message +	"\n\nWas throw by:\n" +	
Size = System.Co	<pre>onvert.ToInt32(S.Trim());</pre>		ex.Source + "\n\nNot all	load operations completed.!".	
{			"Load File Ex	<pre>xception");</pre>	
txtRestartMod8	B.Text = "";		ClearInterface	Data();	
sr.Read(Data,	0, Size);		}		
for (int $i = 0$ ;	; i < Data.Length; i++)		}		
txtRestartMc	<pre>od8.Text += Data[i].ToString();</pre>				
}			/// <summary></summary>		
			/// Pre-conditions:	None.	
// Get the Hiera	archical Data		/// Post-conditions:		

```
frmMain.cs
May 02, 04 2:03
                                                                  Page 65/186
  111
         All of the current values loaded in the Manipulator, any project
  111
         notes and current image file names have been saved in a SEP
  ///
         project file name based upon the file name string in the
  111
         txtProjectPath TextBox control.
  111
       Description:
  111
         The purpose of this method is to allow the caller to save an SEP
  111
         project file based upon the current values loaded in the
  ///
         interface of the Manipulator. The data saved should include both
  111
         the file name and paths of the images currently loaded within the
  111
         Manipulator and all of the data in the TextBox controls on the
  111
         sub-tabs located under the Console tab, including the txtNotes
  111
         control for the project notes. The project name should be the
  111
         file name and path stored in the txtProjectPath TextBox control.
  111
         If a file with this name already exists, the user should be asked
  ///
         if it is okay to overwrite the pre-existing project file. Lastly,
  111
         this method should do some error checking to make sure this
  111
         function executes properly. If an error is encountered, then the
  111
         ShowWarning() method should be called to display the error to the
  111
         user and the txtError TextBox control should be updated with this
  111
         error information.
  /// </summary>
  private void SaveNewProject()
    // Check to make sure a JPEG is loaded.
    if(txtOriginalFile.Text == "" || !File.Exists(txtOriginalFile.Text))
      ShowWarning(
        "There is NO JPEG file currently loaded!\n" +
         "Project WILL NOT be saved!",
        "Save Project Canceled");
      return;
    }
    // Show the save dialog box
    saveFileDialog1.ShowHelp = false;
    if(saveFileDialog1.ShowDialog() != DialogResult.OK) return;
    // Show warning if file already exists
    // If the users chooses OK, we'll overwrite the file.
    while(File.Exists(saveFileDialog1.FileName.Trim()))
      if(!ShowWarning(
        "Project ALREADY exists!!\n\n" + saveFileDialog1.FileName +
        "\n\nWould you like to overwrite this file?",
        "Project File Already Exists!"))
        if(saveFileDialog1.ShowDialog() != DialogResult.OK) return;
      else break;
    txtProjectPath.Text = saveFileDialog1.FileName.Trim();
    if(File.Exists(txtProjectPath.Text)) File.Delete(txtProjectPath.Text);
    try
      // Create a file to write to
      StreamWriter sr:
      int Size;
      StringBuilder ProjData = new
        StringBuilder(AVE_FILE_SIZE, MAX_FILE_SIZE);
      // Get all the data from the Manipulator in a String for Conversion
      // Write size of the Notes and then the Data
      Size = txtNotes.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtNotes.Text.TrimEnd());
```

```
frmMain.cs
May 02, 04 2:03
                                                                   Page 66/186
      // File Tab Data
      11
      // Write the Original Picture path
      Size = txtOriginalFile.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtOriginalFile.Text.TrimEnd());
      // Write the Manipulated Picture path
      if(File.Exists(txtManipulatedFile.Text.TrimEnd()))
        Size = txtManipulatedFile.Text.TrimEnd().Length;
        ProjData.Append(Size.ToString() + "\n");
        if (Size > 0) ProjData.Append(txtManipulatedFile.Text.TrimEnd());
      else
        Size = txtOriginalFile.Text.TrimEnd().Length;
        ProjData.Append(Size.ToString() + "\n");
        if (Size > 0) ProjData.Append(txtOriginalFile.Text.TrimEnd());
      // Write the File Size data
      Size = txtFileSize.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtFileSize.Text.TrimEnd());
      // Write the Comments
      Size = txtComments.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtComments.Text.TrimEnd());
      // Header Tab Data
      11
      // Write the Start of Compression Marker
      Size = txtStartHuffman.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if(Size > 0) ProjData.Append(txtStartHuffman.Text.TrimEnd());
      // Write the Start of Compression Header Size
      Size = txtStartHuffmanSize.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtStartHuffmanSize.Text.TrimEnd());
      // Write the Precision
      Size = txtPrecision.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtPrecision.Text.TrimEnd());
      // Write the Huffman Lines
      Size = txtNumberHuffmanLines.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtNumberHuffmanLines.Text.TrimEnd());
      // Write the Huffman Samples
      Size = txtNumberHuffmanSamples.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if(Size > 0) ProjData.Append(txtNumberHuffmanSamples.Text.TrimEnd());
      // Write the Number of Image Components
      Size = txtNumberImageComponents.Text.TrimEnd().Length;
      ProjData.Append(Size.ToString() + "\n");
      if (Size > 0) ProjData.Append(txtNumberImageComponents.Text.TrimEnd());
```

// Write the Number of Components

May 02, 04 2:03	frmMain.cs	Page 67/186	May 02, 04 2:03	frmMain.cs	Page 68/186
Size = txtCompon	ents.Text.TrimEnd().Length;	_	ProjData Append	(Size.ToString() + "\n");	
ProjData.Append( if(Size > 0) Pro	Size.ToString() + "\n"); jData.Append(txtComponents.Text.T	rimEnd());	if(Size > 0) Pr	ojData.Append(lblHuffmanOriginal4.T	<pre>ext.TrimEnd());</pre>
	J	- () / /	Size = txtHuffm	anOriginal4.Text.TrimEnd().Length;	
// // Huffman Table //	Data		ProjData.Append if(Size > 0) Pr	<pre>(Size.ToString() + "\n"); ojData.Append(txtHuffmanOriginal4.To)</pre>	ext.TrimEnd());
//			// Write Compre	ssion Table 5 Data	
// Write Compres	sion Table 1 Data		Size = lblHuffm	an5.Text.TrimEnd().Length;	
Size = lblHuffma	nl.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append( if(Size > 0) Pro	<pre>Size.ToString() + "\n"); jData.Append(lblHuffman1.Text.Tri</pre>	mEnd());	if(Size > 0) Pr	ojData.Append(lblHuffman5.Text.Trim	End());
			Size = txtHuffm	an5.Text.TrimEnd().Length;	
Size = txtHuIIma:	nl.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	End()).
if(Size > 0) Pro	iData.Append(txtHuffmanl.Text.Tri	mEnd()):	II(512e > 0) PI	ojbaca.Append(cxchurrmans.rext.rrrm	2110());
11(0120 > 0) 110			Size = lblHuffm	anOriginal5.Text.TrimEnd().Length;	
Size = lblHuffma:	nOriginal1.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append(	<pre>Size.ToString() + "\n");</pre>		if(Size > 0) Pr	ojData.Append(lblHuffmanOriginal5.Te	ext.TrimEnd());
if(Size > 0) Pro	jData.Append(lblHuffmanOriginal1.	Text.TrimEnd());			
			Size = txtHuffm	anOriginal5.Text.TrimEnd().Length;	
Size = txtHuIIma:	<pre>nOriginall.Text.TrimEnd().Length; Size ToString() + "\n");</pre>		ProjData.Append	(Size.ToString() + "\n");	out TrimEnd()).
if(Size > 0) Pro	iData Append(tytHuffmanOriginal)	Text TrimEnd()).	II(SIZE > 0) PI	ojbaca.Append(cxchurrmanoriginars.io	ext. If InEnd());
11(5120 > 0) 110	Jbaca.Appena (excharimanoriginari.	iexe. ii imbild () / ,	// Write Compre	ssion Table 6 Data	
// Write Compres	sion Table 2 Data		Size = lblHuffm	an6.Text.TrimEnd().Length;	
Size = lblHuffma	n2.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append(	Size.ToString() + "\n");		if(Size > 0) Pr	ojData.Append(lblHuffman6.Text.Trim	End());
if(Size > 0) Pro	jData.Append(lblHuffman2.Text.Tri	mEnd());			
Ciao tutluffmo	n2 Tout TwimEnd() Longth.		Size = txtHuffm	an6.Text.TrimEnd().Length;	
ProjData Append(	Size ToString() + "\n").		if (Size > 0) Pr	(SIZE.IOSLIING() + "(N"); ojData Append(tytHuffman6 Teyt Trim	End() ·
if(Size > 0) Pro	iData.Append(txtHuffman2.Text.Tri	mEnd()):	11(5120 > 0) 11		
( ,,,,	J		Size = lblHuffm	anOriginal6.Text.TrimEnd().Length;	
Size = lblHuffma	nOriginal2.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append(	Size.ToString() + "\n");		if(Size > 0) Pr	ojData.Append(lblHuffmanOriginal6.Te	ext.TrimEnd());
if(Size > 0) Pro	jData.Append(lblHuffmanOriginal2.	Text.TrimEnd());			
Sizo - tytUuffma	nOriginal? Toxt TrimEnd() Longth.		Size = txtHuIIm ProjData Appond	anoriginal6.lext.lrimEnd().Lengtn;	
ProjData Append(	Size.ToString() + "\n"):		if(Size > 0) Pr	oiData.Append(txtHuffmanOriginal6.Te	ext.TrimEnd()):
if(Size > 0) Pro	jData.Append(txtHuffmanOriginal2.	Text.TrimEnd());		- J	
			// Write Compre	ssion Table 7 Data	
// Write Compres	sion Table 3 Data		Size = lblHuffm	an7.Text.TrimEnd().Length;	
Size = lblHuffma	n3.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append(	Size.ToString() + "\n");	mEnd()).	if(Size > 0) Pr	ojData.Append(lblHuffman/.Text.Trim	End());
11(S12e > 0) Pro	JData.Append(IDIHUIIMans.lext.III	mena());	Size - tytHuffm	an7 Text TrimEnd() Length.	
Size = txtHuffma	n3.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append(	Size.ToString() + "\n");		if(Size > 0) Pr	ojData.Append(txtHuffman7.Text.Trim	End());
if(Size > 0) Pro	jData.Append(txtHuffman3.Text.Tri	mEnd());			
al			Size = lblHuffm	anOriginal7.Text.TrimEnd().Length;	
Size = lblHuffma: DrojData Appond(	nOriginal3.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	out TrimEnd()).
if (Size > 0) Pro	iData Append(lblHuffmanOriginal3	Text TrimEnd()).	II (SIZE > 0) PI	ojbaca.Append(ibiHulimanoriginal/.i	ext.IfImEnd());
11(5120 > 0) 110	Jbaca.Appena(ibinatimanoriginais.	iexe.iiiiiiiii()),	Size = txtHuffm	anOriginal7.Text.TrimEnd().Length:	
Size = txtHuffma:	nOriginal3.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append(	Size.ToString() + "\n");		if(Size > 0) Pr	ojData.Append(txtHuffmanOriginal7.Te	ext.TrimEnd());
if(Size > 0) Pro	jData.Append(txtHuffmanOriginal3.	Text.TrimEnd());			
			// Write Compre	ssion Table 8 Data	
// Write Compres	sion lable 4 Data		Size = iblHuffm	ano.lext.Trimend().Length;	
ProjData Append(	Size.ToString() + "\n"):		if(Size > 0) Pr	oiData.Append(lblHuffman8.Text_Trim	End()):
if(Size > 0) Pro	jData.Append(lblHuffman4.Text.Tri	mEnd());	11(0120 > 0) 11	o joacappena (ioinarimano.iext.iiim	
	5 11- ,		Size = txtHuffm	an8.Text.TrimEnd().Length;	
Size = txtHuffma:	n4.Text.TrimEnd().Length;		ProjData.Append	(Size.ToString() + "\n");	
ProjData.Append(	<pre>Size.ToString() + "\n");</pre>		if(Size > 0) Pr	ojData.Append(txtHuffman8.Text.Trim	End());
if(Size > 0) Pro	JData.Append(txtHuffman4.Text.Tri	mEnd());	Cigo lblu-ff-	anOviginal Post ThimEnd() Incoth	
Size = lblHuffma	nOriginal4.Text.TrimEnd().Length.		ProiData Append	(Size.ToString() + "\n"):	
ioinattilla	, <u>, , , , , , , , , , , , , , , , , , </u>		pouca.mppona	· · · · · · · · · · · · · · · · · · ·	

May (	02, 04 2:03	frmMain.cs	Page 69/186	May (	02, 04 2:03	frmMain.cs	Page 70/186
	if(Size > 0) ProjData.	Append(lblHuffmanOriginal8.Te	<pre>xt.TrimEnd());</pre>		Size = lblQuan ProjData.Appen	tizerOriginalMarker3.Text.TrimEnd().Le d(Size.ToString() + "\n");	ength;
	Size = txtHuffmanOrigi ProjData.Append(Size.T if(Size > 0) ProjData.	<pre>nal8.Text.TrimEnd().Length; oString() + "\n"); Append(txtHuffmanOriginal8.Text Append(txtHuffmanOriginal8.Text Append(txtHuffmanOriginal8.Text Append(txtHuffmanOriginal8.Text Append(txtHuffmanOriginal8.Text)</pre>	<pre>&gt;xt.TrimEnd());</pre>	());	if(Size > 0)	ProjData.Append(lblQuantizerOriginal	lMarker3.Text.TrimEnd
	// // Quantizer Table Dat //	a			Size = txtQuan ProjData.Appen if(Size > 0) P	tizerOriginal3.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(txtQuantizerOriginal3.T	<pre>Fext.TrimEnd());</pre>
	<pre>// Write the Quantizer Size = lblQuantizerMar ProjData.Append(Size.T if(Size &gt; 0) ProjData.</pre>	Table 1 Data kerl.Text.TrimEnd().Length; oString() + "\n"); Append(lblQuantizerMarkerl.Te	ext.TrimEnd());		// Write the Q Size = lblQuan ProjData.Appen if(Size > 0) P	uantizer Table 4 Data tizerMarker4.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(lblQuantizerMarker4.Te;	<pre>xt.TrimEnd());</pre>
	Size = txtQuantizerTab ProjData.Append(Size.T if(Size > 0) ProjData.	<pre>leNum1.Text.TrimEnd().Length; oString() + "\n"); Append(txtQuantizerTableNum1.</pre>	Text.TrimEnd());		Size = txtQuan ProjData.Appen if(Size > 0) P	tizerTableNum4.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(txtQuantizerTableNum4.T	<pre>Iext.TrimEnd());</pre>
	Size = txtQuantizer1.T ProjData.Append(Size.T if(Size > 0) ProjData.	ext.TrimEnd().Length; oString() + "\n"); Append(txtQuantizer1.Text.Tr:	mEnd());		Size = txtQuan ProjData.Appen if(Size > 0) P	tizer4.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(txtQuantizer4.Text.Trim	nEnd());
	Size = lblQuantizerOri ProjData.Append(Size.T if(Size > 0)	<pre>ginalMarker1.Text.TrimEnd().I oString() + "\n");</pre>	.ength;		Size = lblQuan ProjData.Appen if(Size > 0)	<pre>tizerOriginalMarker4.Text.TrimEnd().Le d(Size.ToString() + "\n");</pre>	ength;
());	ProjDa	ta.Append(lblQuantizerOrigina	alMarker1.Text.TrimEnd	());		ProjData.Append(lblQuantizerOrigina)	LMarker4.Text.TrimEnd
	Size = txtQuantizerOri ProjData.Append(Size.T if(Size > 0) ProjData.	<pre>ginal1.Text.TrimEnd().Length; oString() + "\n"); Append(txtQuantizerOriginal1</pre>	Text.TrimEnd());		Size = txtQuan ProjData.Appen if(Size > 0) P	tizerOriginal4.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(txtQuantizerOriginal4.T	<pre>Iext.TrimEnd());</pre>
	<pre>// Write the Quantizer Size = lblQuantizerMar ProjData.Append(Size.T if(Size &gt; 0) ProjData.</pre>	Table 2 Data ker2.Text.TrimEnd().Length; oString() + "\n"); Append(lblOuantizerMarker2.Te	ext.TrimEnd());		// // Application //	Data	
	Size = txtQuantizerTab ProjData.Append(Size.T if(Size > 0) ProjData.	<pre>leNum2.Text.TrimEnd().Length; oString() + "\n"); Append(txtQuantizerTableNum2</pre>	.Text.TrimEnd());		<pre>// Write the A Size = lblAppl ProjData.Appen if(Size &gt; 0) P</pre>	<pre>pplication Data 1 icationMarker1.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(lblApplicationMarker1.T)</pre>	<pre>Iext.TrimEnd());</pre>
	Size = txtQuantizer2.T ProjData.Append(Size.T if(Size > 0) ProjData.	ext.TrimEnd().Length; oString() + "\n"); Append(txtQuantizer2.Text.Tr:	mEnd());		Size = txtAppl ProjData.Appen if(Size > 0) P	icationDatal.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(txtApplicationDatal.Tex	<pre>xt.TrimEnd());</pre>
	Size = lblQuantizerOri ProjData.Append(Size.T if(Size > 0) ProjDa	<pre>ginalMarker2.Text.TrimEnd().I oString() + "\n"); ta_Append(lbl0uantizer0rigin;</pre>	ength;		// Write the A Size = lblAppl ProjData.Appen	<pre>pplication Data 2 icationMarker2.Text.TrimEnd().Length; d(Size.ToString() + "\n"); roiData Append(lblapnlicationMarker2 1</pre>	Text TrimEnd()).
());	Size = txtQuantizerOri ProjData.Append(Size.T	<pre>ginal2.Text.TrimEnd().Length; oString() + "\n");</pre>			Size = txtAppl ProjData.Appen if(Size > 0) P	<pre>icationData2.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(txtApplicationData2.Tex</pre>	<pre>xt.TrimEnd());</pre>
	<pre>// Write the Quantizer Size = lblQuantizerMar ProjData.Append(Size.T if(Size.2)</pre>	<pre>Table 3 Data ker3.Text.TrimEnd().Length; oString() + "\n"); Depend()blouwntiaceMarker2 Text.</pre>	<pre>iext.irimEnd());</pre>		// Write the A Size = lblAppl ProjData.Appen if(Size > 0) P	pplication Data 3 icationMarker3.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(lblApplicationMarker3.T	<pre>Iext.TrimEnd());</pre>
	Size = txtQuantizerTab ProjData.Append(Size.T if(Size > 0) ProjData.	<pre>leNum3.Text.TrimEnd().Length; oString() + "\n"); Append(txtQuantizerTableNum3</pre>	<pre>.rext.TrimEnd());</pre>		Size = txtAppl ProjData.Appen if(Size > 0) P	<pre>icationData3.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(txtApplicationData3.Tex</pre>	<pre>xt.TrimEnd());</pre>
	Size = txtQuantizer3.T ProjData.Append(Size.T if(Size > 0) ProjData.	ext.TrimEnd().Length; oString() + "\n"); Append(txtQuantizer3.Text.Tr:	mEnd());		<pre>// Write the A Size = lblAppl ProjData.Appen if(Size &gt; 0) P</pre>	<pre>pplication Data 4 icationMarker4.Text.TrimEnd().Length; d(Size.ToString() + "\n"); rojData.Append(lblApplicationMarker4.T</pre>	<pre>Fext.TrimEnd());</pre>

May 02, 04 2:03	frmMain.cs	Page 71/186	May 02, 04 2:03	frmMain.cs	Page 72/186
Size = txtApplica	ationData4.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(txtRestart.Text.TrimEnd	d());
ProjData.Append(S	Size.ToString() + "\n");	- TraimEnd())	// Waste the Numb	on of Lince Merken Date	
11(Size > 0) Pro	Juaca. Append (cxtAppircationData4. lex	(, 1 f 1 m E n Q ());	// Write the Numberl	inesMarker Text TrimEnd() Length.	
// Write the App	lication Data 5		ProjData.Append(S	Size.ToString() + "\n");	
Size = lblApplica	ationMarker5.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(lblNumberLinesMarker.Te	ext.TrimEnd());
ProjData.Append(S	<pre>Size.ToString() + "\n");</pre>				
if(Size > 0) Pro	jData.Append(lblApplicationMarker5.Te	ext.TrimEnd());	Size = txtNumberL	ines.Text.TrimEnd().Length;	
Size - tytapplic	ationData5 Text TrimEnd() Length.		if (Size > 0) Proj	Data Append(tytNumberLines Text Tr	imEnd()).
ProiData.Append(S	Size.ToString() + "\n");		11(3120 > 0) 110)	baca.Appena (excitatioer littles, text, it	
if(Size > 0) Pro	jData.Append(txtApplicationData5.Tex	t.TrimEnd());	// Write the Expa	und Marker Data	
			Size = lblExpandM	<pre>larker.Text.TrimEnd().Length;</pre>	
// Write the App	lication Data 6		ProjData.Append(S	Size.ToString() + "\n");	
Size = IblApplica ProjData Appond(S	ationMarker6.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(lblExpandMarker.Text.T	rimend());
if(Size > 0) Pro-	iData.Append(lblApplicationMarker6.Te	ext.TrimEnd()):	Size = $txtExpand$ .	Text.TrimEnd().Length:	
	, , , , , , , , , , , , , , , , , , ,		ProjData.Append(S	Size.ToString() + "\n");	
Size = txtApplica	ationData6.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(txtExpand.Text.TrimEnd	());
ProjData.Append(S	Size.ToString() + "\n");				
if(Size > 0) Pro	JData.Append(txtApplicationData6.Tex	t.TrimEnd());	// Write the Rest	art Mod 8 Data	
// Write the App	lication Data 7		ProjData Append(S	Size ToString() + "\n").	
Size = lblApplica	ationMarker7.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(txtRestartMod8.Text.Tr:	imEnd());
ProjData.Append(S	Size.ToString() + "\n");				
if(Size > 0) Pro	jData.Append(lblApplicationMarker7.T	ext.TrimEnd());	// Write the Hier	archical Data	
			Size = lblHierarc	chialMarker.Text.TrimEnd().Length;	
Size = txtApplica	ationData/.lext.lrimEnd().Lengtn;		if (Size > 0) Proj	Data Appond(lblHiorarchialMarkor T	ovt TrimEnd()).
if(Size > 0) Pro-	iData.Append(txtApplicationData7.Tex	t.TrimEnd()):	11(312e > 0) FIOJ	Data.Append(IDINIerateniaiMarker.10	ext. II India()),
	)		Size = txtHierarc	hial.Text.TrimEnd().Length;	
// Write the App	lication Data 8		ProjData.Append(S	Size.ToString() + "\n");	
Size = lblApplica	ationMarker8.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(txtHierarchial.Text.Tr	imEnd());
ProjData.Append(	<pre>Size.ToString() + "\n"); iData Append(lblApplicationMarker@ T</pre>	out TrimEnd()).	// Write the Erre	Doto	
11(312e > 0) PIO	Jbaca.Append(ibiAppiicacionMarkers.i	ext. II InEnd());	Size = txtError.T	Yext.TrimEnd().Length:	
Size = txtApplica	ationData8.Text.TrimEnd().Length;		ProjData.Append(S	Size.ToString() + "\n");	
ProjData.Append(S	Size.ToString() + "\n");		if(Size > 0) Proj	Data.Append(txtError.Text.TrimEnd(	));
if(Size > 0) Pro	jData.Append(txtApplicationData8.Tex	t.TrimEnd());			
// Write the App	lication Data 9		// // Encoded Data T	ab	
Size = lblApplica	ationMarker9.Text.TrimEnd().Length;		// 111000000 Data 1		
ProjData.Append(S	<pre>Size.ToString() + "\n");</pre>				
if(Size > 0) Pro	jData.Append(lblApplicationMarker9.Te	ext.TrimEnd());	// Write the Scan	Header Data	
			Size = txtScanHea	der.Text.TrimEnd().Length;	
Size = txtApplica ProjData Appond()	ationDatay.Text.TrimEnd().Length;		if (Size > 0) Proj	Data Append(tytScapHeader Text Tri	mEnd()).
if(Size > 0) Pro-	iData.Append(txtApplicationData9.Tex	t.TrimEnd()):	11(512e > 0) 110)	Data.Append(txtStanneader.Text.TTT	
	)		Size = txtEncoded	<pre>IData.Text.TrimEnd().Length;</pre>	
// Write the App	lication Data 10		ProjData.Append(S	Size.ToString() + "\n");	
Size = lblApplica	ationMarker10.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(txtEncodedData.Text.Tr	imEnd());
if (Size > 0) Pro-	<pre>Size.ToString() + "\n"); iData_Appond(lblApplicationMarkor10 '</pre>	Toxt TrimEnd()).	Sizo - tytOrigina	Hoador Toxt TrimEnd() Longth.	
11(5120 > 0) 110	Jbaca.Append(ibiAppiicacionMarkerio.	iexc.iiiimina()),	ProiData Append(S	Size.ToString() + "\n"):	
Size = txtApplica	ationData10.Text.TrimEnd().Length;		if(Size > 0) Proj	Data.Append(txtOriginalHeader.Text	.TrimEnd());
ProjData.Append(S	Size.ToString() + "\n");				
if(Size > 0) Pro	jData.Append(txtApplicationData10.Te:	<pre>xt.TrimEnd());</pre>	Size = txtOrigina	lEncodedData.Text.TrimEnd().Length	;
1.1			ProjData.Append(S	<pre>Size.ToString() + "\n");</pre>	Tout TwimEnd()).
// // Misc Tab Data			11(312e > 0) PIOJ	Data.Append(txtoriginalEncodedData	. iext. if india());
//			//		
			// Write the data	to a file	
// Write the Rest	tart Marker Data		//		
Size = lblRestart	tMarker.Text.TrimEnd().Length;		sr = new StreamWr	<pre>iter(txtProjectPath.Text.Trim(), factors)</pre>	alse);
if (Size > 0) Pro-	<pre>Size.iOString() + "\n"); iData Append(lblRestartMarker Toyt T</pre>	rimEnd()).	sr.write(ProjData	l);	
II (SIZE > 0) FIO	Joaca, Appena (IDINESCAL CHAINEL, 16XC, 1.	() / ,	sr = null:		
Size = txtRestart	t.Text.TrimEnd().Length;		}		
ProjData.Append(S	Size.ToString() + "\n");		catch(Exception EX)		
		I			

May 02, 04 2:03	frmMain.cs	Page 73/186 Ma	ay 02, 04 2:03	frmMain.cs	Page 74/186
{ ShowWarning( "Warning an ex	cention occured.\n\n" +		/// and 15 and an	error message box, txtError, will	be displayed to
"Exception Erro EX.Message + "\	r:\n" + n\nWas throw by:\n" +		/// /// <param name="Val&lt;/td&gt;&lt;td&gt;ue"/> The Value parameter is an inte	ger value between	
EX.Source + "\n\nNot all sa	ve operations completed.!",		/// 0 and 15 (4-bits /// <returns>Functio</returns>	). n returns a char based upon the he	xadecimal value of
"Save File Exce }	ption");		/// the parameter. private char Convert</td <td>returns&gt; (int Value)</td> <td></td>	returns> (int Value)	
}			{ switch(Value)		
#endregion Common Met	hods		case 0: return ' case 1: return ' case 2: return '	0'; 1'; 2';	
#region Methods to Co	nvert from Binary to ACSII		case 3: return ' case 4: return ' case 5: return '	3'; 4'; 5';	
/// <summary> /// Pre-conditions:</summary>	None.		case 6: return ' case 7: return ' case 8: return '	6'; 7'; 8';	
<pre>/// Post-conditions: /// The LowBits pa /// based upon the /// bit-index of t /// parameter is s /// value of bits</pre>	rameter is set to an ASCII charac value of bits at positions 0 thr he OneByte parameter passed in. et to an ASCII character of 0 to at positions 4 through 7 of the b	ter between 0 to F, ough 3 of the The HighBits F, based upon the it-index of the	case 9: return ' case 10: return case 11: return case 12: return case 13: return case 14: return	9'; 'a'; 'b'; 'c'; 'd';	
<pre>/// OneByte parame /// Description: /// The purpose of /// convert an 8-b</pre>	ter passed in. this method is to allow the call it binary value to two ASCII char	er to easily	case 15: return default: { ShowWarning(	'É';	
<pre>/// the hexadecima /// functionality, /// integer values /// should call th /// returns a char /// hexadecimal re /// returned in th</pre>	<pre>1 value of these 8-bits. To perf this method should split the One , each with 4 bits in them. Then e Convert() method that takes an for each of these two 4-bit valu presentation of each. Then, each e two reference parameters.</pre>	by the parameter into by the parameter into this function integer and es to get the char should be	"Function \" "character!! return 'X'; } }	char Convert(int);\" encountered a \nThis is a SERIOUS error! Please	n unrecognized " + inform developer.");
/// <param name="OneB&lt;br&gt;/// between 0 and 255&lt;br&gt;/// byte.&lt;/param&gt;&lt;/td&gt;&lt;td&gt;yte"/> The OneByte parameter is an (8-bits), representing the value	integer value of one	<pre>/// <summary> /// Pre-conditions: /// Post-conditions:</summary></pre>	None.		
<pre>/// <param lang"<="" name="High /// char where the ch /// of the OneByte pa /// char more " pre=""/></pre>	Bits">The HighBits parameter is a ar value resulting from the 4 mos rameter can be stored.	reference to a t significant bits	/// All of the da /// parameter is /// controls for	ta for the JPEG image based upon t loaded into all of the appropriate the user to view.	he FilePath interface TextBox
/// vhere the char va /// the OneByte param private void SetCharV ref char	lue resulting from the 4 least si eter can be stored. alues(int OneByte, ref char HighB LowBits)	its,	/// The purpose o /// JPEG image in /// within the Ma /// file in binar	f this method is to load the binar to the all of the appropriate Text nipulator interface. This functio y mode and reads all the data from	y file data for a Box data fields n opens the JPEG it. Every byte
<pre>High = OneByte % 25 Low = High % 16; High = High &gt;&gt; 4;</pre>	6; // Get 8 bits // Get the bottom 4 bits // Keep the top 4 bits		<pre>/// read from the /// and is stored /// load all of t /// interface, th</pre>	in the OriginalDataStream data me he data in the OriginalDataStream e LoadInterfaceData() method is ca	mber. Then, to string in to the .lled. Lastly,
HighBits = Convert( LowBits = (Convert( }	High); Low));		<pre>/// this method s /// function exec /// ShowWarning() /// user and the /// error informa</pre>	hould do some error checking to ma utes properly. If an error is enc method should be called to displa txtError TextBox control should be tion.	ke sure this ountered, then the y the error to the updated with this
<pre>/// <summary> /// Pre-conditions: /// Post-conditions: /// Post-conditions:</summary></pre>	None.		<pre>///  /// <param name="Fil /// path to a JPEG i private word LoadDia&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;ePath"/>The FilePath parameter is t mage.</pre>	he file name and	
/// A character ba /// parameter pass /// Description:	ed in should be returned.	me intedet	<pre>private void LoadPic {   try</pre>	curevata(String FliePath)	
<pre>/// The purpose of /// 4-bit value of /// its hexadecima /// M-^QXM-^R if t</pre>	this function allows the caller the parameter to an ASCII charac l value. This function will retu he value of the parameter is not	to convert the ter representing rn the character between the value of 0	{ char Top1 = 'X'; char Bottom1 = '	x';	

May 02, 04 2:03	frmMain.cs	Page 75/186	May 02, 04 2:03	frmMain.cs	Page 76/186
// Open the Origi if(OriginalFile ! OriginalFile = Fi	nal File to Setup Data = null) OriginalFile.Close(); le.OpenRead(FilePath);		<pre>/// to check for /// of the file /// interface. /// method shoul</pre>	errors are possible errors in the str and to make sure no exceptions occur v If an error is encountered, then the 3 d be called to display the error to th	ucture or format when loading the ShowWarning() he user and the
<pre>// Set start valu OriginalDataStrea Value = 0; FiloSize = 0;</pre>	mes m.Length = 0;		<pre>/// txtError Tex /// information. ///  /// (param pamor"Hop</pre>	tBox control should be updated with th	his error
<pre>// Read out the f while(Value != -1</pre>	ile )		/// for a JPEG imag /// hexadecimal val /// file.	e converted to ASCII characters repres ue of each byte found in the original	senting the JPEG
<pre>{     Value = Origina     if(Value == -1)     FileSize++;</pre>	<pre>lFile.ReadByte(); break;</pre>		<pre>private void LoadIn {     char Top1 = 'X';     char Bottom1 = 'X</pre>	<pre>terfaceData(ref StringBuilder HexChars ';</pre>	;)
SetCharValues(V OriginalDataStr OriginalDataStr	<pre>falue, ref Top1, ref Bottom1); eam.Append(Top1.ToString()); eam.Append(Bottom1.ToString());</pre>		<pre>bool Read = true;</pre>		
}	when complete		<pre>int FileLeng = He: int Count = 0; int Temp:</pre>	xChars.Length;	
OriginalFile.Clos	<pre>which complete e();</pre>		Loading = new frm	Load();	
<pre>// Process the fi txtFileSize.Text LoadInterfaceData</pre>	<pre>le string and load windows forms = FileSize + " bytes"; (ref OriginalDataStream);</pre>	with data	ClearData(); EncodedData.Lengt	h = 0;	
}			Loading.StartLoad	<pre>ing(0, FileLeng, 2);</pre>	
<pre>catch(Exception ex) {     if(ShowWarning(</pre>			<pre>wnile(Count &lt; File {     Top1 = HexChars</pre>	Countl:	
"This program h ex.ToString() + "Unhandled Exce	as encountered an UNHANDLED Excep • "\n\nDo you want to close this p ption Occurred!!"	tion!!\n\n" + program?",	Count++; Bottom1 = HexCh. Count++; FileOrder Engue	ars[Count];	
{ menuExit.Perfor	mClick();		FileOrder.Enque	ue(Bottom1);	
}			<pre>// Update the l Loading.UpdateA this.Update();</pre>	oading form ndIncrement();	
}			if(Top1 == 'f' - {	&& Bottoml == 'f')	
/// <summary> /// Pre-conditions:</summary>	None.		<pre>// Read in the Top1 = HexChas Count++</pre>	e next byte to check file marker rs[Count];	
/// All of the cha /// broken apart a /// the Manipulato	racter data contained in the HexC nd stored in the appropriate Text r.	hars parameter is Box data fields in	Bottom1 = Hex Count++; FileOrder.Eng	Chars[Count];	
/// Description: /// The purpose of	this method is to take an string	of ASCII	FileOrder.Eng	ueue(Bottom1);	
/// its various fr /// corresponding	ames and then input all of this d TextBox data field in the interfa	ata to its .ce. As such, this	// Update the	loading form and check for the Cancel	l button
/// function is on /// performs many /// through the da	e of the largest functions in the tasks during its execution. This ta in the HexChars parameter pass	Manipulator and method should read red in. Every time	if(!Loading.U { if(ShowWarn	pdateAndIncrement())	
/// a file marker /// Queue data mem	is found, it should be enqueued i ber. Then, the data found behind	nto the FileOrder	"You have "are you	choosen to cancel this load operation SURE you want to stop, " +	1, " +
/// marker should /// control in the /// account for ev	be loaded into its corresponding e interface of the Manipulator. S very possible marker found within	ince we have to the JPEG standard,	"ALL IOAd "Are you "Cancel L	ed data will be LOSI!\n\n " + sure you want to cancel?", oading?"))	
/// this function /// statements to	should be implemented with a numb satisfy all possibilities. Also,	er of switch as this function	{ ClearData	();	
/// encounters the /// appropriate fi /// should be set.	le structure data members of the Lastly, this method should do l	JPEG Manipulator ots of error	break; } }		

Page 78/186

```
frmMain.cs
                                  frmMain.cs
May 02, 04 2:03
                                                                   Page 77/186
                                                                                     May 02, 04 2:03
         switch(Top1)
         { // JPEG FILE MARKERS, Pg 106 in "JPEG" by: Pennebaker & Mitchell
                                                                                                } // End of: case '0';
           case '0': // Marker ff0X
                                                                                                case '1': goto case 'b';
                                                                                                case '2': goto case 'b';
             switch(Bottom1)
                                                                                                case '3': goto case 'b';
                                                                                                case '4': goto case 'b';
               case '0': // Marker ff00 - Marker Not Defined
                                                                                                case '5': goto case 'b';
                                                                                                case '6': goto case 'b';
                                                                                                case '7': goto case 'b';
                 txtError.Text +=
                                                                                                case '8': goto case 'b';
                   "\nError: Marker NOT defined " +
                   "\n\t-- Marker ff00 was found at byte index: " +
                                                                                                case '9': goto case 'b';
                   ((int)(Count - 4)).ToString();
                                                                                                case 'a': goto case 'b';
                                                                                                case 'b':
                 txtError.Update();
                                                                                                { // Marker ff10 to ffbf - Reserved
                break;
                                                                                                  txtError.Text +=
               case '1': // Marker ff01
                                                                                                     "\nPossible Error: Reserved Marker Found!! " +
                                                                                                    "\n\t-- Marker ff" + Top1.ToString() + Bottom1.ToString()+
                 txtError.Text +=
                                                                                                    " was found at byte index: " +
                   "\nPossible Error: Marker found Temporary use for " +
                                                                                                     ((int)(Count - 4)).ToString();
                   "Arthmetic Encoding " +
                                                                                                  txtError.Update();
                   "\n\t-- Marker ff01 was found at byte index: " +
                                                                                                  break;
                   ((int)(Count - 4)).ToString();
                 txtError.Update();
                                                                                                case 'c': // marker ffcX - huffman tables
                 break;
                                                                                                  switch(Bottom1)
              case '2': goto case 'f';
               case '3': goto case 'f';
                                                                                                     // Start of: Nondifferential Huffman-Coding Frames
              case '4': goto case 'f';
                                                                                                    case '0': // marker ffc0 - Baseline DCT
              case '5': goto case 'f';
              case '6': goto case 'f';
                                                                                                      string info;
              case '7': goto case 'f';
                                                                                                      Read = false;
              case '8': goto case 'f';
              case '9': goto case 'f';
                                                                                                      txtStartHuffman.Text = "ffc0";
              case 'a': goto case 'f';
              case 'b': goto case 'f';
                                                                                                      // Read in the Frame Size to set values
              case 'c': goto case 'f';
                                                                                                      Top1 = HexChars[Count];
              case 'd': goto case 'f';
                                                                                                      Count++;
               case 'e': goto case 'f';
                                                                                                      Bottom1 = HexChars[Count]:
              case 'f':
                                                                                                      Count++;
                                                                                                      FrameSize = SetByteValue(Top1, Bottom1);
                 // Marker ff02 to ff0f - Reserved
                                                                                                      FrameSize = FrameSize << 8;</pre>
                 txtError.Text +=
                                                                                                                             // to get the rest of the counter
                   "\nPossible Error: Reserved Marker Found!! " +
                                                                                                      Top1 = HexChars[Count];
                   "\n\t-- Marker ff0" + Bottom1.ToString()+
                                                                                                      Count++;
                   " was found at byte index: " +
                                                                                                      Bottom1 = HexChars[Count];
                   ((int)(Count - 4)).ToString();
                                                                                                      Count++;
                 txtError.Update();
                                                                                                      FrameSize += SetByteValue(Top1, Bottom1);
                break;
                                                                                                      // Load the size on the interface
                                                                                                      txtStartHuffmanSize.Text = FrameSize.ToString();
               1
                                                                                                      // Update the loading form
              default:
                                                                                                      Loading.LoadProgressValue += 2;
                                                                                                      Loading.UpdateAndIncrement();
                 txtError.Text +=
                                                                                                      this.Update();
                   "\nError: Invalid File Marker Read!! " +
                   "\n\t-- Marker ff0" + Bottom1.ToString()+
                                                                                                      // Get Precision - 1 byte
                   " was found at byte index: " +
                                                                                                      txtPrecision.Text = HexChars[Count].ToString();
                   ((int)(Count - 4)).ToString();
                                                                                                      Count++;
                 txtError.Update();
                                                                                                      txtPrecision.Text += HexChars[Count].ToString();
                                                                                                      Count++;
                 break;
                                                                                                      // Update the loading form
                                                                                                      Loading.UpdateAndIncrement();
             } // End of: switch(Bottom1)
                                                                                                      this.Update();
             break;
                                                                                                      // Get the number of lines - 2 bytes
```

May 02, 04 2:03	frmMain.cs	Page 79/186 Ma	y 02, 04 2:03	frmMain.cs	Page 80/186
May 02, 04 2:03	<pre>frmMain.cs txtNumberHuffmanLines.Text = HexChars[Cour Count++; txtNumberHuffmanLines.Text += HexChars[Cour Count++; txtNumberHuffmanLines.Text += HexChars[Cour Count++; txtNumberHuffmanLines.Text += HexChars[Cour Count++; // Update the loading form Loading.LoadProgressValue += 2; Loading.UpdateAndIncrement(); this.Update(); // Get the number of samples per line - 2 txtNumberHuffmanSamples.Text = HexChars[Cour Count++; txtNumberHuffmanSamples.Text += HexChars[Count++; txtNumberImageComponents.Text += HexChars[Count++; frameSize =</pre>	Page 79/186Matl].ToString();Int].ToString();Int].ToString();Int].ToString();bytesInt].ToString();count].ToString();Int].ToString(); <tr< td=""><td>y 02, 04 2:03 Cound info txtC( txtCd } break; case '1' { if(lbl1 else i: else i:</td><td><pre>frmMain.cs t++; += Topl.ToString() + Bottoml.ToString(); omponents.Text += info; omponents.Text += "\n";  : // marker ffcl - Extended Sequential DCT HuffmanMarker1.Text == "")</pre></td><td><pre>Page 80/186 "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc2"; "ffc2"; "ffc2"; "ffc2"; "ffc2"; "ffc2";</pre></td></tr<>	y 02, 04 2:03 Cound info txtC( txtCd } break; case '1' { if(lbl1 else i: else i:	<pre>frmMain.cs t++; += Topl.ToString() + Bottoml.ToString(); omponents.Text += info; omponents.Text += "\n";  : // marker ffcl - Extended Sequential DCT HuffmanMarker1.Text == "")</pre>	<pre>Page 80/186 "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc1"; "ffc2"; "ffc2"; "ffc2"; "ffc2"; "ffc2"; "ffc2";</pre>
rs[Count-1]);	<pre>Count++; FrameSize = SetByteValue(HexCh // Update the loading form Loading.UpdateAndIncrement(); this Update();</pre>	ars[Count-2], HexCha	else i: else i: else i:	lblHuffmanMarker3.Text = f(lblHuffmanMarker4.Text == "") lblHuffmanMarker4.Text = f(lblHuffmanMarker5.Text == "") lblHuffmanMarker5.Text = f(lblHuffmanMarker6.Text == "") lblHuffmanMarker6.Text =	"ffc2"; "ffc2"; "ffc2"; "ffc2";
	<pre>info = "Identifier, Horizontal, Vertical, txtComponents.Text = info; for(int a = FrameSize; a &gt; 0; a)</pre>	Q-Table: \n";	else i: break; }	<pre>lbHuffmanMarker7.Text = f(lbHuffmanMarker7.Text = f(lbHuffmanMarker8.Text =</pre>	"ffc2"; "ffc2";
	<pre>{     // Component identifier     Top1 = HexChars[Count];     Count++;     Bottom1 = HexChars[Count];     Count++;     info = Top1.ToString() + Bottom1.ToStrin     // Horizontal and Vertical Sampling fact     Top1 = HexChars[Count];     Count++;     Bottom1 = HexChars[Count];     Count++;     info += Top1.ToString() + ", " +     Bott     Bott</pre>	or on1.ToString() + ",	case '3' { if(lbll else i: else i: else i: else i: else i: else i:	<pre>: // marker ffC3 - Lossless (Sequential) HuffmanMarker1.Text == "")</pre>	"ffc3"; "ffc3"; "ffc3"; "ffc3"; "ffc3"; "ffc3";
";	<pre>// Quantization Table Selector Top1 = HexChars[Count]; Count++; Bottom1 = HexChars[Count];</pre>		else i: break; } // End	lblHuffmanMarker7.Text = f(lblHuffmanMarker8.Text == "") lblHuffmanMarker8.Text = of: Nondifferential Huffman-Coding Frames	"ffc3"; "ffc3";

May 02, 04 2:03	frmMain.cs	Page 81/186	May 02, 04 2:03	frmMain.cs	Page 82/186
				else if(lblHuffmanMarker2.Text == "")	
	case '4': // marker ffc4 - Define Huffman Marker			else if(lblHuffmanMarker3.Text == "")	= "IIC/";
	<pre>{     if(lb]HuffmanMarker1.Text == "")</pre>			<pre>lblHuffmanMarker3.Text else if(lblHuffmanMarker4.Text == "")</pre>	. = "ffc7";
	lblHuffmanMarker1.Text =	= "ffc4";		lblHuffmanMarker4.Text	= "ffc7";
	else if(lblHuffmanMarker2.Text == "") lblHuffmanMarker2.Text =	= "ffc4";		else if(lblHuffmanMarker5.Text == "") lblHuffmanMarker5.Text	= "ffc7";
	else if(lblHuffmanMarker3.Text == "")	= "ffc4"·		<pre>else if(lblHuffmanMarker6.Text == "")</pre>	= "ffc7".
	else if(lblHuffmanMarker4.Text == "")			else if(lblHuffmanMarker7.Text == "")	
	= IDIHUIIMANMARKEr4.1ext else if(lblHuffmanMarker5.Text == "")	= "IIC4";		<pre>LDIHUIIManMarker/.lext else if(lblHuffmanMarker8.Text == "")</pre>	= "IIC/";
	<pre>lblHuffmanMarker5.Text = else if(lblHuffmanMarker6.Text == "")</pre>	= "ffc4";		<pre>lblHuffmanMarker8.Text break:</pre>	. = "ffc7";
	lblHuffmanMarker6.Text =	= "ffc4";		} // End of. Differential Uniffman Coding Energy	
	lblHuffmanMarker7.Text = )	= "ffc4";		// End of: Differencial nuriman-couling frames	
	<pre>else if(lblHuffmanMarker8.Text == "")</pre>	= "ffc4";		case '8': // marker ffc8 - Reserved for JPEG Ext	ensions
	break;			{	
	ſ			"\nPossible Error: Reserved For JPEG Extensi	ons Marker"+
	// Start of: Differential Huffman-Coding Frames		byte index: " +	"Found!!\n\t Marker FF	CD was found at
	<pre>case '5': // marker ffc5 - Differential Sequential </pre>	L DCT	-	<pre>((int)(Count - 4)).ToString(); tytError_Update();</pre>	
	<pre>if(lblHuffmanMarker1.Text == "")</pre>			Read = false; // Skip reading values for this	marker
	else if(lblHuffmanMarker2.Text == "")	= "1105";		<pre>preak; }</pre>	
	<pre>lblHuffmanMarker2.Text = else if(lblHuffmanMarker3.Text == "")</pre>	= "ffc5";			
	lblHuffmanMarker3.Text =	= "ffc5";		<pre>// Start of: Nondifferential Arithmetic-Coding case '9': // marker ffc9 - Extended Sequential</pre>	/ Frames
	lblHuffmanMarker4.Text =	= "ffc5";		{	501
	else if(lblHuffmanMarker5.Text == "") lblHuffmanMarker5.Text =	= "ffc5";		<pre>if(lbiHuffmanMarker1.lext == "")</pre>	= "ffc9";
	<pre>else if(lblHuffmanMarker6.Text == "")</pre>	= "ffc5";		<pre>else if(lblHuffmanMarker2.Text == "")</pre>	: = "ffc9";
	else if(lblHuffmanMarker7.Text == "")	- "ffc5".		else if(lblHuffmanMarker3.Text == "")	- "ffc9".
	else if(lblHuffmanMarker8.Text == "")	"		else if(lblHuffmanMarker4.Text == "")	- 1109 ,
	<pre></pre>	= "IIC5";		<pre>LDIHUIIManMarker4.lext else if(lblHuffmanMarker5.Text == "")</pre>	= "IIC9";
	<pre>} case '6': // marker ffc6 - Differential Progressiv</pre>	ve DCT		<pre>lblHuffmanMarker5.Text else if(lblHuffmanMarker6.Text == "")</pre>	. = "ffc9";
	{     if(lb]HuffmanMarkor1 Toyt "")			lblHuffmanMarker6.Text	. = "ffc9";
	lblHuffmanMarker1.Text =	= "ffc6";		lblHuffmanMarker7.Text	= "ffc9";
	else if(lblHuffmanMarker2.Text == "") lblHuffmanMarker2.Text =	= "ffc6";		else if(lblHuffmanMarker8.Text == "") lblHuffmanMarker8.Text	= "ffc9";
	<pre>else if(lblHuffmanMarker3.Text == "")</pre>	= "ffc6":		break;	
	else if(lblHuffmanMarker4.Text == "")	, "ffo6",		case 'a': // marker ffca - Progressive DCT	
	else if(lblHuffmanMarker5.Text == "")	- 1100 ;		<pre>if(lblHuffmanMarker1.Text == "")</pre>	
	<pre> lblHuffmanMarker5.Text =   else if(lblHuffmanMarker6.Text == "")</pre>	= "ffc6";		lblHuffmanMarkerl.Text else if(lblHuffmanMarker2.Text == "")	= "ffca";
	lblHuffmanMarker6.Text =	= "ffc6";		lblHuffmanMarker2.Text else if(lblHuffmanMarker3.Text == "")	. = "ffca";
	lblHuffmanMarker7.Text =	= "ffc6";		lblHuffmanMarker3.Text	. = "ffca";
	lblHuffmanMarker8.Text == "")	= "ffc6";		lblHuffmanMarker4.Text	= "ffca";
	break; }			<pre>else if(lblHuffmanMarker5.Text == "")</pre>	: = "ffca";
	case '7': // marker ffc7 - Differential Lossless			else if(lblHuffmanMarker6.Text == "")	- "ffca".
	<pre>if(lblHuffmanMarker1.Text == "")</pre>			else if(lblHuffmanMarker7.Text == "")	- IICa";
	lblHuffmanMarker1.Text =	= "ffc7";	1	lblHuffmanMarker7.Text	. = "ffca";

May 02, 04 2:03	frmMain.cs	Page 83/186	May 02, 04
	else if(lblHuffmanMarker8.Text == "")	"ffoo".	
	break;	IICa ;	
	} case 'b': // marker ffcb - Lossless (Sequential) {		
	<pre>if(lblHuffmanMarker1.Text == "") </pre>	"	
	<pre>iDiHuIImanMarker1.lext = else if(lblHuffmanMarker2.Text == "")</pre>	"IICD";	
	else if(lblHuffmanMarker3.Text == "")	"ffch",	
	else if(lblHuffmanMarker4.Text == ")	IICD ;	
	else if(lblHuffmanMarker5.Text == "")	"IICD";	
	<pre>iblHuffmanMarker5.Text = else if(lblHuffmanMarker6.Text == "")</pre>	"iicb";	
	lblHuffmanMarker6.Text = else if(lblHuffmanMarker7.Text == "")	"ffcb";	
	<pre>lblHuffmanMarker7.Text = else if(lblHuffmanMarker8.Text == "")</pre>	"ffcb";	
	<pre>lblHuffmanMarker8.Text = break;</pre>	"ffcb";	
	} // End of: Nondifferential Arithmetic-Coding Fra	mes	
	,,		
	<pre>case 'c': // marker ffcc - Define Arithmetic Condit {</pre>	tioning Tables	
	<pre>if(lblHuffmanMarker1.Text == "")</pre>	"ffcc";	
	<pre>else if(lblHuffmanMarker2.Text == "")</pre>	"ffcc";	
	<pre>else if(lblHuffmanMarker3.Text == "")</pre>	"ffcc":	
	else if(lblHuffmanMarker4.Text == "") lblHuffmanMarker4 Text =	"ffcc"·	
	else if(lblHuffmanMarker5.Text == "")	"ffcc",	
	else if(lblHuffmanMarker6.Text == "")	lice;	
	else if(lblHuffmanMarker7.Text == "")	"1100";	
	<pre>iblHuffmanMarker/.Text = else if(lblHuffmanMarker8.Text == "")</pre>	"fice";	
	<pre>lblHuffmanMarker8.Text = break;</pre>	"ffcc";	
	}		
	// Start of: Differential Arithmetic-Coding Frame case 'd': // marker ffcd - Differential Sequential	es DCT	
	{     if(lblHuffmanMarker1.Text == "")		
	<pre>lblHuffmanMarker1.Text = else if(lblHuffmanMarker2.Text == "")</pre>	"ffcd";	
	<pre>lblHuffmanMarker2.Text = else if(lblHuffmanMarker3 Text == "")</pre>	"ffcd";	
	lbHuffmanMarker3.Text =	"ffcd";	
	lbHuffmanMarker4.Text =	"ffcd";	
	lblHuffmanMarker5.Text =	"ffcd";	
	else ll(lDlHuffmanMarker6.Text == "") lblHuffmanMarker6.Text =	"ffcd";	
	<pre>else if(lblHuffmanMarker7.Text == "")</pre>	"ffcd";	
	<pre>else if(lblHuffmanMarker8.Text == "")</pre>	"ffcd";	

2:03	frmMain.cs	Page 84/186
} C	ase 'e': // marker ffce - Differential Progressive	DCT
1	if(lblHuffmanMarker1.Text == "") lblHuffmanMarker1.Text =	"ffce";
	<pre>else if(lblHuffmanMarker2.Text == "")</pre>	"ffce";
	<pre>else if(lblHuffmanMarker3.Text == "")</pre>	"ffce";
	<pre>else if(lblHuffmanMarker4.Text == "")</pre>	"ffce":
	<pre>else if(lblHuffmanMarker5.Text == "")</pre>	"ffce":
	else if(lblHuffmanMarker6.Text == "")	"ffce",
	else if(lblHuffmanMarker7.Text == "")	"ffco",
	else if(lblHuffmanMarker8.Text == "")	"IICe";
	<pre>ibiHuiimanMarker8.lext = break;</pre>	"IICe";
} C	ase 'f': // marker ffcf - Differential Lossless	
{	if(lblHuffmanMarker1.Text == "")	
	<pre>lblHuffmanMarker1.Text = else if(lblHuffmanMarker2.Text == "")</pre>	"ffcf";
	<pre>lblHuffmanMarker2.Text = else if(lblHuffmanMarker3.Text == "")</pre>	"ffcf";
	<pre>lblHuffmanMarker3.Text = else if(lblHuffmanMarker4.Text == "")</pre>	"ffcf";
	lblHuffmanMarker4.Text =	"ffcf";
	<pre>blHuffmanMarker5.Text = else if(lblHuffmanMarker6 Text "")</pre>	"ffcf";
	blHuffmanMarker6.Text =	"ffcf";
	lblHuffmanMarker7.Text =	"ffcf";
	else if(lbHuffmanMarker8.lext == "") lbHuffmanMarker8.Text =	"ffcf";
}	break;	
	// End of: Differential Arithmetic-Coding Frames	
d {	efault:	
	<pre>txtError.Text +=   "\nError: Invalid File Marker Read!! " +   "\n\t Marker ffc" + Bottoml.ToString()+   " was found at byte index: " +   ((int)(Count - 4)).ToString(); txtError.Update();</pre>	
}	break;	
}	// End of: switch(Bottom1)	
if(	Read)	
{ 7 C	<pre>/ Read in the Frame Size to set values op1 = HexChars[Count]; ount++;</pre>	
B C F F	ottomi = HexChars[Count]; ount++; rameSize = SetByteValue(Top1, Bottom1); rameSize = FrameSize << 8;	
T	<pre>// to get the rest of the counter op1 = HexChars[Count]; ount++:</pre>	

break;

Page 86/186

May 02, 04 2:03	frmMain.cs	Page 85/186 May	02, 04 2:03	frmMain.cs	Page 86/18
Bottom Count+	1 = HexChars[Count]; +;		} else if(t>	tHuffman4.Text == "")	
FrameS	ize -= 2; // For the 2 bytes that h	hold the frame size	{ SizeOfHu while(Fr	ffman[3] = FrameSize + 2; rameSize > 0)	
// Upd Loadin Loadin	ate the loading form g.LoadProgressValue += 2; g.UpdateAndIncrement();		{	are counting down the FrameSize t HexChars[Count];	o start the stream.
this.U	<pre>pdate();</pre>		Count+ Botton	+; 1 = HexChars[Count];	
if(txt {	Huffman1.Text == "")		Count+ FrameS	ize;	
size whil	e(FrameSize > 0)		txtHui	Bottom1.ToS	String() + " ";
۱ // То	We are counting down the FrameSize	to start the stream.	// Updat	e the loading form	
Co Bo Co	<pre>unt+; ttom1 = HexChars[Count]; unt++;</pre>		Loading. Loading. this.Upc	LoadProgressValue += SizeOfHuffma UpdateAndIncrement(); late();	an[3] - 2;
Fr	ameSize; tHuffman1.Text += Top1.ToString() +		} else if(t>	tHuffman5.Text == "")	
}	Bottomi.ic	Sstring() + " ";	{ SizeOfHu while(Fr	affman[4] = FrameSize + 2; ameSize > 0)	
// U Load Load this	<pre>pdate the loading form ing.LoadProgressValue += SizeOfHuffm ing.UpdateAndIncrement(); .Update():</pre>	nan[0] - 2;	{ // We Top1 = Count+	<pre>are counting down the FrameSize t HexChars[Count]; +:</pre>	o start the stream.
} else i	f(txtHuffman2.Text == "")		Botton Count+ FrameS	<pre>il = HexChars[Count]; +; size:</pre>	
Size whil	OfHuffman[1] = FrameSize + 2; e(FrameSize > 0)		t×tHuf	fman5.Text += Top1.ToString() + Bottom1.ToS	String() + " ";
// To	<pre>We are counting down the FrameSize p1 = HexChars[Count]; unt++:</pre>	to start the stream.	// Updat	e the loading form LoadProgressValue += SizeOfHuffma	an[4] - 2:
Bo Co Fr	<pre>ttom1 = HexChars[Count]; unt++; ameSize:</pre>		Loading. this.Upc	<pre>UpdateAndIncrement(); late();</pre>	
tx	tHuffman2.Text += Top1.ToString() + Bottom1.To	<pre>&gt;String() + " ";</pre>	else if(tx {	<pre>tHuffman6.Text == "")</pre>	
}	ndata tha laading faum		SizeOfHu while(Fr	affman[5] = FrameSize + 2; cameSize > 0)	
// U Load Load	<pre>ing.LoadProgressValue += SizeOfHuffm ing.UpdateAndIncrement(); Update():</pre>	nan[1] - 2;	// We Topl =	are counting down the FrameSize t HexChars[Count];	o start the stream.
} else i	f(txtHuffman3.Text == "")		Botton Count+ FrameS	<pre>il = HexChars[Count]; '; '; ';</pre>	
Size	OfHuffman[2] = FrameSize + 2; e(FrameSize > 0)		txtHuf	fman6.Text += Top1.ToString() + Bottom1.ToS	String() + " ";
// To	We are counting down the FrameSize p1 = HexChars[Count];	to start the stream.	// Updat	e the loading form	
Bo	<pre>unt++; ttom1 = HexChars[Count]; unt++; amoSizo;</pre>		Loading. Loading. this.Upc	<pre>UpdateAndIncrement(); late();</pre>	m[J] - Z;
tx	tHuffman3.Text += Topl.ToString() + Bottoml.To	String() + " ";	else if(tx	<pre>tHuffman7.Text == "")</pre>	
}			SizeOfHu while(Fr	uffman[6] = FrameSize + 2; rameSize > 0)	
// U Load Load this	<pre>pdate the loading form ing.LoadProgressValue += SizeOfHuffm ing.UpdateAndIncrement(); .Update();</pre>	nan[2] - 2;	{ // We Top1 = Count+	are counting down the FrameSize t HexChars[Count];	o start the stream

May 02, 04 2:03	frmMain.cs	Page 87/186 N	lay 02, 04 2:03	frmMain.cs	Page 88/186
	<pre>Bottom1 = HexChars[Count];</pre>		Ca	ase '9':	
	Count++; FrameSize;		{	// Marker IId9 : End of image // Covered bv: case ffda	
	<pre>txtHuffman7.Text += Top1.ToString() +</pre>			break;	
	Bottom1.ToString	() + " ";	}		
			Ca	ase 'a':	
	<pre>// Update the loading form Loading LoadProgressValue += SizeOfHuffman[6] -</pre>	- 2:	{	// Marker fida : Start of Scan	
	Loading.UpdateAndIncrement();	-,		int i = 0;	
1	<pre>this.Update();</pre>			Top1 = 'X'	
e	else if(txtHuffman8.Text == "")			Bottom1 = 'X';	
{	SizeOfHuffman[7] = FrameSize + 2:			FrameSize = 0;	
	while (FrameSize > 0)			// Get Scan Header	
	{     // We are counting down the FrameSize to star	rt the stream		Top1 = HexChars[Count];	
	Top1 = HexChars[Count];	e ene seream.		Bottom1 = HexChars[Count];	
	Count++; Bottom1 - HeyChars[Count]:			Count++; FrameSize - SetByteValue(Top1 Bottom1):	
	Count++;			FrameSize = FrameSize << 8;	
	<pre>FrameSize; tytHuffman8 Toyt += Top1 ToString() +</pre>			<pre>Top1 = HexChars[Count]; Count++:</pre>	
	Bottom1.ToString	() + " ";		Bottom1 = HexChars[Count];	
	}			Count++; FrameSizo += SotButoValue(Top1 _ Bottom1);	
	// Update the loading form			SizeOfScanHeader = FrameSize;	
	Loading.LoadProgressValue += SizeOfHuffman[7] -	- 2;		FrameSize -= 2;	
	this.Update();			// Update the loading form	
}				Loading.LoadProgressValue += 2;	
{	126			this.Update();	
	// Show an error.			for(i = 0, i < FramoSizo, i++)	
}				{	
۱ /	(/ End of . if (Pood) .			Top1 = HexChars[Count];	
els	se			Bottom1 = HexChars[Count];	
{ 	ead = true.			Count++; txtScanHeader Text +=	
}				Top1.ToString() + Bottom1.ToString() + " ";	
bre	pak:			} txtScanHeader.Update():	
} //	End of: case 'c': // marker ffcX				
case	'd': // marker ffdX			// Update the loading form Loading.LoadProgressValue +=	
{				((txtScanHeader.Text.Leng	gth * 2)/3) -
SW1 {	tch(Bottom1)	2;		Loading.UpdateAndIncrement();	
c	case '0': goto case '7';			this.Update();	
c	ase '2': goto case '7';			// Get the encoded data stream	
c	case '3': goto case '7';			<pre>temp = HexChars.Length - (Count + 4); EncodedDate Incount (0)</pre>	
c	ase '5': goto case '7';			HexChars.ToString().Subst	tring(Count, t
c	case '6': goto case '7';	emj	p));		
{	// Marker ffd0 to ffd7			// Update the loading form	
	<pre>txtRestartMod8.Text = ((int)(Count - 4)).ToStr: broak.</pre>	ing();		Loading.LoadProgressValue += EncodedData.Length -	- 2;
}	bican,			this.Update();	
~	rase /8/:			OriginalEncodedData = EncodedData ToString().	
{	// Marker ffd8 : Start of Image			originarineoucupaca - incodeupaca.roscring();	
۱	break;			int MaxDisplay = 10240; // 5k in file size	
1				(	

```
frmMain.cs
                                                                                                                         frmMain.cs
 May 02, 04 2:03
                                                                     Page 89/186
                                                                                       May 02, 04 2:03
                                                                                                                                                          Page 90/186
                    txtEncodedData.Text = EncodedData.ToString();
                                                                                                          SizeOfOuantizer[0] = FrameSize + 3;
                                                                                                          // Read out the table data
                  else
                                                                                                          while(FrameSize > 0)
                                          txtEncodedData.Text =
                                                  EncodedData.ToString().Substri
                                                                                                            // We are counting down the FrameSize to start the
ng(0, MaxDisplay);
                                                                                                                                // stream.
                                                                                                            Top1 = HexChars[Count];
                                                                                                            Count++;
                                                                                                            Bottom1 = HexChars[Count];
                  Count += temp;
                                                                                                           Count++;
                  txtEncodedData.Update();
                                                                                                           FrameSize--;
                                                                                                            txtQuantizer1.Text += Top1.ToString() +
                  Top1 = 'f';
                                                                                                                                        Bottom1.ToString() + " ";
                  Bottom1 = 'f';
                                                                                                          // Update the loading form
                                                                                                          Loading.LoadProgressValue += SizeOfQuantizer[0] - 2;
                  break;
                                                                                                          Loading.UpdateAndIncrement();
                                                                                                          this.Update();
                case 'b':
                { // Marker ffdb : Define Quantization Table
                                                                                                        else if(txtQuantizer2.Text == "")
                  if(lblQuantizerMarker1.Text == "")
                                          lblQuantizerMarker1.Text = "ffdb";
                                                                                                          // Read in the table Number - 1 byte
                  else if(lblQuantizerMarker2.Text == "")
                                                                                                         Top1 = HexChars[Count];
                                          lblQuantizerMarker2.Text = "ffdb";
                                                                                                          Count++;
                  else if(lblQuantizerMarker3.Text == "")
                                                                                                          Bottom1 = HexChars[Count];
                                          lblQuantizerMarker3.Text = "ffdb";
                                                                                                          Count++;
                  else if(lblQuantizerMarker4.Text == "")
                                                                                                          FrameSize--;
                                          lblQuantizerMarker4.Text = "ffdb";
                                                                                                          txtQuantizerTableNum2.Text =
                                                                                                           Top1.ToString() + Bottom1.ToString();
                  // Read in the Frame Size to set values
                  Top1 = HexChars[Count];
                                                                                                          // Update the loading form
                                                                                                          Loading.UpdateAndIncrement();
                  Count++;
                  Bottom1 = HexChars[Count];
                                                                                                          this.Update();
                  Count++;
                  FrameSize = SetByteValue(Top1, Bottom1);
                                                                                                          // 2 for framesize field and 1 for table number
                  FrameSize = FrameSize << 8;
                                                                                                          SizeOfQuantizer[1] = FrameSize + 3;
                                        // to get the rest of the counter
                  Top1 = HexChars[Count];
                                                                                                          // Read out the table data
                  Count++;
                                                                                                          while(FrameSize > 0)
                  Bottom1 = HexChars[Count];
                  Count++;
                                                                                                            // We are counting down the FrameSize to start the
                  FrameSize += SetByteValue(Top1, Bottom1);
                                                                                                                            // stream.
                  FrameSize -= 2;
                                                                                                            Top1 = HexChars[Count];
                                        // For the 2 bytes that hold the frame s
                                                                                                            Count++;
ize
                                                                                                           Bottom1 = HexChars[Count];
                                                                                                           Count++;
                  // Update the loading form
                                                                                                           FrameSize--;
                  Loading.LoadProgressValue += 2;
                                                                                                            txtQuantizer2.Text += Top1.ToString() +
                  Loading.UpdateAndIncrement();
                                                                                                                                        Bottom1.ToString() + " ";
                  this.Update();
                                                                                                          // Update the loading form
                  if(txtQuantizer1.Text == "")
                                                                                                          Loading.LoadProgressValue += SizeOfQuantizer[1] - 2;
                                                                                                          Loading.UpdateAndIncrement();
                    // Read in the table Number - 1 byte
                                                                                                          this.Update();
                    Top1 = HexChars[Count];
                    Count++;
                                                                                                        else if(txtQuantizer3.Text == "")
                    Bottom1 = HexChars[Count];
                    Count++;
                                                                                                          // Read in the table Number - 1 byte
                    FrameSize--:
                                                                                                         Top1 = HexChars[Count];
                    txtQuantizerTableNum1.Text =
                                                                                                          Count++;
                      Top1.ToString() + Bottom1.ToString();
                                                                                                         Bottom1 = HexChars[Count];
                                                                                                         Count++;
                    // Update the loading form
                                                                                                         FrameSize--;
                    Loading.UpdateAndIncrement();
                                                                                                          txtQuantizerTableNum3.Text =
                    this.Update();
                                                                                                           Top1.ToString() + Bottom1.ToString();
                    // 2 for framesize field and 1 for table number
                                                                                                          // Update the loading form
```

<pre>Lasting.bdotchdinceremet(); thisdbotchdinceremet(</pre>	May 02, 04 2:03	frmMain.cs	Page 91/186	May 02, 04 2:03	frmMain.cs	Page 92/186
<pre>// 2 tor remaining () = fraction () to take number firstDecomposition () = fraction () // We are conting down tor Promotion to totate the // We are conting down tor Promotion to totate the // We are conting down tor Promotion to totate the // We are conting down tor Promotion to totate the // We are conting down tor Promotion to totate the // We are conting down tor Promotion to totate the // We are conting down tor Promotion tor () // Promotion = Promotion () // Promo</pre>		<pre>Loading.UpdateAndIncrement(); this.Update();</pre>		case 'c': {	er ffdc : Define number of lines,	4 bytes
<pre>// Kead out the scale data while("reading form the FrameWise to start the</pre>		<pre>// 2 for framesize field and 1 for tabl SizeOfQuantizer[2] = FrameSize + 3;</pre>	e number	// Read Top1 = H	out 4 bytes lexChars[Count];	
<pre>// We are counting down the PrameSize to start the // We are counting down the PrameSize to start the // Topi - MaxChars(Count); Count+; count+;</pre>		<pre>// Read out the table data while(FrameSize &gt; 0)</pre>		Count++; Bottom1 Count++;	= HexChars[Count];	
<pre>Topl = fexChars(Count);</pre>		{ // We are counting down the FrameSize // stream.	to start the	FrameSiz FrameSiz	<pre>se = SetByteValue(Topl, Bottoml); se = FrameSize &lt;&lt; 8;</pre>	f the counter
<pre>Count+; FrameSize; FrameSize; fully detained form // underg.logderygreatylum (=&gt; StoofQuantizer[2] = 2; Loading.logderygreatylum (=&gt; StoofQuantizer[2] = 2; Loading.logde</pre>		Top1 = HexChars[Count]; Count++; Bottom1 = HexChars[Count];		Top1 = H Count++; Bottom1	<pre>lexChars[Count]; = HexChars[Count];</pre>	
<pre>Bottoml.roString() + " "; // Dpdate the loading form Loading.LoadProgressWalue += SizeDTQuantizer[2] - 2; Loading.LoadProgressWalue += SizeDTQuantizer[2] - 2; Loading.LoadProgressWalue += SizeDTQuantizer[2] - 2; Loading.Update[1] // Read in the toble Number - 1 byte Topl = HexChars[Count]; Count+; Bottoml = HexChars[Count]; Count+; Dottom: - HexChars[Count]; Count+; Bottoml = HexChars[Count]; Count+; Dottom: - HexChars[Count]; Count+; HexChars[Count]; Count+; Dottom: - HexChars[Count]; Count+; HexChars[Count]; Count+; Dottom: - HexChars[Count]; Count+; He</pre>		Count++; FrameSize; txtQuantizer3.Text += Top1.ToString()	+	Count++; FrameSiz FrameSiz	<pre>se += SetByteValue(Top1, Bottom1); se = FrameSize &lt;&lt; 8;</pre>	
<pre>loading.LoadfrogreesVilue - SizeOfQuantizer[2] - 2; Loading.UpdateAninerment(); this.Update(); class if(rxQuantizer4.Text "") { // Read in the table Number - 1 byte Topi - Machara[Count]; Count+; FrameSize - TraimElize -= StateVilue(Tep1, Bottom1); Count+; Topi - Machara[Count]; Count+; TraimElize-; Topi.ToString() + Bottom1.ToString(); // Update the loading form Loading.LoadFrogreesVilue (Tep1, Bottom1); // Update the loading form Loading.LoadFrogreesVilue (Tep1, Bottom1); // Update the loading form SizeOfQuantizer[3] - FrameSize to start the // Meak out the table data while(FrameSize &gt; 0) { // Meak counting form Loading.LoadFrogreesVilue (Tep1, Bottom1); // Dyta = the States(Count]; Count+; Topi = Mexchars(Count]; Count+; Topi = Mexchars(Count]; Count+; TrameSize + StateVilue(Tep1, Bottom1); TrameSize + StateVilue(Tep1, Bottom1); Count+; TrameSize + StateVilue(Tep1, Bottom1); TrameSize</pre>		Bottom1.T } // Update the loading form	oString() + " ";	Top1 = H Count+;	<pre> // to get the rest of   lexChars[Count];</pre>	f the counter
<pre>intervent() is if (it to intervent()) i</pre>		<pre>Loading.LoadProgressValue += SizeOfQuan Loading.UpdateAndIncrement(); this.Update():</pre>	tizer[2] - 2;	Bottom1 Count+; FrameSiz	<pre>= HexChars[Count]; re += SetByteValue(Top1, Bottom1);</pre>	
<pre>// Read in the table Number - 1 byte Top1 = HexChars[Count]; Count+; Count+; Count+; Count+; FrameSize = SethyteValue(Top1, Bottom1); Count+; FrameSize = SethyteValue(Top1, Bottom1); // Update the loading form Loading.UpdateAndIncrement(); this.Update(); // Update the loading form Loading.UpdateAndIncrement(); this.Update(); // We are counting down the FrameSize to start the // Top1 = HexChars[Count]; Count+; Bottom1 = HexChars</pre>		<pre>} } else if(txtQuantizer4.Text == "") </pre>		FrameSiz	<pre>ice = FrameSize &lt;&lt; 8;</pre>	f the counter
<pre>bottom1_ = haxChars[Count]; Count+; FrameSize += SetByteValue(Topl, Bottom1); // Update the loading form Loading.UpdateAddIncrement(); this.Update(); // Dpdate the loading form Loading.UpdateAddIncrement(); this.Update(); // Z for frameSize field and 1 for table number SizeOfQuantizer[3] = FrameSize + 3; // Nead out the table data while(FrameSize &gt; 0) { // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // Stream. Topl = HaxChars[Count]; Count++; FrameSize=-; tatQuantizer[3] = SizeOfQuantizer[3] = 2; Loading.UpdateAndIncrement(); this.Update(); // Update the loading form Loading.LoadProgressValue += SizeOfQuantizer[3] = 2; Loading.UpdateAndIncrement(); this.Update(); } break; } // Show an error } break; } // Show an error } } break; } // Show an error } } // Show an error } // Show an error // Show an error</pre>		<pre>// Read in the table Number - 1 byte Top1 = HexChars[Count]; CountLine</pre>		Count++; Bottoml	= HexChars[Count];	
<pre>// Solve the humber to files data trops.rest. // Update the loading form Loading.UpdateAndIncrement(); this.Update(); // 2 for framesize field and 1 for table number SizeOfQuartizer[3] = FrameSize + 3; // Read out the table data while(FrameSize &gt; 0) { // We are counting down the FrameSize to start the rop1 = HexChars[Count]; Count++; Bottoml.Fostring() + txtQuartizer4.Text = Top1.ToString() + txtQuartizer4.Text = Top1.ToString() + txtQuartizer4.Text = SizeOfQuartizer[3] - 2; Loading.UpdateAndIncrement(); this.Update(); } // Read out the table data while(FrameSize &gt; 0) { // We are counting down the FrameSize to start the rop1 = HexChars[Count]; Count++; Bottoml.ToString() + txtQuartizer4.Text = Top1.ToString() + txtQuartizer4.Text = SizeOfQuartizer[3] - 2; Loading.UpdateAndIncrement(); this.Update(); } // Read out the table data while(frameSize = Count); Count++; Bottoml.ToString() + txtQuartizer4.Text = Top1.ToString() + this.Update(); } // Show an error } // Show an error // Show a</pre>		Bottom1 = HexChars[Count]; Count++;		FrameSiz	<pre>:e += SetByteValue(Top1, Bottom1);</pre>	
<pre>// Update the loading form Loading.UpdateAndIncrement(); this.Update(); // 2 for framesize field and 1 for table number SizeOfQuantizer[3] = FrameSize + 3; // Read out the table data while(FrameSize &gt; 0) { // We are counting down the FrameSize to start the // Stream. Top1 = HexChars[Count]; Count+; Ending.LoadProgressValue += 2; Loading.Update.The FrameSize + 3; // Read out the table data while(FrameSize &gt; 0) { // We are counting down the FrameSize to start the // Stream. Top1 = HexChars[Count]; Count+; FrameSize; txtQuantizer4.Text += Top1.ToString() + txtQuantizer4.Text += Top1.ToString() + Loading.UpdateAndIncrement(); this.Update(); } // Update the loading form Loading.UpdateAndIncrement(); this.Update(); } // Show an error ; // Show an error ; // Show an error ; // Show an error ; break; } break; } </pre>		<pre>rfameSize-; txtQuantizerTableNum4.Text = Top1.ToString() + Bottom1.ToString();</pre>		// Store NumberOf	Lines = FrameSize;	
<pre>// 2 for framesize field and 1 for table number SizeOfQuantizer[3] = FrameSize + 3; // Read out the table data while(FrameSize &gt; 0) { // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // Read out 4 bytes Top1 = HexChars[Count]; Count++; Bottoml = HexChars[Count]; Count++; FrameSize=-; txtQuantizer4.Text += Top1.ToString() + ""; } // Update the loading form Loading.LoadProgressValue += SizeOfQuantizer[3] - 2; Loading.UpdateAndIncmement(); this.Update(); } break; } // Show an error } } break; } // Show an error } } break; } // Update the loading form Loading.LoadProgressValue += SizeOfQuantizer[3] - 2; Loading.UpdateAndIncmement(); this.Update(); } break; } // Show an error } } break; } // Show an error } } break; } } // Show an error } } break; } // Show an error } } break; } } // Show an error } } break; } } break; } } break; } break; } break; } break; } Show an error } break; break; break; break; } break; break; break; break; break; break; break; break; break; break; break; break; break; break; break; break break; break; break; b</pre>		<pre>// Update the loading form Loading.UpdateAndIncrement(); this.Update();</pre>		// Updat Loading. Loading. this.Upd	<pre>le the loading form LoadProgressValue += 2; UpdateAndIncrement(); late();</pre>	
<pre>// Read out the table data while (FrameSize &gt; 0) {     // Kead out the table data while (FrameSize &gt; 0)     // We are counting down the FrameSize to start the     // Stream.     Top1 = HexChars[Count];     Count+;     FrameSize-;     txtQuantizer4.Text += Top1.ToString() + " ";     // Loading.toadProgressValue += SizeOfQuantizer[3] - 2;     Loading.UpdateAndIncrement();     this.Update();     else     {         // Show an error         /</pre>		<pre>// 2 for framesize field and 1 for tabl SizeOfQuantizer[3] = FrameSize + 3;</pre>	e number	lblNumbe txtNumbe FrameSiz	<pre>erLinesMarker.Text = "ffdc"; erLines.Text = FrameSize.ToString() e = 0:</pre>	;
<pre>// We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // We are counting down the FrameSize to start the // Warker ffdd : Define restart interval, 4 bytes // Marker ffdd : Define restart interval, 4 bytes // Read out 4 bytes Top1 = HexChars[Count]; Count++; FrameSize; txtQuantizer4.Text += Top1.ToString() + " "; } // Update the loading form Loading.LoadProgressValue += SizeOfQuantizer[3] - 2; Loading.UpdateAndIncrement(); this.Update(); } else {// Show an error // Show an error } break; } break; } // Warker ffdd : Define restart interval, 4 bytes // Marker ffdd : Define restart interval, 4 bytes // Marker ffdd : Define restart interval, 4 bytes // Marker ffdd : Define restart interval, 4 bytes // Read out 4 bytes // Read out 4 bytes Top1 = HexChars[Count]; Count++; Bottom1 = HexChars[Count]; Count++; Bottom1 = HexChars[Count]; Count++; FrameSize = SetByteValue(Top1, Bottom1); FrameSize = FrameSize &lt;&lt; 8; // to get the rest of the counter Top1 = HexChars[Count]; Count++; Bottom1 = HexChars[Count]; Count++;</pre>		<pre>// Read out the table data while(FrameSize &gt; 0) {</pre>		<pre>break; }</pre>		
<pre>// Read out 4 bytes Dottom1 = HexChars[Count]; Count++; FrameSize; txtQuantizer4.Text += Top1.ToString() +</pre>		<pre>// We are counting down the FrameSize</pre>	to start the	case 'd': { // Mark	er ffdd : Define restart interval,	4 bytes
<pre>Gotton1, FrameSize; txtQuantizer4.Text += Top1.ToString() +</pre>		Count+; Bottom1 = HexChars[Count];		// Read Top1 = H Count++	out 4 bytes lexChars[Count];	
<pre>} } FrameSize = FrameSize &lt;&lt; 8; // Update the loading form Loading.UpdateAndIncrement(); this.Update(); else { // Show an error } break; } </pre> FrameSize += SetByteValue(Top1, Bottom1); Frame		FrameSize; txtQuantizer4.Text += Top1.ToString()	+	Bottoml Count++; FrameSiz	<pre>= HexChars[Count]; re = SetByteValue(Top1 Bottom1);</pre>	
<pre>Loading.B</pre>		<pre>} // Update the loading form Loading LoadProgrossValue += SizeOfOuan</pre>	tizor[3] = 2.	FrameSiz	<pre>ic SetEycevalue(lopi) SetEemily ic = FrameSize &lt;&lt; 8;</pre>	f the counter
<pre>selse {     else     {         // Show an error      }         break;     } </pre> <pre>break; } </pre> <pre>count++; Bottom1 = HexChars[Count]; Count++; FrameSize += SetByteValue(Top1, Bottom1); </pre>		<pre>Loading.DoadingGlessvarue '= Sizeorgaan Loading.UpdateAndIncrement(); this.Update();</pre>		Count++; Bottoml	= HexChars[Count];	
<pre>// Show an error // to get the rest of the counter Top1 = HexChars[Count]; Count++; Bottom1 = HexChars[Count]; Count++; FrameSize += SetByteValue(Top1, Bottom1); </pre>		else {		FrameSiz FrameSiz	<pre>te += SetByteValue(Top1, Bottom1); te = FrameSize &lt;&lt; 8;</pre>	f the counter
break; } Bottom1 = Hexchars[Count]; Count++; FrameSize += SetByteValue(Top1, Bottom1);		<pre>// Show all effor }</pre>		Top1 = H Count++;	<pre>// to get the rest of lexChars[Count]; // to get the rest of // to get the rest of</pre>	r the counter
Disamalises Disamalises // U.	}	break;		Bottomi Count+; FrameSiz	<pre>= nexcnars[count]; ce += SetByteValue(Top1, Bottom1); ce = EnomoSize &lt;&lt; 2;</pre>	

```
frmMain.cs
                                   frmMain.cs
 May 02, 04 2:03
                                                                     Page 93/186
                                                                                       May 02, 04 2:03
                                                                                                                                                           Page 94/186
                                         // to get the rest of the counter
                  Top1 = HexChars[Count];
                                                                                                        // Read out 3 bytes
                                                                                                        Top1 = HexChars[Count];
                  Count++;
                  Bottom1 = HexChars[Count];
                                                                                                        Count++;
                                                                                                        Bottom1 = HexChars[Count];
                  Count++;
                  FrameSize += SetByteValue(Top1, Bottom1);
                                                                                                        Count++;
                                                                                                        FrameSize = SetByteValue(Top1, Bottom1);
                  // Store Restart Data
                                                                                                        FrameSize = FrameSize << 8;</pre>
                  RestartInterval = FrameSize;
                                                                                                                               // to get the rest of the counter
                                                                                                        Top1 = HexChars[Count];
                  // Update the loading form
                                                                                                        Count++;
                  Loading.LoadProgressValue += 2;
                                                                                                        Bottom1 = HexChars[Count];
                  Loading.UpdateAndIncrement();
                                                                                                        Count++;
                  this.Update();
                                                                                                        FrameSize += SetByteValue(Top1, Bottom1);
                                                                                                        FrameSize = FrameSize << 8;</pre>
                  lblRestartMarker.Text = "ffdd";
                                                                                                                               // to get the rest of the counter
                  txtRestart.Text = FrameSize.ToString();
                                                                                                        Top1 = HexChars[Count];
                  FrameSize = 0;
                                                                                                        Count++;
                                                                                                        Bottom1 = HexChars[Count];
                  break;
                                                                                                        Count++;
                                                                                                        FrameSize += SetByteValue(Top1, Bottom1);
                case 'e':
                { // Marker ffde : Define Hierarchial Progression
                                                                                                        // Update the loading form
                                                                                                        Loading.LoadProgressValue += 1;
                  lblHierarchialMarker.Text = "ffde";
                                                                                                        Loading.UpdateAndIncrement();
                                                                                                        this.Update();
                  // Read in the Frame Size to set values
                  Top1 = HexChars[Count];
                                                                                                        // Store the data
                  Count++;
                                                                                                        ExpandImage = FrameSize;
                  Bottom1 = HexChars[Count];
                  Count++;
                                                                                                        lblExpandMarker.Text = "ffdf";
                  FrameSize = SetByteValue(Top1, Bottom1);
                                                                                                        txtExpand.Text = FrameSize.ToString();
                  FrameSize = FrameSize << 8;</pre>
                                                                                                        FrameSize = 0;
                                         // to get the rest of the counter
                                                                                                        break;
                  Top1 = HexChars[Count];
                  Count++;
                  Bottom1 = HexChars[Count];
                                                                                                      default:
                  Count++;
                  FrameSize += SetByteValue(Top1, Bottom1);
                                                                                                        txtError.Text +=
                  SizeOfProgression = FrameSize;
                                                                                                          "\nError: Invalid File Marker Read!! " +
                  FrameSize -= 2;
                                                                                                          "\n\t-- Marker ffd" + Bottom1.ToString()+
                                        // For the 2 bytes that hold the frame s
                                                                                                          " was found at byte index: " +
ize
                                                                                                          ((int)(Count - 4)).ToString();
                                                                                                        txtError.Update();
                  while(FrameSize > 0)
                                                                                                        break;
                                                                                                      1
                  {
                    // We are counting down the FrameSize to start the stream.
                    Top1 = HexChars[Count];
                                                                                                    } // End of: switch(Bottom1)
                    Count++;
                    Bottom1 = HexChars[Count];
                                                                                                    break:
                    Count++;
                    FrameSize--;
                                                                                                  } // End of: case 'd': // marker ffdX
                    txtHierarchial.Text +=
                                                 Top1.ToString() + Bottom1.ToStri
ng() + " ";
                                                                                                  case 'e': // marker ffeX
                  }
                                                                                                         // e0 to ef - Reserved for application data
                  // Update the loading form
                                                                                                    if(lblApplicationMarker1.Text == "")
                  Loading.LoadProgressValue +=
                                                                                                                        lblApplicationMarker1.Text = "ffe" + Bottom1;
                                           ((txtHierarchial.Text.Length * 2)/3) -
                                                                                                    else if(lblApplicationMarker2.Text == "")
2;
                                                                                                                        lblApplicationMarker2.Text = "ffe" + Bottom1;
                  Loading.UpdateAndIncrement();
                                                                                                    else if(lblApplicationMarker3.Text == "")
                  this.Update();
                                                                                                                        lblApplicationMarker3.Text = "ffe" + Bottom1;
                                                                                                    else if(lblApplicationMarker4.Text == "")
                  break;
                                                                                                                        lblApplicationMarker4.Text = "ffe" + Bottom1;
                }
                                                                                                    else if(lblApplicationMarker5.Text == "")
                                                                                                                        lblApplicationMarker5.Text = "ffe" + Bottom1;
                case 'f':
                                                                                                    else if(lblApplicationMarker6.Text == "")
                   // Marker ffdf : Expand Reference Images, 3 bytes
                                                                                                                        lblApplicationMarker6.Text = "ffe" + Bottom1;
```

```
frmMain.cs
May 02, 04 2:03
                                                                   Page 95/186
             else if(lblApplicationMarker7.Text == "")
                                 lblApplicationMarker7.Text = "ffe" + Bottom1;
             else if(lblApplicationMarker8.Text == "")
                                 lblApplicationMarker8.Text = "ffe" + Bottom1;
             else if(lblApplicationMarker9.Text == "")
                                 lblApplicationMarker9.Text = "ffe" + Bottom1;
             else if(lblApplicationMarker10.Text == "")
                                 lblApplicationMarker10.Text = "ffe" + Bottom1;
             // Read in the Frame Size to set values
             Top1 = HexChars[Count];
             Count++;
             Bottom1 = HexChars[Count];
             Count++;
             FrameSize = SetByteValue(Top1, Bottom1);
             FrameSize = FrameSize << 8;
                               // to get the rest of the counter
             Top1 = HexChars[Count];
             Count++;
             Bottom1 = HexChars[Count];
             Count++;
             FrameSize += SetByteValue(Top1, Bottom1);
             FrameSize -= 2; // For the 2 bytes that hold the frame size
             // Update the loading form
             Loading.LoadProgressValue += 2;
             Loading.UpdateAndIncrement();
             this.Update();
             if(txtApplicationData1.Text == "")
               SizeOfAppData[0] = FrameSize + 2;
               while(FrameSize > 0)
                 // We are counting down the FrameSize to start the stream.
                 Top1 = HexChars[Count];
                 Count++;
                 Bottom1 = HexChars[Count];
                 Count++;
                 FrameSize--;
                 txtApplicationData1.Text += Top1.ToString() +
                                         Bottom1.ToString() + " ";
               // Update the loading form
               Loading.LoadProgressValue += SizeOfAppData[0] - 2;
               Loading.UpdateAndIncrement();
               this.Update();
             else if(txtApplicationData2.Text == "")
               SizeOfAppData[1] = FrameSize + 2;
               while (FrameSize > 0)
                 // We are counting down the FrameSize to start the stream.
                 Top1 = HexChars[Count];
                 Count++;
                 Bottom1 = HexChars[Count];
                 Count++;
                 FrameSize--:
                 txtApplicationData2.Text += Top1.ToString() +
                                         Bottom1.ToString() + " ";
               // Update the loading form
               Loading.LoadProgressValue += SizeOfAppData[1] - 2;
               Loading.UpdateAndIncrement();
               this.Update();
             else if(txtApplicationData3.Text == "")
```

```
frmMain.cs
May 02, 04 2:03
                                                                   Page 96/186
               SizeOfAppData[2] = FrameSize + 2;
               while(FrameSize > 0)
                 // We are counting down the FrameSize to start the stream.
                 Top1 = HexChars[Count];
                 Count++;
                 Bottom1 = HexChars[Count];
                Count++;
                 FrameSize--;
                txtApplicationData3.Text += Top1.ToString() +
                                         Bottom1.ToString() + " ";
               // Update the loading form
              Loading.LoadProgressValue += SizeOfAppData[2] - 2;
              Loading.UpdateAndIncrement();
              this.Update();
            else if(txtApplicationData4.Text == "")
              SizeOfAppData[3] = FrameSize + 2;
              while (FrameSize > 0)
                 // We are counting down the FrameSize to start the stream.
                 Top1 = HexChars[Count];
                Count++;
                Bottom1 = HexChars[Count];
                Count++;
                 FrameSize--;
                 txtApplicationData4.Text += Top1.ToString() +
                                         Bottom1.ToString() + " ";
               // Update the loading form
              Loading.LoadProgressValue += SizeOfAppData[3] - 2;
              Loading.UpdateAndIncrement();
              this.Update();
            else if(txtApplicationData5.Text == "")
               SizeOfAppData[4] = FrameSize + 2;
              while (FrameSize > 0)
                 // We are counting down the FrameSize to start the stream.
                Top1 = HexChars[Count];
                Count++;
                Bottom1 = HexChars[Count];
                Count++;
                 FrameSize--:
                 txtApplicationData5.Text += Top1.ToString() +
                                         Bottom1.ToString() + " ";
               // Update the loading form
              Loading.LoadProgressValue += SizeOfAppData[4] - 2;
              Loading.UpdateAndIncrement();
              this.Update();
            else if(txtApplicationData6.Text == "")
               SizeOfAppData[5] = FrameSize + 2;
              while(FrameSize > 0)
                // We are counting down the FrameSize to start the stream.
                Top1 = HexChars[Count];
                Count++;
                Bottom1 = HexChars[Count];
                 Count++;
                FrameSize--:
                 txtApplicationData6.Text += Topl.ToString() +
                                         Bottom1.ToString() + " ";
               1
```

May 02, 04 2:03	frmMain.cs	Page 97/186 M	ay 02, 04 2:03	frmMain.cs	Page 98/186
<pre>// Upd Loadir Loadir this.U } else if( { SizeOf while( { // W Topl Cour Bott Cour Fram txtA } // Upd Loadir this.U } else if( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf while( { SizeOf While( { SizeOf While( { SizeOf While( { SizeOf Sour Bott Cour Fram txtA } } // Upd Loadir Loadir Loadir Cour Fram txtA } } // Upd Cour Fram txtA } } // Upd Loadir Loadir Loadir Loadir Cour Fram txtA } } else if( { SizeOf While( { SizeOf Sott Cour Fram txtA } } Sott Cour Fram txtA } } Size( SizeOf Sott Cour Fram txtA } SizeOf Sott Cour Fram txtA } Size( SizeOf Sott Cour Fram txtA } Sott Sott Sott Sott Sott Sott Sott So</pre>	<pre>date the loading form ng.LoadProgressValue += SizeOfAppData[5] - ng.UpdateAndIncrement(); Update(); (txtApplicationData7.Text == "") fAppData[6] = FrameSize + 2; (FrameSize &gt; 0) We are counting down the FrameSize to star 1 = HexChars[Count]; nt++; meSize; ApplicationData7.Text += Topl.ToString() + Bottoml.ToString() + date the loading form ng.LoadProgressValue += SizeOfAppData[6] - ng.UpdateAndIncrement(); Update(); (txtApplicationData8.Text == "") fAppData[7] = FrameSize + 2; (FrameSize &gt; 0) We are counting down the FrameSize to star 1 = HexChars[Count]; nt++; meSize; ApplicationData8.Text += Topl.ToString() + Bottoml.ToString() + date the loading form ng.LoadProgressValue += SizeOfAppData[7] - ng.UpdateAndIncrement(); Update(); (txtApplicationData9.Text == "") fAppData[8] = FrameSize + 2; (FrameSize &gt; 0) We are counting down the FrameSize to star 1 = HexChars[Count]; nt++; meSize; ApplicationData9.Text == "") fAppData[8] = FrameSize + 2; (FrameSize &gt; 0) We are counting down the FrameSize to star 1 = HexChars[Count]; nt++; meSize; ApplicationData9.Text == "") fAppData[8] = FrameSize + 2; (FrameSize &gt; 0) We are counting down the FrameSize to star 1 = HexChars[Count]; nt++; toml = HexChars[Count]; nt++; toml = HexChars[Count]; nt++; (txtApplicationData9.Text += Topl.ToString() + Bottoml.ToString() +</pre>	- 2; t the stream. " "; - 2; t the stream. " "; - 2; ir t the stream. " "; - 2;	<pre>} // Lo Lo Lo th } case ' {     swit     {         cae         ca         ca         ca</pre>	Count++; Bottoml = HexChars[Count]; Count++; FrameSize; txtApplicationDatal0.Text += Topl.ToString Bottoml.ToString() 'Update the loading form bading.LoadProgressValue += SizeOfAppData[9 bading.UpdateAndIncrement(); his.Update(); tk; f': // marker fffX tch(Bottoml) use '0': goto case 'd'; use '1': goto case 'd'; use '2': goto case 'd'; use '3': goto case 'd'; use '3': goto case 'd'; use '3': goto case 'd'; use '6': goto case 'd'; use '8': goto case 'd'; use '8': goto case 'd'; use '8': goto case 'd'; use '8': goto case 'd'; use '1': goto case 'd'; use '2': goto case 'd'; use 'd': // marker fff0 to fffd: Reserved for JPEG Ext "Found!!\n\t Marker ff" + Topl.ToString() + ((int)(Count - 4)).ToString(); txtError.Update(); break; use 'e': // marker fffe - Comments // Read in the Frame Size to set values Topl = HexChars[Count]; Count++; Bottoml = HexChars[Count]; Count++; FrameSize = SetByteValue(Topl, Bottoml); FrameSize = FrameSize << 8;	<pre>() + + " "; ] - 2; extensions ensions Marker "+ g() + " was found at byte f the counter</pre>
SizeOf	<pre>fAppData[9] = FrameSize + 2; (FrameSize &gt; 0)</pre>	ize	•	-	
while( { // w	(FrameSize > 0) We are counting down the FrameSize to star	t the stream.		// Update the loading form Loading,LoadProgressValue += 2:	
Topl	<pre>1 = HexChars[Count];</pre>	e ene ocreant.		Loading.UpdateAndIncrement();	

May 02, 04 2:03	frmMain.cs	Page 99/186	May 02, 04 2:03	frmMain.cs	Page 100/186
	<pre>this.Update();</pre>		txtError.Text "\nError: In	+= valid Marker Found!! " +	
	<pre>while(FrameSize &gt; 0) {</pre>		"\n\t Mark	er ff" + Topl.ToString() + Botto	<pre>ml.ToString() +</pre>
	<pre>// We are counting down the FrameSize to st Top1 = HexChars[Count]; Count++; Bottom1 = HexChars[Count]; Count++; FrameSizo==:</pre>	tart the stream.	<pre>" was found. txtError.Updat ShowWarning( "\nLoad Oper ")</pre>	"; e(); ation was canceled" +	properly fromatted"+
	<pre>Temp = SetByteValue(Top1, Bottom1); txtComments.Text += (char)Temp; }</pre>		" to be a JP break; }	EG.");	Topolly Homaccou +
	<pre>// Update the loading form Loading.LoadProgressValue += txtComments.Text Loading.UpdateAndIncrement(); this.Update();</pre>	Length - 2;	} } // End of: while(C	ount < FileLeng)	
	break;		<pre>Loading.Dispose();</pre>		
3	,		} // End of: private v	oid LoadInterfaceData(ref jfile	HexChars)
I C	case 'f': // marker ffff Marker Not Defined				
ſ	<pre>txtError.Text +=    "\nError: Marker NOT defined " +    "\n\t Marker ffff was found at byte inde:</pre>	<: " +	#endregion Methods to	Convert from Binary to ACSII	
	<pre>((int)(Count - 4)).ToString(); txtError.Update(); break;</pre>		#region Methods to Con	vert from ACSII to Binary	
} d	<pre>default: { txtError.Text += "\nError: Invalid File Marker Read!! " + "\n\t Marker ffd" + Bottom1.ToString()+ " was found at byte index." +</pre>		<pre>/// <summary> /// Thie Method is use /// char value. The m /// if it 'a' to 'f' ( /// </summary> /// <param name="HexVa /// &lt;returns&gt;Returns T /// to 'f' (also 'b' t &lt;/pre&gt;&lt;/td&gt;&lt;td&gt;d to check if a char value is a&lt;br&gt;ethod returns TRUE if the char i&lt;br&gt;also 'A' to 'F'), otherwise FALS&lt;br&gt;lue"/>The CHAR value to check.RUE if the char is '0' to '9' or o 'F') otherwise FALSE is return</pre>	valid Hexadecimal s '0' to '9' or E is returned. param> : if it 'a' pred c(returns)	
}	<pre>((int)(Count - 4)).ToString(); txtError.Update(); break;</pre>		<pre>if (HexValue == '0'       HexValue == '3'        HexValue == '6'        HexValue == '6'    </pre>	<pre>x (char HexValue) HexValue == '1'    HexValue == HexValue == '4'    HexValue == HexValue == '7'    HexValue ==</pre>	: '2'    '5'    '8'
bre	eak;		{ return true;		
}			} else		
defau { txt " " ( txt bre }	<pre>alt: LError.Text += '\nError: Invalid File Marker Read!! " + '\n\t Marker ff" + Topl.ToString() + Bottoml ' was found at byte index: " + ((int)(Count - 4)).ToString(); LError.Update(); eak;</pre>	.ToString()+	<pre>{     HexValue = Char.To     if(HexValue == 'a'         HexValue == 'd'     {         return true;     }     else return false; } </pre>	Lower(HexValue);    HexValue == 'b'    HexValue    HexValue == 'e'    HexValue =	== 'c'    = 'f')
} // E	End of: switch(Top1)		/// <summary></summary>		
} // End else	d of: if(Top1 == 'f' && Bottom1 == 'f')		<pre>/// Pre-conditions: /// Post-conditions:</pre>	None.	
{ if(Show "\nIn "\nIm	Warning( nvalid File Marker Read!" + nage maybe damaged or image may not be properly "to be a JPEG \n\nLoad Operation Cancol	y fromatted "+	<pre>/// The LowBits and /// then combined t /// function. /// Description: /// The purpose of</pre>	HighBits parameters are convert o form the byte value that is re	ed to integers and turned by this
{	to be a orne. An Annoad Operation Cance.	LTEA: ))	/// convert two ASC	II characters, between 0 to F, t	to their binary

```
frmMain.cs
                                                                                                                         frmMain.cs
 May 02, 04 2:03
                                                                   Page 101/186
                                                                                       May 02, 04 2:03
                                                                                                                                                         Page 102/186
   111
           values and then combine them to form a one-byte value. This
   111
           function should call the Convert() method that takes a char and
   111
          returns a byte for each of these two parameters to get the
   111
           integer value of each. Then, it should combine both of these
                                                                                          /// <summary>
   111
           integer values to form one full byte value. Finally, this byte
                                                                                         /// Pre-conditions:
                                                                                                                 None.
   111
          value should be returned when the function exits.
                                                                                         /// Post-conditions:
   /// </summary>
                                                                                                All of the character data contained in each of the data TextBox
                                                                                         111
   /// <param name="HighBits">The HighBits parameter is an ASCII character
                                                                                         111
                                                                                                controls for the JPEG file is recombined and input, in order, into
   /// that represents a value of 0 to 15, in the form of 0 to F, for the 4
                                                                                         111
                                                                                                the File parameter passed.
   /// most significant bits of the byte that will be returned.</param>
                                                                                         111
                                                                                              Description:
   /// <param name="LowBits">The LowBits parameter is an ASCII character
                                                                                         111
                                                                                                The purpose of this method is to take all of the data currently
   /// that represents a value of 0 to 15, in the form of 0 to F, for the 4
                                                                                         111
                                                                                                 loaded in the ManipulatorM-^Rs interface and recombine these values
   /// least significant bits of the byte that will be returned.</param>
                                                                                         111
                                                                                                 into one large byte array. This byte array will contain all of the
                                                                                         111
   /// <returns>Function returns a byte value based upon the parameters
                                                                                                binary data in the exact form the as the current ASCII chars loaded
                                                                                         111
                                                                                                 in the data fields of the Manipulator. As such, this function is
   /// passed in.</returns>
                                                                                                one of the largest functions in the Manipulator and performs many tasks during its execution. This function should start dequeuing
   private byte SetByteValue(char HighBits, char LowBits)
                                                                                         111
                                                                                         111
     High = Convert(HighBits); // Get 4 high bits
                                                                                         111
                                                                                                and re-enqueuing the markers stored in the FileOrder Queue. For
     High = High << 4; // Shift up 4 bits
                                                                                         111
                                                                                                each file marker found in this queue, the data in the corresponding
     High += Convert(LowBits); // Add on the lower bits
                                                                                         111
                                                                                                 interface data TextBox should be processed. This function should
                                                                                         111
                                                                                                read the data from the particular TextBox, convert this data to
     return (byte)High;
                                                                                         111
                                                                                                binary and then input the resulting data into the File byte array
                                                                                         111
                                                                                                parameter passed into this function. Lastly, this method should do
                                                                                         111
                                                                                                lots error checking to make sure this function executes properly.
   /// <summary>
                                                                                         111
                                                                                                If an error is encountered, then the ShowWarning() method should be
    /// Pre-conditions:
                           None.
                                                                                          111
                                                                                                called to display the error to the user and the txtError TextBox
   /// Post-conditions:
                                                                                         111
                                                                                                control should be updated with this error information.
   111
          An integer representing the binary value of the hexadecimal ASCII
                                                                                         /// </summary>
   111
           character parameter passed will be returned.
                                                                                         /// <param name="File">The File parameter is storage space for the new
   111
                                                                                         /// file byte array. All the data for the new JPEG image will be based on
        Description:
   111
          The purpose of this function allows the caller to convert an ASCII
                                                                                         /// the conversion of the ASCII characters that are currently loaded in
   111
           character between 0 and F to its corresponding integer value of 0
                                                                                         /// all of the data fields of the ManipulatorM-^Rs interface.</param>
   111
          to 15. This function will return a M-^V1 if the char parameter
                                                                                         /// <returns>Function returns True if it completes successfully, else
   111
          passed in is not between the value of 0 and F and an error
                                                                                          /// False.</returns>
   111
          message will be displayed for the user.
                                                                                         private bool CreateManipulatedPicture(ref byte[] File)
   /// </summary>
                                                                                          { // Returns true if completed correctly.
   /// <param name="Hex">The Hex parameter is an ASCII character between 0 \,
                                                                                           try
   /// and F.</param>
   /// <returns)Function returns an int based upon the hexadecimal value of
                                                                                             Loading = new frmLoad();
   /// the char parameter.</returns>
   private int Convert(char Hex)
                                                                                             char A = 'f', B = 'f', C = 'X', D = 'X';
                                                                                             int count = 0, HuffmanNumber = 0, QuantizerNumber = 0;
     switch(Hex.ToString().ToLower()[0])
                                                                                                     int AppDataNumber = 0;
       case '0': return 0;
                                                                                             if (File != null) File = null;
       case '1': return 1;
                                                                                             File = new byte[MAX_BYTES];
       case '2': return 2;
       case '3': return 3;
                                                                                             Loading.StartLoading(0, FileSize, 1);
       case '4': return 4:
       case '5': return 5;
                                                                                             while (A == 'f' \&\& B == 'f')
       case '6': return 6;
       case '7': return 7:
                                                                                               A = (char)FileOrder.Dequeue();
        case '8': return 8;
                                                                                               B = (char)FileOrder.Dequeue();
       case '9': return 9;
                                                                                               C = (char)FileOrder.Dequeue();
       case 'a': return 10:
                                                                                               D = (char)FileOrder.Dequeue();
       case 'b': return 11;
                                                                                               FileOrder.Enqueue(A);
       case 'c': return 12;
                                                                                               FileOrder.Enqueue(B);
       case 'd': return 13;
                                                                                               FileOrder.Enqueue(C);
        case 'e': return 14;
                                                                                               FileOrder.Enqueue(D);
       case 'f': return 15;
       default:
                                                                                               NewData[count] = SetByteValue(A, B);
                                                                                               count++;
         ShowWarning(
                                                                                               NewData[count] = SetByteValue(C, D);
            "Function \"int Convert(char); \" encountered an unrecognized " +
                                                                                               count++;
                        "character!!\nThis is a SERIOUS error! Please inform dev
eloper.");
                                                                                                // Update the loading form
          return -1;
                                                                                               if (Loading.Canceled)
      }
                                                                                                  Loading.Dispose();
```

May 02, 04 2:03	frmMain.cs	Page 103/186 May (	2, 04 2:03	frmMain.cs	Page 104/186
return }	n false;		{ txtErro	r.Text +=	
Loading Loading this.Upo	.UpdateAndIncrement(); .UpdateAndIncrement(); date();		"\nEr "\n\t " was	<pre>ror: Invalid File Marker Read!! Marker ff0" + D.ToString()+ found in the original stream. " Marker and data</pre>	" + ' +
// If we if(A ==	e are at the end of the file, we'll break 'f' && B == 'f' && C == 'd' && D == '9') br	eak;	txtErro	r.Update();	A NOT WITCHEN TO NEW T
if(A == {	'f' && B == 'f')		}		
switch { //	h(C) JPEG FILE MARKERS, Pg 106 in "JPEG" by: // Pennebaker & Mitchell		} // End of break;	: switch(D)	
case	e '0': // Marker ff0X		} // End of:	case '0';	
{ 	witch(D)		case '1': got	o case 'b';	
{	case '0': // Marker ff00 - Marker Not Defin	ed	case '2': got case '3': got	o case 'b'; o case 'b'; o case 'b';	
	txtError.Text +=		case '5': got	o case 'b';	
1	"\nError: Marker NOT defined " +	iginal filo" +	case '6': got	o case 'b';	
	"\n\t Marker FF00 Was found in the or " stream! Marker an	d data NOT written	case '8': got	o case 'b'; o case 'b';	
to new file.";			case '9': got	o case 'b';	
	<pre>txtError.Update(); break.</pre>		case 'a': got	o case 'b';	
	}		{ // Marker f	f10 to ffbf - Reserved	
	case '1': // Marker ff01		txtError.Te	xt +=	
	{ tytError Text +=		"\nError: "\n\t M	Reserved Marker Found!! " + arker ff" + C ToString() + D ToS	String()+
	"\nError: Marker found Temporary use fo " Encoding\n\t Marker	r Arthmetic" + FF01 was found in	" was fou	nd in the original stream. " + " Marker and data NOT wri	itten to new file.";
the " +	"original file stre	am. Marker and data	txtError.Up break;	date();	
NOT WITCHIN I	<pre>"to new file."; txtError.Update();</pre>		case 'c': // :	marker ffcX - huffman tables	
	break;		{ bool Read =	true;	
	} case '2': goto case 'f':		switch(D)		
	case '3': goto case 'f';		{		
	case '4': goto case 'f';		// Start	of: Nondifferential Huffman-Codi	ing Frames
	case '6': goto case '1';		case '0':	// marker ficu - Baseline DCI	
	case '7': goto case 'f';		// Mani	pulated 01-18-2004	
	case '8': goto case 'f';		// Head	erSize = 2 because 2 bytes for s	size field
	case 'a': goto case 'f';		char To	p, Bottom;	
	case 'b': goto case 'f';		byte []	HeaderData = new byte[100];	
	case 'c': goto case 'f';			Provision 1 Puto	
	case 'd': goto case 'f'; case 'f':		txtPrec if(txtP	ision.Text = txtPrecision.Text.T recision.Text.Length < 2)	<pre>frim();</pre>
	{     // Marker ff02 to ff0f - Reserved     tytError Text +-		{ ShowW	arning("The Precision on the Hea	aders Tab, must" +
	<pre>"\nError: Reserved Marker Found!! " + "\n\t Marker ff0" + D.ToString()+ " was found in the original stream. " +</pre>		"Ra "Wa txtPr	ndom values will be added to sol rning, image data altered!"); ecision.Text = "00":	lve this problem!",
	" Marker and data N	OT written to new f	}		
ile.";	tytError Undato().		Top = t	xtPrecision.Text[0];	
	break;		HeaderD	<pre>- txtFrectston.text[1]; ata[HeaderSize] = SetBvteValue(]</pre>	[op, Bottom);
	}		HeaderS	ize++;	<u> </u>
	default:		// Upda	te the loading form	

```
frmMain.cs
                                                                                                                        frmMain.cs
 May 02, 04 2:03
                                                                   Page 105/186
                                                                                      May 02, 04 2:03
                                                                                                                                                         Page 106/186
                    Loading.UpdateAndIncrement();
                                                                                                         HeaderData[HeaderSize] = SetByteValue(Top, Bottom);
                    this.Update();
                                                                                                         HeaderSize++;
                    // Set Number Lines - 2 Bytes
                                                                                                         // Update the loading form
                                                                                                         Loading.UpdateAndIncrement();
                    txtNumberHuffmanLines.Text =
                                                txtNumberHuffmanLines.Text.Trim(
                                                                                                         this.Update();
);
                    if(txtNumberHuffmanLines.Text.Trim().Length < 5)</pre>
                                                                                                         int k = 0;
                      ShowWarning("The number of Lines on the Headers Tab, "+
                                                                                                         // Get rid of "Identifier, Horizontal, Vertical, Q-Table: \n
                                                                                     " at
                                                "must be EXACTLY 2 bytes!\n" +
                        "Random values will be added to solve this problem!",
                                                                                                         // the beginning of the control.
                        "Warning, image data altered!");
                                                                                                         string CData = txtComponents.Text.ToString();
                        txtNumberHuffmanLines.Text = "00 00";
                                                                                                         while (CData[k] != ' \setminus n') k++;
                                                                                                         k++:
                    Top = txtNumberHuffmanLines.Text[0];
                    Bottom = txtNumberHuffmanLines.Text[1];
                                                                                                         // Get all the component data
                    HeaderData[HeaderSize] = SetByteValue(Top, Bottom);
                                                                                                         CData = CData.Substring(k,
                                                                                                                                      (txtComponents.Text.Length - k))
                    HeaderSize++;
                    Top = txtNumberHuffmanLines.Text[3];
                    Bottom = txtNumberHuffmanLines.Text[4];
                                                                                                         k = 0;
                    HeaderData[HeaderSize] = SetByteValue(Top, Bottom);
                    HeaderSize++;
                                                                                                         // Get all of the components
                                                                                                         byte NewSize = 0;
                    // Update the loading form
                                                                                                         int SizeIndex = HeaderSize - 1;
                    Loading.UpdateAndIncrement();
                                                                                                         bool Done = false;
                    Loading.UpdateAndIncrement();
                    this.Update();
                                                                                                         while(k < CData.Length)
                    // Set Number of samples per line - 2 Bytes
                                                                                                           // Move to the next data
                    txtNumberHuffmanSamples.Text =
                                                                                                           while((CData[k] == ' ' || CData[k] == ','
                                                                                                                || CData[k] == ' \n' || CData[k] == ' \t')
                                                txtNumberHuffmanSamples.Text.Tri
m();
                                                                                                                && (k < CData.Length))
                    if(txtNumberHuffmanSamples.Text.Trim().Length < 5)</pre>
                                                                                                             k++;
                      ShowWarning("The number of Samples per Line on the
                                                                                                             if(!(k < CData.Length))
                                                  Headers Tab, must be EXACTLY 2
bytes!\n" +
                                                                                                               Done = true;
                        "Random values will be added to solve this problem!",
                                                                                                               break;
                        "Warning, image data altered!");
                      txtNumberHuffmanSamples.Text = "00 00";
                    Top = txtNumberHuffmanSamples.Text[0];
                                                                                                           if(Done) break;
                    Bottom = txtNumberHuffmanSamples.Text[1];
                    HeaderData[HeaderSize] = SetByteValue(Top, Bottom);
                                                                                                           // Get Component identifier - 1 byte
                    HeaderSize++;
                                                                                                           Top = CData[k];
                    Top = txtNumberHuffmanSamples.Text[3];
                                                                                                           k++;
                    Bottom = txtNumberHuffmanSamples.Text[4];
                                                                                                           Bottom = CData[k];
                    HeaderData[HeaderSize] = SetByteValue(Top, Bottom);
                                                                                                           k++;
                    HeaderSize++;
                                                                                                           HeaderData[HeaderSize] = SetByteValue(Top, Bottom);
                                                                                                           HeaderSize++;
                    // Update the loading form
                    Loading.UpdateAndIncrement();
                                                                                                           Loading.UpdateAndIncrement();
                    this.Update();
                                                                                                             && (k < CData.Length))
                    // Get number of image components - 1 Byte
                                                                                                             k++;
                    txtNumberImageComponents.Text =
                                                                                                             if(!(k < CData.Length))
                                                txtNumberImageComponents.Text.Tr
im();
                                                                                                               Done = true;
                    if(txtNumberImageComponents.Text.Length < 2)
                                                                                                               break;
                      ShowWarning("The Number of Image Components will be
                                                  calculated!\n",
                                                                                                           if (Done)
                        "Warning, image data altered!");
                      txtNumberImageComponents.Text = "00";
                                                                                                             Bottom = '0';
                                                                                                             HeaderData[HeaderSize] = SetByteValue(Top, Bottom);
                    Top = txtPrecision.Text[0];
                                                                                                             HeaderSize++;
                    Bottom = txtPrecision.Text[1];
                                                                                                             NewSize++;
```

May 02, 04 2:03	frmMain.cs	Page 107/186 May 0	2, 04 2:03	frmMain.cs	Page 108/186
	<pre>// Update the loading form Loading.UpdateAndIncrement(); this.Update();</pre>			<pre>// Get Quantization Table Selector - Top = CData[k]; k++;</pre>	· 1 byte
	break; }			Bottom = CData[k]; k++; HeaderData[HeaderSize] = SetByteValu	ue(Top, Bottom);
	// Get Horizontal and Vertical Samp	ling factor - 4 bits ea		HeaderSize++;	
ch, or	Top = CData[k]; k++;			<pre>// Update the loading form Loading.UpdateAndIncrement(); this.Update();</pre>	
	// For Horizontal and Vertical Samp	ling factor – 4 bits ea		NewSize++; }	
ch	<pre>while((CData[k] == ' '    CData[k] =</pre>	== ',' = '\t')		<pre>// Set the new Number of Components HeaderData[SizeIndex] = NewSize;</pre>	
	<pre>{     k++;     if(!(k &lt; CData.Length))     / </pre>			<pre>// Set the new Header Frame size HeaderData[0] = (byte)((HeaderSize &gt;&gt; HeaderData[1] = (byte)(HeaderSize % 25</pre>	8) % 256); 56);
	Done = true; break;			<pre>// Now copy the HeaderData for(int i = 0; i &lt; HeaderSize; i++) {</pre>	
	} if(Done)			<pre>NewData[count] = HeaderData[i]; count++;</pre>	
	Bottom = '0'; HeaderData[HeaderSize] = SetByteV HeaderSize++;	alue(Top, Bottom);		, Read = false; // Skip reading values a	it end of loop
	NewSize++;			// End of change break;	
	<pre>// Update the loading form Loading.UpdateAndIncrement(); this.Update();</pre>		} ca {	se '1': // marker ffc1 - Extended Sequ	ential DCT
	break; }		}	<pre>// Implemented generically in this ver break;</pre>	sion
	Bottom = CData[k];		ca {	se '2': // marker ffc2 - Progressive D	)CT
	k++; HeaderData[HeaderSize] = SetByteVal HeaderSize++:	ue(Top, Bottom);	}	<pre>// Implemented generically in this ver break;</pre>	sion
	while((CData[k] == ' '    CData[k] =	== ' '	ca	se '3': // marker ffc3 - Lossless (Seq	[uential)
	<pre>// CData[k] == '\n'    CData[k] ==</pre>	= '\t')	1	<pre>// Implemented generically in this ver break;</pre>	sion
	<pre>k++; if(!(k &lt; CData.Length)) {</pre>		//	End of: Nondifferential Huffman-Codin	ıg Frames
	Done = true;		ca	se '4': // marker ffc4 - Define Huffma	ın Marker
	} } if(Done)		1 1	<pre>// Implemented generically in this ver break;</pre>	sion
			,		
	HeaderData[HeaderSize] = SetByteV HeaderSize++;	alue(Top, Bottom);	// ca	Start of: Differential Huffman-Coding se '5': // marker ffc5 - Differential	/ Frames Sequential DCT
	// Update the loading form		1	<pre>// Implemented generically in this ver break;</pre>	sion
	<pre>Loading.UpdateAndIncrement(); this.Update();</pre>		} ca	se '6': // marker ffc6 - Differential	Progressive DCT
	break; }		ł	<pre>// Implemented generically in this ver break;</pre>	sion

```
frmMain.cs
                                                                                                                         frmMain.cs
 May 02, 04 2:03
                                                                   Page 109/186
                                                                                       May 02, 04 2:03
                                                                                                                                                         Page 110/186
                                                                                                          txtError.Text +=
                  case '7': // marker ffc7 - Differential Lossless
                                                                                                            "\nError: Invalid File Marker Read!! " +
                                                                                                            "\n\t-- Marker ffc" + D.ToString()+
                    // Implemented generically in this version
                                                                                                            " was found in original file stream. " +
                                                                                                            "Marker and data not written to new file.";
                    break;
                                                                                                          break:
                  // End of: Differential Huffman-Coding Frames
                  case '8': // marker ffc8 - Reserved for JPEG Extensions
                                                                                                      } // End of: switch(D)
                    txtError.Text +=
                                                                                                      if (Loading.Canceled)
                      "\nError: Reserved For JPEG Extensions Marker Found !! "+
                      "\n\t-- Marker ffcd" +
                                                                                                        Loading.Dispose();
                      " was found in the original file stream." +
                                                                                                        return false;
                                           "\nMarker and data not written to the
new file.";
                    txtError.Update();
                                                                                                      if(Read)
                    Read = false; // Skip reading values for this marker
                                                                                                        byte Byte1, Byte2;
                    break;
                                                                                                        int SizeIndex = count;
                                                                                                        // Move ahead of the size field
                  // Start of: Nondifferential Arithmetic-Coding Frames
                                                                                                        count++;
                  case '9': // marker ffc9 - Extended Sequential DCT
                                                                                                        count++;
                    // Implemented generically in this version
                                                                                                        if(HuffmanNumber == 0)
                    break;
                                                                                                          int t;
                  case 'a': // marker ffca - Progressive DCT
                                                                                                          string NewHuff = "";
                                                                                                          char Nibble;
                    // Implemented generically in this version
                                                                                                          // Update the table we're reading
                    break;
                                                                                                          HuffmanNumber++;
                  case 'b': // marker ffcb - Lossless (Sequential)
                                                                                                          // Read out the content of the TextBox and
                                                                                                          // check to get only the valid HEX value chars
                    // Implemented generically in this version
                                                                                                          for (int x = 0; x < txtHuffman1.Text.Length; x++)
                    break;
                  // End of: Nondifferential Arithmetic-Coding Frames
                                                                                                            Nibble = txtHuffman1.Text[x];
                                                                                                            if(IsValidHex(Nibble))
                                                                                                              NewHuff += Nibble.ToString();
                  case 'c': // marker ffcc -
                            //Define Arithmetic Conditioning Tables
                                                                                                          // Check to make sure the size of the new
                    // Implemented generically in this version
                                                                                                          // huffman table is correct and if not, fix
                    break;
                                                                                                          if((NewHuff.Length % 2) == 1)
                                                                                                           NewHuff += "0";
                                                                                                          // Recalculated the size of the field and
                  // Start of: Differential Arithmetic-Coding Frames
                                                                                                          // write back to the new file string
                  case 'd': // marker ffcd - Differential Sequential DCT
                                                                                                          // for the 2 bytes of size
                                                                                                          t = (NewHuff.Length + 4)/2;
                    // Implemented generically in this version
                                                                                                          Byte2 = (byte)(t \% 256);
                                                                                                          t >>= 8;
                    break;
                                                                                                          Byte1 = (byte)(t \% 256);
                  case 'e': // marker ffce - Differential Progressive DCT
                                                                                                          NewData[SizeIndex] = Byte1;
                                                                                                          SizeIndex++;
                    // Implemented generically in this version
                                                                                                          NewData[SizeIndex] = Byte2;
                    break;
                                                                                                          // Update the loading form
                  case 'f': // marker ffcf - Differential Lossless
                                                                                                          Loading.UpdateAndIncrement();
                                                                                                          Loading.UpdateAndIncrement();
                    // Implemented generically in this version
                                                                                                          this.Update();
                    break:
                                                                                                          // Now write the new huffman table to the
                  // End of: Differential Arithmetic-Coding Frames
                                                                                                          // NewData string.
                                                                                                          for (int x = 0; x < NewHuff.Length; x+=2)
                  default:
                                                                                                           NewData[count] = SetByteValue(
```

```
frmMain.cs
                                                                                                                         frmMain.cs
May 02, 04 2:03
                                                                   Page 111/186
                                                                                      May 02, 04 2:03
                                                                                                                                                          Page 112/186
                       NewHuff[x], NewHuff[x+1]);
                                                                                                            Nibble = txtHuffman3.Text[x];
                     count++;
                                                                                                            if(IsValidHex(Nibble))
                     Loading.UpdateAndIncrement();
                                                                                                              NewHuff += Nibble.ToString();
                     this.Update();
                                                                                                          // Check to make sure the size of the new
                 else if (HuffmanNumber == 1)
                                                                                                          // huffman table is correct and if not, fix
                                                                                                          if((NewHuff.Length % 2) == 1)
                   int t;
                                                                                                           NewHuff += "0";
                   string NewHuff = "";
                   char Nibble;
                                                                                                          // Recalculated the size of the field and
                                                                                                          // write back to the new file string
                   // Update the table we're reading
                                                                                                          // for the 2 bytes of size
                   HuffmanNumber++;
                                                                                                          t = (NewHuff.Length + 4)/2;
                                                                                                          Byte2 = (byte)(t % 256);
                   // Read out the content of the TextBox and
                                                                                                          t >>= 8;
                                                                                                         Bytel = (byte)(t % 256);
NewData[SizeIndex] = Bytel;
                   // check to get only the valid HEX value chars
                   for(int x = 0; x < txtHuffman2.Text.Length; x++)</pre>
                                                                                                          SizeIndex++;
                     Nibble = txtHuffman2.Text[x];
                                                                                                         NewData[SizeIndex] = Byte2;
                     if(IsValidHex(Nibble))
                       NewHuff += Nibble.ToString();
                                                                                                          // Update the loading form
                                                                                                          Loading.UpdateAndIncrement();
                                                                                                          Loading.UpdateAndIncrement();
                   // Check to make sure the size of the new
                                                                                                          this.Update();
                   // huffman table is correct and if not, fix
                   if((NewHuff.Length % 2) == 1)
                                                                                                          // Now write the new huffman table to the
                    NewHuff += "0";
                                                                                                          // NewData string.
                                                                                                          for(int x = 0; x < NewHuff.Length; x+=2)
                   // Recalculated the size of the field and
                   // write back to the new file string
                                                                                                            NewData[count] = SetByteValue(
                   // for the 2 bytes of size
                                                                                                             NewHuff[x], NewHuff[x+1]);
                   t = (NewHuff.Length + 4)/2;
                                                                                                            count++;
                   Byte2 = (byte) (t % 256);
                                                                                                            Loading.UpdateAndIncrement();
                   t >>= 8;
                                                                                                            this.Update();
                   Byte1 = (byte) (t % 256);
                   NewData[SizeIndex] = Byte1;
                   SizeIndex++;
                                                                                                        else if (HuffmanNumber == 3)
                   NewData[SizeIndex] = Byte2;
                                                                                                         int t;
                   // Update the loading form
                                                                                                         string NewHuff = "";
                   Loading.UpdateAndIncrement();
                                                                                                         char Nibble;
                   Loading.UpdateAndIncrement();
                   this.Update();
                                                                                                          // Update the table we're reading
                                                                                                          HuffmanNumber++;
                   // Now write the new huffman table to the
                   // NewData string.
                                                                                                          // Read out the content of the TextBox and
                   for (int x = 0; x < NewHuff.Length; x+=2)
                                                                                                          // check to get only the valid HEX value chars
                                                                                                          for (int x = 0; x < txtHuffman4.Text.Length; x++)
                     NewData[count] = SetByteValue(
                       NewHuff[x], NewHuff[x+1]);
                                                                                                           Nibble = txtHuffman4.Text[x];
                     count++;
                                                                                                           if(IsValidHex(Nibble))
                     Loading.UpdateAndIncrement();
                                                                                                              NewHuff += Nibble.ToString();
                     this.Update();
                                                                                                          // Check to make sure the size of the new
                 else if(HuffmanNumber == 2)
                                                                                                          // huffman table is correct and if not, fix
                                                                                                          if((NewHuff.Length % 2) == 1)
                   int t;
                                                                                                           NewHuff += "0";
                   string NewHuff = "";
                   char Nibble;
                                                                                                          // Recalculated the size of the field and
                                                                                                          // write back to the new file string
                   // Update the table we're reading
                                                                                                          // for the 2 bytes of size
                   HuffmanNumber++;
                                                                                                          t = (NewHuff.Length + 4)/2;
                                                                                                          Byte2 = (byte)(t \% 256);
                   // Read out the content of the TextBox and
                                                                                                          t >>= 8;
                   // check to get only the valid HEX value chars
                                                                                                          Byte1 = (byte)(t % 256);
                   for (int x = 0; x < txtHuffman3.Text.Length; x++)
                                                                                                          NewData[SizeIndex] = Byte1;
                                                                                                          SizeIndex++;
```

```
frmMain.cs
                                                                                                                          frmMain.cs
May 02, 04 2:03
                                                                   Page 113/186
                                                                                       May 02, 04 2:03
                                                                                                                                                           Page 114/186
                   NewData[SizeIndex] = Byte2;
                                                                                                           int t;
                   // Update the loading form
                                                                                                          string NewHuff = "";
                   Loading.UpdateAndIncrement();
                                                                                                          char Nibble;
                   Loading.UpdateAndIncrement();
                   this.Update();
                                                                                                           // Update the table we're reading
                                                                                                          HuffmanNumber++;
                   // Now write the new huffman table to the
                   // NewData string.
                                                                                                          \ensuremath{{//}}\xspace Read out the content of the TextBox and
                   for (int x = 0; x < NewHuff.Length; x+=2)
                                                                                                           // check to get only the valid HEX value chars
                                                                                                          for(int x = 0; x < txtHuffman6.Text.Length; x++)</pre>
                     NewData[count] = SetByteValue(
                       NewHuff[x], NewHuff[x+1]);
                                                                                                            Nibble = txtHuffman6.Text[x];
                     count++;
                                                                                                             if(IsValidHex(Nibble))
                     Loading.UpdateAndIncrement();
                                                                                                              NewHuff += Nibble.ToString();
                     this.Update();
                                                                                                          // Check to make sure the size of the new
                 else if (HuffmanNumber == 4)
                                                                                                          // huffman table is correct and if not, fix
                                                                                                          if((NewHuff.Length % 2) == 1)
                                                                                                            NewHuff += "0";
                   int t;
                   string NewHuff = "";
                                                                                                          // Recalculated the size of the field and
                   char Nibble;
                                                                                                          // write back to the new file string
                   // Update the table we're reading
                                                                                                          // for the 2 bytes of size
                   HuffmanNumber++;
                                                                                                          t = (NewHuff.Length + 4)/2;
                                                                                                          Byte2 = (byte)(t % 256);
                   // Read out the content of the TextBox and
                                                                                                          t >>= 8;
                   // check to get only the valid HEX value chars
                                                                                                          Byte1 = (byte)(t % 256);
NewData[SizeIndex] = Byte1;
                   for (int x = 0; x < txtHuffman5.Text.Length; x++)
                                                                                                          SizeIndex++;
                     Nibble = txtHuffman5.Text[x];
                                                                                                          NewData[SizeIndex] = Byte2;
                     if(IsValidHex(Nibble))
                       NewHuff += Nibble.ToString();
                                                                                                           // Update the loading form
                                                                                                          Loading.UpdateAndIncrement();
                                                                                                          Loading.UpdateAndIncrement();
                   // Check to make sure the size of the new
                                                                                                          this.Update();
                   // huffman table is correct and if not, fix
                   if((NewHuff.Length % 2) == 1)
                                                                                                           // Now write the new huffman table to the
                     NewHuff += "0";
                                                                                                           // NewData string.
                                                                                                           for(int x = 0; x < NewHuff.Length; x+=2)
                   // Recalculated the size of the field and
                   // write back to the new file string
                                                                                                            NewData[count] = SetBvteValue(
                   // for the 2 bytes of size
                                                                                                              NewHuff[x], NewHuff[x+1]);
                   t = (NewHuff.Length + 4)/2;
                                                                                                             count++;
                   Byte2 = (byte)(t \% 256);
                                                                                                             Loading.UpdateAndIncrement();
                   t >>= 8;
                                                                                                             this.Update();
                   Byte1 = (byte) (t % 256);
                   NewData[SizeIndex] = Byte1;
                   SizeIndex++;
                                                                                                        else if (HuffmanNumber == 6)
                   NewData[SizeIndex] = Byte2;
                                                                                                          int t;
                   // Update the loading form
                                                                                                          string NewHuff = "";
                   Loading.UpdateAndIncrement();
                                                                                                          char Nibble;
                   Loading.UpdateAndIncrement();
                   this.Update();
                                                                                                           // Update the table we're reading
                                                                                                          HuffmanNumber++;
                   // Now write the new huffman table to the
                   // NewData string.
                                                                                                          // Read out the content of the TextBox and
                   for (int x = 0; x < NewHuff.Length; x+=2)
                                                                                                           // check to get only the valid HEX value chars
                                                                                                          for (int x = 0; x < txtHuffman7.Text.Length; x++)
                     NewData[count] = SetByteValue(
                       NewHuff[x], NewHuff[x+1]);
                                                                                                            Nibble = txtHuffman7.Text[x];
                     count++;
                                                                                                            if(IsValidHex(Nibble))
                     Loading.UpdateAndIncrement();
                                                                                                               NewHuff += Nibble.ToString();
                     this.Update();
                                                                                                           // Check to make sure the size of the new
                 else if (HuffmanNumber == 5)
                                                                                                           // huffman table is correct and if not, fix
```

```
frmMain.cs
                                                                                                                          frmMain.cs
May 02, 04 2:03
                                                                    Page 115/186
                                                                                       May 02, 04 2:03
                                                                                                                                                            Page 116/186
                   if((NewHuff.Length % 2) == 1)
                                                                                                           // Now write the new huffman table to the
                     NewHuff += "0";
                                                                                                           // NewData string.
                                                                                                           for(int x = 0; x < NewHuff.Length; x+=2)
                   // Recalculated the size of the field and
                   // write back to the new file string
                                                                                                             NewData[count] = SetByteValue(
                   // for the 2 bytes of size
                                                                                                               NewHuff[x], NewHuff[x+1]);
                   t = (NewHuff.Length + 4)/2;
                                                                                                             count++;
                   Byte2 = (byte) (t % 256);
                                                                                                             Loading.UpdateAndIncrement();
                   t<sup>>>= 8;</sup>
                                                                                                             this.Update();
                   Byte1 = (byte)(t % 256);
                   NewData[SizeIndex] = Byte1;
                   SizeIndex++;
                                                                                                         else
                   NewData[SizeIndex] = Byte2;
                                                                                                           txtError.Text +=
                   // Update the loading form
                                                                                                             "\nError: Too Many Huffman Tables!! " +
"\n\t-- Marker ff" + C.ToString() + D.ToString()+
                   Loading.UpdateAndIncrement();
                   Loading.UpdateAndIncrement();
                                                                                                             " was found in the original stream. " +
                   this.Update();
                                                                                                                                  " Marker and data NOT written to new f
                                                                                      ile.";
                   // Now write the new huffman table to the
                                                                                                           txtError.Update();
                   // NewData string.
                                                                                                           return false;
                   for (int x = 0; x < NewHuff.Length; x+=2)
                     NewData[count] = SetByteValue(
                                                                                                       } // End of: if(Read);
                       NewHuff[x], NewHuff[x+1]);
                                                                                                       else
                     count++;
                     Loading.UpdateAndIncrement();
                                                                                                         Read = true;
                     this.Update();
                                                                                                       break:
                 else if (HuffmanNumber == 7)
                                                                                                     } // End of: case 'c': // marker ffcX
                                                                                                     case 'd': // marker ffdX
                   int t;
                   string NewHuff = "";
                   char Nibble;
                                                                                                       switch(D)
                   // Update the table we're reading
                                                                                                         case '0': goto case '7';
                   HuffmanNumber++;
                                                                                                         case '1': goto case '7';
                                                                                                         case '2': goto case '7'
                   // Read out the content of the TextBox and
                                                                                                         case '3': goto case '7';
                   // check to get only the valid HEX value chars
                                                                                                         case '4': goto case '7';
                   for(int x = 0; x < txtHuffman8.Text.Length; x++)</pre>
                                                                                                         case '5': goto case '7';
                                                                                                         case '6': goto case '7';
                     Nibble = txtHuffman8.Text[x];
                                                                                                         case '7':
                     if(IsValidHex(Nibble))
                                                                                                         { // Marker ffd0 to ffd7
                       NewHuff += Nibble.ToString();
                                                                                                           if (Loading.Canceled)
                   // Check to make sure the size of the new
                                                                                                             Loading.Dispose();
                   // huffman table is correct and if not, fix
                                                                                                             return false;
                   if((NewHuff.Length % 2) == 1)
                     NewHuff += "0";
                                                                                                           string NewValue = "";
                   // Recalculated the size of the field and
                                                                                                           char Nibble;
                   // write back to the new file string
                                                                                                           for (int x = 0; x < txtRestartMod8.Text.Length; x += 3)
                   // for the 2 bytes of size
                   t = (NewHuff.Length + 4)/2;
                                                                                                             // Check to make sure the values are correct
                                                                                                             Nibble = txtRestartMod8.Text[x];
                   Byte2 = (byte)(t \% 256);
                   t >>= 8;
                                                                                                             if(IsValidHex(Nibble))
                   Byte1 = (byte) (t % 256);
                                                                                                               NewValue += Nibble.ToString();
                   NewData[SizeIndex] = Byte1;
                   SizeIndex++;
                   NewData[SizeIndex] = Byte2;
                                                                                                           // Make sure the new length is long enough
                                                                                                           if (NewValue.Length < 4)
                                                                                                             NewValue += "0" + "0" + "0" + "0";
                   // Update the loading form
                   Loading.UpdateAndIncrement();
                   Loading.UpdateAndIncrement();
                                                                                                           // Write the new values to the NewData
                   this.Update();
                                                                                                           for (int x = 0; x < 4; x += 2)
```

May 02, 04 2:03	frmMain.cs	Page 117/186	May 02, 04 2:03	frmMain.cs	Page 118/186
	<pre>NewData[count] = SetByteValue( NewValue[x], NewValue[x+1]); count++;</pre>			<pre>// Write the new Scan Header to NewData for(int x = 0; x &lt; NewScan.Length; x += 2)</pre>	
	<pre>// Update the loading form Loading.UpdateAndIncrement(); this.Update();</pre>			<pre>{     NewData[count] = SetByteValue(         NewScan[x], NewScan[x+1]);         count++;</pre>	
	; break; }			<pre>Loading.UpdateAndIncrement(); this.Update(); }</pre>	
	<pre>case '8': { // Marker ffd8 : Start of Image     break;</pre>			<pre>// Check for loading canceled if(Loading.Canceled) {</pre>	
	<pre>} case '9': { // Marker ffd9 · End of image</pre>			Loading.Dispose(); return false; }	
	// Covered by: case ffda break;			// Get Encoded Stream // // UNSAFE - These values ARE ASSUMED VALIE	
	case 'a': { // Marker ffda : Start of Scan			<pre>// since they cannot be altered by the int for(int x = 0; x &lt; EncodedData.Length; x + {     NewData[count] = SetByteValue(</pre>	erface = 2)
	<pre>byte Byte1, Byte2; int SizeIndex = count; int t;</pre>			<pre>EncodedData[x], EncodedData[x+1]); count++; // Check for loading canceled</pre>	
	<pre>// Check for loading canceled if(Loading.Canceled) {</pre>			<pre>if (Loading.Canceled) {    Loading.Dispose();</pre>	
	Loading.Dispose(); return false; }			return false; } else {	
	<pre>// Move past the size field count++; count++;</pre>			<pre>Loading.UpdateAndIncrement(); this.Update(); }</pre>	
	<pre>char Nibble; string NewScan = "";</pre>			break; }	
	<pre>// Get Scan Header for(int x = 0; x &lt; txtScanHeader.Text.Leng {</pre>	gth; x++)		<pre>case 'b': { // Marker ffdb : Define Quantization Tabl hute Putel:</pre>	e
	<pre>if(IsValidHex(Nibble))     NewScan += Nibble.ToString(); }</pre>			byte Byte2; int SizeIndex = count; int t;	
	<pre>// Check to make sure the new size is vali if((NewScan.Length % 2) == 1) NewScan += "0";</pre>	.d		char Nibble;	
	<pre>// Calculate new Scan Header size t = ((NewScan.Length + 4)/2); Byte2 = (byte)(t % 256); t &gt;&gt;= 8:</pre>			<pre>if(Loading.Canceled) {    Loading.Dispose();    return false; }</pre>	
	Bytel = (byte)(t % 256); NewData[SizeIndex] = Bytel; SizeIndex++; NewData[SizeIndex] = Byte2:			// Move past the size field count++; count++:	
	<pre>// Update Loading Form Loading.UpdateAndIncrement(); Loading.UpdateAndIncrement(); this.Update();</pre>			<pre>if(QuantizerNumber == 0) {     // Update the table we're reading     OuantizerNumber++:</pre>	

```
frmMain.cs
                                                                                                                         frmMain.cs
May 02, 04 2:03
                                                                   Page 119/186
                                                                                      May 02, 04 2:03
                                                                                                                                                          Page 120/186
                                                                                                            for(int x = 0; x < txtQuantizer2.Text.Length; x++)</pre>
                     // Get the table number
                     if(txtQuantizerTableNum1.Text.Length < 2)
                                                                                                              Nibble = txtQuantizer2.Text[x];
                       txtOuantizerTableNum1.Text = "0" + "0";
                                                                                                              if(IsValidHex(Nibble))
                     Nibble = txtQuantizerTableNum1.Text[0];
                                                                                                                NewQuant += Nibble.ToString();
                     if(!IsValidHex(Nibble)) Nibble = '0';
                     NewQuant += Nibble;
                     Nibble = txtQuantizerTableNum1.Text[1];
                                                                                                            // Check to make sure the size of the new
                     if(!IsValidHex(Nibble)) Nibble = '0';
                                                                                                            // huffman table is correct and if not, fix
                     NewQuant += Nibble;
                                                                                                            if((NewQuant.Length % 2) == 1)
                                                                                                              NewQuant += "0";
                     // Read out the content of the TextBox and
                     // check to get only the valid HEX value chars
                                                                                                            // Recalculated the size of the field and
                     for(int x = 0; x < txtQuantizer1.Text.Length; x++)</pre>
                                                                                                            // write back to the new file string
                                                                                                            // for the 2 bytes of size
                       Nibble = txtQuantizer1.Text[x];
                                                                                                            t = (NewQuant.Length + 4)/2;
                                                                                                            Byte2 = (byte)(t \ \% \ 256);
                       if(IsValidHex(Nibble))
                         NewQuant += Nibble.ToString();
                                                                                                            t >>= 8;
                                                                                                            Byte1 = (byte)(t % 256);
                                                                                                            NewData[SizeIndex] = Byte1;
                     // Check to make sure the size of the new
                                                                                                            SizeIndex++;
                     // huffman table is correct and if not, fix
                                                                                                            NewData[SizeIndex] = Byte2;
                     if((NewQuant.Length % 2) == 1)
                       NewQuant += "0";
                                                                                                            // Update the loading form
                                                                                                            Loading.UpdateAndIncrement();
                     // Recalculated the size of the field and
                                                                                                            Loading.UpdateAndIncrement();
                     // write back to the new file string
                                                                                                            this.Update();
                     // for the 2 bytes of size
                     t = (NewQuant.Length + 4)/2;
                                                                                                            // Now write the new huffman table to the
                                                                                                            // NewData string.
                     Byte2 = (byte)(t % 256);
                     t >>= 8;
                                                                                                            for(int x = 0; x < NewQuant.Length; x+=2)</pre>
                     Byte1 = (byte)(t % 256);
                     NewData[SizeIndex] = Byte1;
                                                                                                              NewData[count] = SetByteValue(
                     SizeIndex++;
                                                                                                                NewQuant[x], NewQuant[x+1]);
                     NewData[SizeIndex] = Byte2;
                                                                                                              count++;
                                                                                                              Loading.UpdateAndIncrement();
                     // Update the loading form
                                                                                                              this.Update();
                     Loading.UpdateAndIncrement();
                     Loading.UpdateAndIncrement();
                     this.Update();
                                                                                                          else if (OuantizerNumber == 2)
                     // Now write the new huffman table to the
                                                                                                            // Update the table we're reading
                     // NewData string.
                                                                                                            OuantizerNumber++;
                     for (int x = 0; x < NewQuant.Length; x + = 2)
                                                                                                            // Get the table number
                       NewData[count] = SetByteValue(
                                                                                                            if(txtQuantizerTableNum3.Text.Length < 2)
                         NewQuant[x], NewQuant[x+1]);
                                                                                                              txtQuantizerTableNum3.Text = "0" + "2";
                                                                                                            Nibble = txtQuantizerTableNum3.Text[0];
                       count++;
                       Loading.UpdateAndIncrement();
                                                                                                            if(!IsValidHex(Nibble)) Nibble = '0';
                       this.Update();
                                                                                                            NewQuant += Nibble;
                                                                                                            Nibble = txtQuantizerTableNum3.Text[1];
                                                                                                            if(!IsValidHex(Nibble)) Nibble = '0';
                   else if (QuantizerNumber == 1)
                                                                                                            NewOuant += Nibble;
                     // Update the table we're reading
                                                                                                            // Read out the content of the TextBox and
                     QuantizerNumber++;
                                                                                                            // check to get only the valid HEX value chars
                                                                                                            for(int x = 0; x < txtQuantizer3.Text.Length; x++)</pre>
                     // Get the table number
                     if(txtQuantizerTableNum2.Text.Length < 2)</pre>
                                                                                                              Nibble = txtQuantizer3.Text[x];
                       txtQuantizerTableNum2.Text = "0" + "1";
                                                                                                              if(IsValidHex(Nibble))
                     Nibble = txtQuantizerTableNum2.Text[0];
                                                                                                                NewQuant += Nibble.ToString();
                     if(!IsValidHex(Nibble)) Nibble = '0';
                     NewQuant += Nibble;
                     Nibble = txtQuantizerTableNum2.Text[1];
                                                                                                            // Check to make sure the size of the new
                     if(!IsValidHex(Nibble)) Nibble = '0';
                                                                                                            // huffman table is correct and if not, fix
                     NewQuant += Nibble;
                                                                                                            if((NewQuant.Length % 2) == 1)
                                                                                                              NewQuant += "0";
                     // Read out the content of the TextBox and
                     // check to get only the valid HEX value chars
                                                                                                            // Recalculated the size of the field and
```

May 02, 04 2:03	frmMain.cs	Page 121/186	May 02, 04 2:03	frmMain.cs	Page 122/186
	// write back to the new file string			<pre>this.Update();</pre>	
	<pre>// for the 2 bytes of size t = (NewQuant.Length + 4)/2; Byte2 = (byte)(t % 256); t &gt;&gt;= 8; Byte1 = (byte)(t % 256); NewData[SizeIndex] = Byte1; SizeIndex++; NewData[SizeIndex] = Byte2;</pre>			<pre>// Now write the new huffman table to the // NewData string. for(int x = 0; x &lt; NewQuant.Length; x+=2) {     NewData[count] = SetByteValue(         NewQuant[x], NewQuant[x+1]);         count++:</pre>	
	<pre>// Update the loading form Loading.UpdateAndIncrement(); Loading.UpdateAndIncrement(); this.Update();</pre>			<pre>Loading.UpdateAndIncrement();     this.Update();     }     else {</pre>	
	<pre>// Now write the new huffman table to the // NewData string. for(int x = 0; x &lt; NewQuant.Length; x+=2) { </pre>			<pre>// Output an error txtError.Text +=     "\nError: Too Many Quantizer Tables!! "     "\n\t Marker ff" + C.ToString() + D.T</pre>	+ oString()+
	<pre>NewData[count] = SetByteValue( NewQuant[x], NewQuant[x+1]);</pre>			" was found in the original stream. " + "Marker and dat	a NOT written to
	<pre>count++; Loading.UpdateAndIncrement(); this.Update(); }</pre>		new file.";	<pre>txtError.Update();   return false; }</pre>	
	$i = \frac{1}{2}$			, broak.	
	{			<pre>}</pre>	
	<pre>// Update the table we're reading QuantizerNumber++;</pre>			<pre>case 'c': { // Marker ffdc : Define number of lines, 4</pre>	bytes
	<pre>// Get the table number if(txtQuantizerTableNum4.Text.Length &lt; 2) txtQuantizerTableNum4.Text = "0" + "3"; Nibble = txtQuantizerTableNum4.Text[0]; if(!IsValidHex(Nibble)) Nibble = '0'; NewQuant += Nibble; Nibble = txtQuantizerTableNum4.Text[1]; if(!IsValidHex(Nibble)) Nibble = '0'; NewQuant += Nibble;</pre>			<pre>byte Byte1; byte Byte2; byte Byte3; byte Byte4; int t; t = NumberOfLines;</pre>	
	<pre>// Read out the content of the TextBox and // check to get only the valid HEX value ch for(int x = 0; x &lt; txtQuantizer4.Text.Lengt { Nibble = txtQuantizer4.Text[x]; if(IsValidHex(Nibble)) NewQuant += Nibble.ToString();</pre>	ars h; x++)		Byte4 = (byte)(t % 256); Byte3 = (byte)(t % 256); t >>= 8; Byte2 = (byte)(t % 256); t >>= 8; Byte1 = (byte)(t % 256);	;
	}			<pre>NewData[count] = Byte1; count++;</pre>	
	<pre>// Check to make sure the size of the new // huffman table is correct and if not, fix if((NewQuant.Length % 2) == 1) NewQuant += "0";</pre>			<pre>Loading.UpdateAndIncrement(); NewData[count] = Byte2; count++; Loading.UpdateAndIncrement();</pre>	
	<pre>// Recalculated the size of the field and // write back to the new file string // for the 2 bytes of size t = (NewQuant.Length + 4)/2; Byte2 = (byte)(t % 256);</pre>			<pre>NewData[count] = Bytes; count++; Loading.UpdateAndIncrement(); NewData[count] = Byte4; count++; Loading.UpdateAndIncrement();</pre>	
	t >>= 8; Byte1 = (byte)(t % 256); NewData[SizeIndex] = Byte1:			<pre>this.Update();</pre>	
	SizeIndex++;			break;	
	NewData[Sizeindex] = Bytez;			3	
	<pre>// Update the loading form Loading.UpdateAndIncrement(); Loading.UpdateAndIncrement();</pre>			<pre>case 'd': { // Marker ffdd : Define restart interval,</pre>	4 bytes

byte Byte1; byte Byte2; byte Byte3: Byte1 = (byte)(t %	% 256);
byte Byte4; int t; NewData[SizeIndex] NewData[SizeIndex]	<pre>% 256); ] = Byte1; ] = Byte2;</pre>
<pre>t = RestartInterval; // Update the load: Description of the second second</pre>	<pre>ding form Increment(); Increment(); values to NewData &lt; Progression.Length; x+=2) = SetByteValue( ], Progression[x+1]);</pre>
Loading.UpdateAndIncrement(); NewData[count] = Byte2; count++; Loading.UpdateAndIncrement(); NewData[count] = Byte3; count++; Loading.UpdateAndIncrement(); NewData[count] = Byte4; count++; Loading.UpdateAndIncrement(); NewData[count] = Byte4; count++; Loading.UpdateAndIncrement(); NewData[count] = Byte4; count++; Loading.UpdateAndIncrement(); Loading.UpdateAndIncrement();	ndIncrement(); f loading canceled ed) ();
<pre>this.Update();</pre>	Expand Reference Images, 3 bytes
<pre>{ // Marker ffde : Define Hierarchial Progression     byte Bytel;     byte Byte2;     int SizeIndex = count;     int t;     string Progression = "";     char Nibble;</pre>	es
<pre>// Check to see if loading canceled if(Loading.Canceled) { Loading.Dispose(); return false; } </pre> <pre>Byte3 = (byte)(t % t &gt;&gt;= 8; Byte2 = (byte)(t % byte1 = (byte)(t % } </pre>	<pre>% 256); % 256); % 256); Bute1.</pre>
<pre>// Move past the size field count++; count++; // Read out the contents of the interface for(int x = 0; x &lt; txtHierarchial.Text.Length; x++) { Nibble = txtHierarchial.Text[x]; if(IsValidHex(Nibble)) Progression += Nibble.ToString(); this.Update(); } NewData[count] = By count++; Loading.UpdateAndIi Count+; Loading.UpdateAndIi count++; Loading.UpdateAndIi count</pre>	<pre>Increment(); Byte2; Increment(); Byte3; Increment();</pre>
<pre>} // Check the size of the new field if((Progression.Length % 2) == 1) Progression += "0"; // Calculate the new size txtError.Text += txtError.Text +=</pre>	

```
frmMain.cs
                                                                                                                         frmMain.cs
May 02, 04 2:03
                                                                   Page 125/186
                                                                                       May 02, 04 2:03
                                                                                                                                                          Page 126/186
                     "\n\t-- Marker ffd" + D.ToString()+
                                                                                                            NewAppData[x], NewAppData[x+1]);
                     " was found in the original file stream. " +
                                                                                                          count++;
                     "Marker and data not written to the new file.";
                   txtError.Update();
                                                                                                          Loading.UpdateAndIncrement();
                   break;
                                                                                                          this.Update();
               } // End of: switch(D)
                                                                                                      else if (AppDataNumber == 1)
                                                                                                        AppDataNumber++;
               break;
             } // End of: case 'd': // marker ffdX
                                                                                                        // Read out the interface data
                                                                                                        for(int x = 0; x < txtApplicationData2.Text.Length; x++)</pre>
             case 'e': // marker ffeX
                                                                                                          Nibble = txtApplicationData2.Text[x];
                    // e0 to ef - Reserved for application data
                                                                                                          if(IsValidHex(Nibble))
               byte Bytel;
                                                                                                            NewAppData += Nibble.ToString();
               byte Byte2;
               int SizeIndex = count;
               int t;
                                                                                                        // Check the size of the new data
               string NewAppData = "";
                                                                                                        if((NewAppData.Length % 2) == 1)
               char Nibble;
                                                                                                          NewAppData += "0";
               // Check to see if loading canceled
                                                                                                        // Calculate the size field
               if(Loading.Canceled)
                                                                                                        t = ((NewAppData.Length + 4)/2);
                                                                                                        Byte2 = (byte)(t \% 256);
                                                                                                        t >>= 8;
                 Loading.Dispose();
                 return false;
                                                                                                        Byte1 = (byte)(t % 256);
                                                                                                        NewData[SizeIndex] = Byte1;
                                                                                                        SizeIndex++;
               // Move past size field
                                                                                                        NewData[SizeIndex] = Byte2;
               count++;
               count++;
                                                                                                        // Update the loading form
                                                                                                        Loading.UpdateAndIncrement();
               // Get the correct table
                                                                                                        Loading.UpdateAndIncrement();
               if(AppDataNumber == 0)
                                                                                                        this.Update();
                 AppDataNumber++;
                                                                                                        // Write the new values to NewData
                                                                                                        for(int x = 0; x < NewAppData.Length; x+=2)</pre>
                 // Read out the interface data
                 for(int x = 0; x < txtApplicationData1.Text.Length; x++)</pre>
                                                                                                          NewData[count] = SetByteValue(
                                                                                                            NewAppData[x], NewAppData[x+1]);
                   Nibble = txtApplicationData1.Text[x];
                                                                                                          count++;
                   if(IsValidHex(Nibble))
                     NewAppData += Nibble.ToString();
                                                                                                          Loading.UpdateAndIncrement();
                                                                                                          this.Update();
                 // Check the size of the new data
                 if((NewAppData.Length % 2) == 1)
                                                                                                      else if (AppDataNumber == 2)
                   NewAppData += "0";
                                                                                                        AppDataNumber++;
                 // Calculate the size field
                 t = ((NewAppData.Length + 4)/2);
                                                                                                        // Read out the interface data
                 Byte2 = (byte)(t % 256);
                                                                                                        for(int x = 0; x < txtApplicationData3.Text.Length; x++)</pre>
                 t >>= 8;
                 Byte1 = (byte)(t \% 256);
                                                                                                          Nibble = txtApplicationData3.Text[x];
                 NewData[SizeIndex] = Byte1;
                                                                                                          if(IsValidHex(Nibble))
                 SizeIndex++;
                                                                                                            NewAppData += Nibble.ToString();
                 NewData[SizeIndex] = Byte2;
                 // Update the loading form
                                                                                                        // Check the size of the new data
                 Loading.UpdateAndIncrement();
                                                                                                        if((NewAppData.Length % 2) == 1)
                 Loading.UpdateAndIncrement();
                                                                                                          NewAppData += "0";
                 this.Update();
                                                                                                        // Calculate the size field
                 // Write the new values to NewData
                                                                                                        t = ((NewAppData.Length + 4)/2);
                 for(int x = 0; x < NewAppData.Length; x+=2)</pre>
                                                                                                        Byte2 = (byte)(t % 256);
                                                                                                        t >>= 8;
                   NewData[count] = SetByteValue(
                                                                                                        Byte1 = (byte)(t \% 256);
```

May 02, 04 2:03	frmMain.cs	Page 127/186	May 02, 04 2:03	frmMain.cs	Page 128/186
	NewData[SizeIndex] = Byte1;			if(IsValidHex(Nibble))	
	SizeIndex++;			<pre>NewAppData += Nibble.ToString();</pre>	
	NewData[SizeIndex] = Byte2;			}	
	// Update the loading form			// Check the size of the new data	
	Loading.UpdateAndIncrement();			if((NewAppData.Length % 2) == 1)	
	Loading.UpdateAndIncrement();			NewAppData += "0";	
	this.update();			// Calculate the size field	
	// Write the new values to NewData			t = ((NewAppData.Length + 4)/2);	
	<pre>for(int x = 0; x &lt; NewAppData.Length; x+=2)</pre>			Byte2 = (byte)(t % 256);	
	$\{$ NewData[count] = SetButeValue(			t >>= 8; Bytel = (byte)(t % 256).	
	NewAppData[x], NewAppData[x+1]);			NewData[SizeIndex] = Byte1;	
	count++;			SizeIndex++;	
	Loading.UpdateAndIncrement():			NewData[SizeIndex] = Byte2;	
	this.Update();			// Update the loading form	
	}			Loading.UpdateAndIncrement();	
}	lse if(AppDataNumber == 3)			<pre>this.Update():</pre>	
{					
	AppDataNumber++;			// Write the new values to NewData	
	// Read out the interface data			<pre>for(int x = 0; x &lt; NewAppData.Length; x+=2) {</pre>	
	for (int x = 0; x < txtApplicationData4.Text.Le	ength; x++)		NewData[count] = SetByteValue(	
	{			<pre>NewAppData[x], NewAppData[x+1]);</pre>	
	if(IsValidHex(Nibble))			count++;	
	NewAppData += Nibble.ToString();			Loading.UpdateAndIncrement();	
	}			this.Update();	
	// Check the size of the new data		}	}	
	if((NewAppData.Length % 2) == 1)		e	else if(AppDataNumber == 5)	
	NewAppData += "0";		{	AppDataNumber++ ·	
	// Calculate the size field			nppbacanamber ()	
	t = ((NewAppData.Length + 4)/2);			// Read out the interface data	* 11
	by le2 = (by le)(l < 256); t >>= 8;			for (int $x = 0$ ; $x < txtApplicationData6.lext.$	Length; x++)
	Byte1 = (byte)(t % 256);			Nibble = txtApplicationData6.Text[x];	
	NewData[SizeIndex] = Bytel; SizeIndex++.			if(IsValidHex(Nibble)) NewAppData += Nibble ToString():	
	NewData[SizeIndex] = Byte2;			}	
	// The data that has done from				
	// Update the loading form			if((NewAppData.Length % 2) == 1)	
	Loading.UpdateAndIncrement();			NewAppData += "0";	
	this.Update();			// Calculate the size field	
	// Write the new values to NewData			t = ((NewAppData.Length + 4)/2);	
	<pre>for(int x = 0; x &lt; NewAppData.Length; x+=2)</pre>			Byte2 = (byte)(t % 256);	
	$\{$ NewData[count] = SetButeValue(			t >>= 8; Butel = (bute)(t % 256):	
	NewAppData[x], NewAppData[x+1]);			NewData[SizeIndex] = Byte1;	
	count++;			SizeIndex++;	
	Loading.UpdateAndIncrement();			NewData[SizeIndex] = Bytez;	
	this.Update();			// Update the loading form	
۱	}			Loading.UpdateAndIncrement();	
r e	else if(AppDataNumber == 4)			this.Update();	
{	App Det ellipse			// White the new velues to NewDet-	
	Appualanumber++;			for (int $x = 0$ ; $x < NewAppData.Length: x+=2$ )	
	// Read out the interface data			{	
	<pre>for(int x = 0; x &lt; txtApplicationData5.Text.Le </pre>	ength; x++)		NewData[count] = SetByteValue(	
	<pre>Nibble = txtApplicationData5.Text[x];</pre>			count++;	
## frmMain.cs

```
frmMain.cs
                                                                                                                         frmMain.cs
May 02, 04 2:03
                                                                   Page 129/186
                                                                                       May 02, 04 2:03
                                                                                                                                                          Page 130/186
                                                                                                        NewData[SizeIndex] = Byte2;
                   Loading.UpdateAndIncrement();
                   this.Update();
                                                                                                        // Update the loading form
                                                                                                        Loading.UpdateAndIncrement();
                                                                                                        Loading.UpdateAndIncrement();
               else if (AppDataNumber == 6)
                                                                                                        this.Update();
                 AppDataNumber++;
                                                                                                        // Write the new values to NewData
                                                                                                        for(int x = 0; x < NewAppData.Length; x+=2)</pre>
                 // Read out the interface data
                 for(int x = 0; x < txtApplicationData7.Text.Length; x++)</pre>
                                                                                                          NewData[count] = SetByteValue(
                                                                                                            NewAppData[x], NewAppData[x+1]);
                   Nibble = txtApplicationData7.Text[x];
                                                                                                          count++;
                   if(IsValidHex(Nibble))
                     NewAppData += Nibble.ToString();
                                                                                                          Loading.UpdateAndIncrement();
                                                                                                          this.Update();
                 // Check the size of the new data
                 if((NewAppData.Length % 2) == 1)
                                                                                                      else if (AppDataNumber == 8)
                   NewAppData += "0";
                                                                                                        AppDataNumber++;
                 // Calculate the size field
                 t = ((NewAppData.Length + 4)/2);
                                                                                                        // Read out the interface data
                                                                                                        for(int x = 0; x < txtApplicationData9.Text.Length; x++)</pre>
                 Byte2 = (byte)(t % 256);
                 t >>= 8;
                 Byte1 = (byte)(t % 256);
                                                                                                          Nibble = txtApplicationData9.Text[x];
                 NewData[SizeIndex] = Byte1;
                                                                                                          if(IsValidHex(Nibble))
                 SizeIndex++;
                                                                                                            NewAppData += Nibble.ToString();
                 NewData[SizeIndex] = Byte2;
                 // Update the loading form
                                                                                                        // Check the size of the new data
                 Loading.UpdateAndIncrement();
                                                                                                        if((NewAppData.Length % 2) == 1)
                 Loading.UpdateAndIncrement();
                                                                                                          NewAppData += "0";
                 this.Update();
                                                                                                        // Calculate the size field
                 // Write the new values to NewData
                                                                                                        t = ((NewAppData.Length + 4)/2);
                 for(int x = 0; x < NewAppData.Length; x+=2)</pre>
                                                                                                        Byte2 = (byte) (t % 256);
                                                                                                        t >>= 8;
                   NewData[count] = SetByteValue(
                                                                                                        Byte1 = (byte)(t % 256);
                     NewAppData[x], NewAppData[x+1]);
                                                                                                        NewData[SizeIndex] = Byte1;
                   count++;
                                                                                                        SizeIndex++;
                                                                                                        NewData[SizeIndex] = Byte2;
                   Loading.UpdateAndIncrement();
                   this.Update();
                                                                                                        // Update the loading form
                                                                                                        Loading.UpdateAndIncrement();
                                                                                                        Loading.UpdateAndIncrement();
               else if (AppDataNumber == 7)
                                                                                                        this.Update();
                 AppDataNumber++;
                                                                                                        // Write the new values to NewData
                                                                                                        for (int x = 0; x < NewAppData.Length; x+=2)
                 // Read out the interface data
                 for(int x = 0; x < txtApplicationData8.Text.Length; x++)</pre>
                                                                                                          NewData[count] = SetByteValue(
                                                                                                            NewAppData[x], NewAppData[x+1]);
                   Nibble = txtApplicationData8.Text[x];
                                                                                                          count++;
                   if(IsValidHex(Nibble))
                     NewAppData += Nibble.ToString();
                                                                                                          Loading.UpdateAndIncrement();
                                                                                                          this.Update();
                 // Check the size of the new data
                 if((NewAppData.Length % 2) == 1)
                                                                                                      else if (AppDataNumber == 9)
                   NewAppData += "0";
                                                                                                        AppDataNumber++;
                 // Calculate the size field
                 t = ((NewAppData.Length + 4)/2);
                                                                                                        // Read out the interface data
                 Byte2 = (byte)(t \% 256);
                                                                                                        for (int x = 0; x < txtApplicationData10.Text.Length; x++)
                 t >>= 8;
                 Byte1 = (byte)(t % 256);
                                                                                                          Nibble = txtApplicationData10.Text[x];
                 NewData[SizeIndex] = Byte1;
                                                                                                          if(IsValidHex(Nibble))
                 SizeIndex++;
                                                                                                            NewAppData += Nibble.ToString();
```

## frmMain.cs

```
frmMain.cs
                                                                                                                            frmMain.cs
May 02, 04 2:03
                                                                     Page 131/186
                                                                                         May 02, 04 2:03
                                                                                                                                                             Page 132/186
                                                                                                               " was found in the original stream. "+
                                                                                                                                    "Marker and data NOT written to new fi
                  // Check the size of the new data
                                                                                       le.";
                  if((NewAppData.Length % 2) == 1)
                                                                                                            txtError.Update();
                    NewAppData += "0";
                                                                                                            break;
                  // Calculate the size field
                  t = ((NewAppData.Length + 4)/2);
                                                                                                          case 'e': // marker fffe - Comments
                  Byte2 = (byte)(t \% 256);
                  t >>= 8;
                                                                                                            byte Bytel;
                  Byte1 = (byte)(t % 256);
                                                                                                            byte Byte2;
                  NewData[SizeIndex] = Byte1;
                                                                                                            int SizeIndex = count;
                  SizeIndex++;
                                                                                                            int t;
                                                                                                            string NewComments = "";
                  NewData[SizeIndex] = Byte2;
                                                                                                            char Nibble;
                  // Update the loading form
                  Loading.UpdateAndIncrement();
                                                                                                             // Check if loading canceled
                  Loading.UpdateAndIncrement();
                                                                                                            if (Loading.Canceled)
                  this.Update();
                                                                                                               Loading.Dispose();
                                                                                                              return false;
                  // Write the new values to NewData
                  for(int x = 0; x < NewAppData.Length; x+=2)</pre>
                    NewData[count] = SetByteValue(
                                                                                                             // Read out the interface data
                      NewAppData[x], NewAppData[x+1]);
                                                                                                            for(int x = 0; x < txtComments.Text.Length; x++)</pre>
                    count++;
                                                                                                              Nibble = txtComments.Text[x];
                    Loading.UpdateAndIncrement();
                                                                                                              NewComments += Nibble.ToString();
                    this.Update();
                                                                                                            // Calculate the new field size
                                                                                                            t = NewComments.Length + 2;
                else
                                                                                                            Byte2 = (byte)(t \% 256);
                  // Output an error
                                                                                                            t >>= 8;
                  txtError.Text +=
                                                                                                            Byte1 = (byte) (t % 256);
                    "\nError: Too Many Application Data frames!! " +
"\n\t-- Marker ff" + C.ToString() + D.ToString()+
                                                                                                            NewData[SizeIndex] = Byte1;
                                                                                                            SizeIndex++;
                    " was found in the original stream. "+
                                                                                                            NewData[SizeIndex] = Byte2;
                                         "Marker and data NOT written to new file
.";
                                                                                                             // Update the loading form
                  txtError.Update();
                                                                                                            Loading.UpdateAndIncrement();
                1
                                                                                                            Loading.UpdateAndIncrement();
                                                                                                            this.Update();
                break;
                                                                                                             // Write the new vales to NewData
              - }
                                                                                                             for (int x = 0; x < NewComments.Length; x++)
              case 'f': // marker fffX
                                                                                                              NewData[count] = (byte)NewComments[x];
                switch(D)
                                                                                                              count++;
                  case '0': goto case 'd';
                                                                                                              Loading.UpdateAndIncrement();
                  case '1': goto case 'd';
                                                                                                              this.Update();
                  case '2': goto case 'd';
                  case '3': goto case 'd';
                                                                                                            break;
                  case '4': goto case 'd';
                  case '5': goto case 'd';
                                                                                                          case 'f': // marker ffff -- Marker Not Defined
                  case '6': goto case 'd';
                  case '7': goto case 'd';
                                                                                                            txtError.Text +=
                  case '8': goto case 'd';
                                                                                                               "\nError: Marker NOT defined " +
                  case '9': goto case 'd';
                                                                                                               "\n\t-- Marker ffff was found in the original file "+
                  case 'a': goto case 'd';
                                                                                                                                    "stream.\nMarker and Data not written
                  case 'b': goto case 'd';
                                                                                        to the new file.";
                  case 'c': goto case 'd';
                                                                                                            txtError.Update();
                  case 'd':
                                                                                                            break;
                  { // marker fff0 to fffd: Reserved for JPEG extensions
                    txtError.Text +=
                                                                                                          default:
                      "\nError: Reserved ofr JPEG Extensions marker found !! "+
                      "\n\t-- Marker ff" + C.ToString() + D.ToString()+
                                                                                                            txtError.Text +=
```

```
frmMain.cs
                                                                                                                       frmMain.cs
May 02, 04 2:03
                                                                  Page 133/186
                                                                                     May 02, 04 2:03
                                                                                                                                                        Page 134/186
                     "\nError: Invalid File Marker Read!! " +
                                                                                          // End of: private void CreatedManipulatedPicture()
                     "\n\t-- Marker ffd" + D.ToString()+
                     " was found in the original file stream. " +
                     "Marker and Data not written to the new file.";
                                                                                        /// <summary>
                   txtError.Update();
                                                                                        /// Pre-conditions:
                                                                                                               None.
                                                                                        /// Post-conditions:
                   break;
                                                                                              All of the data for the new JPEG image being created is written to
                                                                                               the file name contained in the txtManipulatedFile TextBox field.
                                                                                        111
              } // End of: switch(D)
                                                                                            Description:
                                                                                        111
                                                                                        111
                                                                                               The purpose of this method is to create a new manipulated image
                                                                                        111
                                                                                               based upon all of the data currently loaded within the Manipulator.
              break;
                                                                                        111
                                                                                               To perform this functionality, this function should call the
                                                                                               CreateManipulatedPicture() method to create a file string to store
                                                                                        111
            default:
                                                                                        111
                                                                                               the new file data. Then, this function should call the WriteFile()
                                                                                        111
                                                                                               method to write all of this data to the new file. Then, to update
              txtError.Text +=
                                                                                        111
                                                                                               the Manipulated picture files, this function should call the
                 "\nError: Invalid File Marker Read!! " +
                                                                                        111
                                                                                               UpdateManipulatedPicture() method. Lastly, this method should do
                                                                                               some error checking to make sure this function executes properly.
                 "\n\t-- Marker ff" + C.ToString() + D.ToString()+
                                                                                        111
                " was found in the original file stream. " +
                                                                                               If an error is encountered, then the ShowWarning() method should
                                                                                        111
                 "Marker and Data not written to the new file.";
                                                                                        111
                                                                                               be called to display the error to the user and the txtError
               txtError.Update();
                                                                                        111
                                                                                              TextBox control should be updated with this error information.
              break;
                                                                                        /// </summary>
                                                                                        private void CreateISEImage()
           } // End of: switch(Top1)
                                                                                          if(!LoadingInterface)
         } // End of: if(Top1 == 'f' && Bottom1 == 'f')
                                                                                            if (CreateManipulatedPicture (ref NewData))
        else
                                                                                              if(ISE != null)
           if(ShowWarning(
             "\nYou have an invalid marker!"))
                                                                                                ISE.Dispose();
                                                                                                ISE = null;
             txtError.Text +=
                                                                                                ISEsmall.Dispose();
               "\nError: Invalid Marker Found!! " +
                                                                                                ISEsmall = null;
               "\n\t-- Marker ff" + C.ToString() + D.ToString() +
               " was found in the original file stream. " +
                                                                                              WriteFile(ref NewData);
               "Marker and Data not written to the new file.";
                                                                                              UpdateManipulatedPicture(this.txtManipulatedFile.Text.Trim());
             txtError.Update();
             ShowWarning(
               "\nYou have an invalid marker! Do you want to continue "+
                                                                                          else
                               "to write to file?");
            break;
                                                                                            ShowWarning(
           1
                                                                                              "The interface is STILL being loaded, you cannot create a " +
                                                                                              "new file until load has finished.",
                                                                                              "Cannot Create New File!");
       } // End of: while (A != 'f' && B != 'f' && C != 'd' && D != 'a')
    catch (Exception ex)
                                                                                        #endregion Methods to Convert from ACSII to Binary
      if(!ShowWarning(
         "Warning, an exception occured:\n\n" +
                                                                                        #endregion ISE Coded Functions
         "Exception Error:\n" +
        ex.Message + "\n\nWas throw by:\n" +
                                                                                        #region Created by Windows Form Designer
        ex.Source +
         "\n\nNot all write operations completed for this updated file,"+
                 " do you want to continue with the load operation?" +
                                                                                        // Variables created by the Visual Studio .NET Form Designer
         "\n(if you choose to continue you will have data loss)",
                                                                                        11
         "Load File Exception"))
                                                                                        private System.Windows.Forms.MainMenu menuFrmMain;
                                                                                        private System.Windows.Forms.MenuItem menuFile;
        Loading.Dispose();
        return false;
                                                                                        private System.Windows.Forms.PictureBox picOriginal;
                                                                                        private System.Windows.Forms.PictureBox picManipulated;
      ClearInterfaceData();
                                                                                        private System.Windows.Forms.PictureBox picOriginalSmall;
                                                                                        private System.Windows.Forms.PictureBox picManipulatedSmall;
    Loading.Dispose();
                                                                                        private System.Windows.Forms.MenuItem menuOpen;
    return true;
                                                                                        private System.Windows.Forms.MenuItem menuExit;
```

May 02, 04	2:03 frmMain.cs	Page 135/186	May 02, 04 2:03	frmMain.cs	Page 136/186
private	System.Windows.Forms.OpenFileDialog openFileDialog;		private System.Wind	ows.Forms.Button btnAddRandomHuffman2;	
			private System.Wind	ows.Forms.Button btnClearHuffman3;	
private	System.ComponentModel.IContainer components;		private System.Wind	ows.Forms.Button btnAddRandomHuffman3;	
nrivate	System Windows Forms ToolTin toolTins:		private System.Wind	ows Forms Button btnAddBandomHuffman1.	
privace	System. Windows. Points. Poor Pp coor Pps,		private System.Wind	ows.Forms.Button btnRestoreHuffman4;	
private	System.Windows.Forms.TabControl tabMain;		private System.Wind	ows.Forms.Button btnRestoreHuffman3;	
			private System.Wind	ows.Forms.Button btnRestoreHuffman2;	
private	System.Windows.Forms.TabPage tabConsol;		private System.Wind	ows.Forms.Button btnRestoreHuffman1;	
private	System.Windows.Forms.TabPage tabUriginal;		private System.Wind	ows.Forms.TextBox txtHuIImanOriginal4;	
private	System.Windows.Forms.SaveFileDialog_saveFileDialog1:		private System.Wind	ows.Forms.Label lblHuffmanOriginal4:	
private	System.Windows.Forms.OpenFileDialog openFileDialog1;		private System.Wind	ows.Forms.TextBox txtHuffman4;	
private	System.Windows.Forms.MenuItem menuOpenProject;		private System.Wind	ows.Forms.Label lblHuffmanMarker4;	
private	System.Windows.Forms.MenuItem menuSaveProject;		private System.Wind	ows.Forms.Label lblHuffman4;	
private	System.Windows.Forms.MenuItem menuItem1;		private System.Wind	ows.Forms.TextBox txtHuffmanOriginal2;	
private	System.Windows.Forms.Menuitem menuNewProject;		private System.Wind	ows.Forms.Label lblHuffmanOriginalMarker2;	
private	System Windows Forms MenuItem menuIdit:		private System Wind	ows Forms TextBox txtHuffman2.	
private	System.Windows.Forms.MenuItem_menuCopy:		private System.Wind	ows.Forms.Label lblHuffmanMarker2:	
private	System.Windows.Forms.MenuItem menuCut;		private System.Wind	ows.Forms.Label lblHuffman2;	
private	System.Windows.Forms.MenuItem menuPaste;		private System.Wind	ows.Forms.TextBox txtHuffmanOriginal3;	
private	System.Windows.Forms.MenuItem menuUpdate;		private System.Wind	ows.Forms.Label lblHuffmanOriginalMarker3;	
private	System.Windows.Forms.MenuItem menuView;		private System.Wind	ows.Forms.Label lblHuffmanOriginal3;	
private	System.Windows.Forms.Menultem menuStretchMode;		private System.Wind	ows.Forms.TextBox txtHuIIman3;	
private	System.Windows.Forms.Menuitem menulargeOriginal;		private System.Wind	ows.Forms.Label lblHuffman3.	
private	System.Windows.Forms.MenuItem_menuLargeManipulated:		private System.Wind	ows.Forms.TextBox_txtHuffmanOriginal1:	
private	System.Windows.Forms.MenuItem menuSmallManipulated;		private System.Wind	ows.Forms.Label lblHuffmanOriginalMarker1;	
private	System.Windows.Forms.MenuItem menuAll;		private System.Wind	ows.Forms.Label lblHuffmanOriginal1;	
private	System.Windows.Forms.TabPage tabProject;		private System.Wind	ows.Forms.TextBox txtHuffman1;	
private	System.Windows.Forms.Label lblNotes;		private System.Wind	ows.Forms.Label lblHuffmanMarker1;	
private	System.Windows.Forms.Button btnUpdatePicture;		private System.Wind	ows.Forms.Label lblHuffmanl;	
private	System Windows Forms Button btnLoadPicture;		private System.Wind	ows.Forms.Button_btnClearHuffman8.	
private	System.Windows.Forms.Label lblFilePath;		private System.Wind	ows.Forms.Button btnAddRandomHuffman8;	
private	System.Windows.Forms.TextBox txtProjectPath;		private System.Wind	ows.Forms.Button btnClearHuffman7;	
private	System.Windows.Forms.Button btnLoad;		private System.Wind	ows.Forms.Button btnAddRandomHuffman7;	
private	System.Windows.Forms.Button btnSave;		private System.Wind	ows.Forms.Button btnClearHuffman6;	
private	System.Windows.Forms.Button btnNew;		private System.Wind	ows.Forms.Button btnAddRandomHuffman6;	
private	System Windows Forms TabPage tabFile.		private System Wind	ows Forms Button btnAddBandomHuffman5.	
private	System.Windows.Forms.Label lblComments;		private System.Wind	ows.Forms.Button btnRestoreHuffman8;	
private	System.Windows.Forms.TextBox txtComments;		private System.Wind	ows.Forms.Button btnRestoreHuffman7;	
private	System.Windows.Forms.TextBox txtFileSize;		private System.Wind	ows.Forms.Button btnRestoreHuffman6;	
private	System.Windows.Forms.Label lblFileSize;		private System.Wind	ows.Forms.Button btnRestoreHuffman5;	
private	System.Windows.Forms.Label lblManipulatedFile;		private System.Wind	ows.Forms.TextBox txtHuffmanOriginal8;	
private	System.Windows.Forms.lextBox txtManipulatedFile;		private System.Wind	ows.Forms.Label lblHuffmanOriginalMarker8;	
private	System.Windows.Forms.TextBox txtOriginalFile:		private System.Wind	ows.Forms.TextBox_txtHuffman8:	
private	System.Windows.Forms.TabPage tabHeaders;		private System.Wind	ows.Forms.Label lblHuffmanMarker8;	
private	System.Windows.Forms.Label lblComponents;		private System.Wind	ows.Forms.Label lblHuffman8;	
private	System.Windows.Forms.Label lblNumberImageComponents;		private System.Wind	ows.Forms.TextBox txtHuffmanOriginal6;	
private	System.Windows.Forms.Label lblNumberHuffmanSamples;		private System.Wind	ows.Forms.Label lblHuffmanOriginalMarker6;	
private	System.Windows.Forms.Label lblNumberHullmanLines;		private System.Wind	ows.Forms.Label ibiHuffmanOriginal6;	
private	System.Windows.Forms.Label lblStartHuffmanSize.		private System.Wind	ows.Forms.lextBox txtHulimano;	
private	System.Windows.Forms.Label lblStartHuffman;		private System.Wind	ows.Forms.Label lblHuffman6;	
private	System.Windows.Forms.RichTextBox txtComponents;		private System.Wind	ows.Forms.TextBox txtHuffmanOriginal7;	
private	System.Windows.Forms.TextBox txtNumberImageComponents	;	private System.Wind	ows.Forms.Label lblHuffmanOriginalMarker7;	
private	System.Windows.Forms.TextBox txtNumberHuffmanSamples;		private System.Wind	ows.Forms.Label lblHuffmanOriginal7;	
private	System.Windows.Forms.TextBox txtNumberHuffmanLines;		private System.Wind	ows.Forms.TextBox txtHuffman7;	
private	System.Windows.Forms.TextBox txtPrecision;		private System.Wind	ows.Forms.Label lblHuffmanMarker/;	
private	System.Windows.Forms.TextBox_txtStartHuffman.		private System Wind	ows.Forms.TextBox_txtHuffmanOriginal5.	
private	System.Windows.Forms.TabPage tabHuffman1;		private System.Wind	ows.Forms.Label lblHuffmanOriginalMarker5:	
private	System.Windows.Forms.Button btnClearHuffman4;		private System.Wind	ows.Forms.Label lblHuffmanOriginal5;	
private	System.Windows.Forms.Button btnAddRandomHuffman4;		private System.Wind	ows.Forms.TextBox txtHuffman5;	
private	System.Windows.Forms.Button btnClearHuffman2;		private System.Wind	ows.Forms.Label lblHuffmanMarker5;	

May 02, 04	2:03 frmMain.cs	Page 137/186	May 02, 04 2:03	frmMain.cs	Page 138/186
private	System.Windows.Forms.Label lblHuffman5;		private System.Windo	ows.Forms.TextBox txtApplicationData3;	
private	System.Windows.Forms.TabPage tabQuantizer;		private System.Windo	ows.Forms.Label lblApplicationMarker3;	
private	System.Windows.Forms.Button btnClearQuantizer4;		private System.Windo	ows.Forms.Label lblApplicationData3;	
private	System.Windows.Forms.Button btnAddRandomQuantizer4;		private System.Windo	<pre>ws.Forms.TextBox txtApplicationData2;</pre>	
private	System.Windows.Forms.Button btnClearQuantizer3;		private System.Windo	ows.Forms.Label lblApplicationMarker2;	
private	System.Windows.Forms.Button btnAddRandomQuantizer3;		private System.Windo	ws.Forms.Label lblApplicationData2;	
private	System.Windows.Forms.Button btnClearQuantizer2;		private System.Windo	<pre>ows.Forms.TextBox txtApplicationData1;</pre>	
private	System.Windows.Forms.Button btnAddRandomQuantizer2;		private System.Windo	ows.Forms.Label lblApplicationMarker1;	
private	System.Windows.Forms.Button btnClearQuantizer1;		private System.Windo	ows.Forms.Label lblApplicationDatal;	
private	System.Windows.Forms.Button btnAddRandomQuantizer1;		private System.Windo	ows.Forms.TabPage tabMisc;	
private	System.Windows.Forms.Button btnRestoreQuantizer4;		private System.Windo	ows.Forms.Label iblExpandMarker;	
private	System.Windows.Forms.Button btnRestoreQuantizer3;		private System.windo	DWS.Forms.lextBox txtExpand;	
private	System.Windows.Forms.Button btnRestoreQuantizerz;		private System.windo	WS.FOINS.Label IDIExpand;	
private	System Windows Forms TextBox tytOuantizerOriginal4.		private System Windo	ws.Forms Label lblHierarchialMarker.	
private	System Windows Forms Label lblOuantizerOriginalMarkerA		private System Windo	ws.Forms Label lblHierarchial.	
private	System Windows Forms Label lblQuantizerOriginal4.	/	private System Windo	ws Forms TextBox txtRestartMod8.	
private	System, Windows, Forms, TextBox, txtOuantizer4:		private System.Windo	ws.Forms.Label lblRestartMod8:	
private	System.Windows.Forms.Label lblOuantizerMarker4:		private System.Windo	ows.Forms.TextBox_txtError:	
private	System.Windows.Forms.Label lblOuantizer4;		private System.Windo	ws.Forms.Label lblError;	
private	System.Windows.Forms.TextBox txtOuantizerOriginal2;		private System.Windo	ws.Forms.Label lblNumberLinesMarker;	
private	System.Windows.Forms.Label lblQuantizerOriginalMarker2	;	private System.Windo	ows.Forms.Label lblRestartMarker;	
private	System.Windows.Forms.Label lblQuantizerOriginal2;		private System.Windo	ows.Forms.TextBox txtNumberLines;	
private	System.Windows.Forms.TextBox txtQuantizer2;		private System.Windo	ows.Forms.Label lblNumberLines;	
private	System.Windows.Forms.Label lblQuantizerMarker2;		private System.Windo	ows.Forms.TextBox txtRestart;	
private	System.Windows.Forms.Label lblQuantizer2;		private System.Windo	ows.Forms.Label lblRestart;	
private	System.Windows.Forms.TextBox txtQuantizerOriginal3;		private System.Windo	<pre>ows.Forms.Label lblQuantizerTableNum1;</pre>	
private	System.Windows.Forms.Label lblQuantizerOriginalMarker3	;	private System.Windo	<pre>ows.Forms.Label txtQuantizerTableNum1;</pre>	
private	System.Windows.Forms.Label lblQuantizerOriginal3;		private System.Windo	<pre>ws.Forms.Label txtQuantizerTableNum2;</pre>	
private	System.Windows.Forms.TextBox txtQuantizer3;		private System.Windo	<pre>ows.Forms.Label lblQuantizerTableNum2;</pre>	
private	System.Windows.Forms.Label lblQuantizerMarker3;		private System.Windo	<pre>ows.Forms.Label txtQuantizerTableNum3;</pre>	
private	System.Windows.Forms.Label lblQuantizer3;		private System.Windo	<pre>ws.Forms.Label lblQuantizerTableNum3;</pre>	
private	System.Windows.Forms.TextBox txtQuantizerOriginal1;		private System.Windo	ows.Forms.Label txtQuantizerTableNum4;	
private	System.Windows.Forms.Label lblQuantizerOriginalMarkerl	;	private System.Windo	ows.Forms.Label lblQuantizerTableNum4;	
private	System.Windows.Forms.Label IblQuantizerOriginall;		private System.Windo	ows.Forms.TabControl tabSubConsole;	
private	System.Windows.Forms.TextBox txtQuantizer1;				
private	System.Windows.Forms.Label ibiQuantizerMarker1;		Handmarian Created k	Windows Form Designer	
private	System.Windows.Forms.Laber ibiQuantizeri;		#endregion created i	y windows form besigner	
private	System Windows Forms Label lblOriginalHeader.		#region Standard Wir	ndows Form Application Methods	
private	System, Windows, Forms, TextBox, txtOriginalHeader:		"region beanaara wii	aowo ronw appreación nechodo	
private	System.Windows.Forms.Label lblScanHeader:				
private	System.Windows.Forms.TextBox txtScanHeader;		/// <summarv></summarv>		
private	System.Windows.Forms.TextBox txtOriginalEncodedData;		/// Pre-conditions:	None.	
private	System.Windows.Forms.Label lblOriginalEncodedData;		/// Post-conditions:		
private	System.Windows.Forms.TextBox txtEncodedData;		/// The frmMain H	form of the application has been construc	ted.
private	System.Windows.Forms.Label lblEncodedData;		/// Parameters: No	one.	
private	System.Windows.Forms.TabPage tabApplicationData;		/// Return values:		
private	System.Windows.Forms.TextBox txtApplicationData10;		/// Form construct	ctor, no return type.	
private	System.Windows.Forms.Label lblApplicationMarker10;		/// Description:		
private	System.Windows.Forms.Label lblApplicationData10;		/// This is the o	constructor for the frmMain Form of the a	pplication.
private	System.Windows.Forms.TextBox txtApplicationData9;		/// This function	n will call the InitializeComponent() met	hod and the
private	System.Windows.Forms.Label lblApplicationMarker9;		/// ISEConstructo	or() to initialize the application.	
private	System.Windows.Forms.Label lblApplicationData9;		///		
private	System.Windows.Forms.TextBox txtApplicationData8;		public frmMain()		
private	System.Windows.Forms.Label lblApplicationMarker8;		{		
private	System.Windows.Forms.Label lbLApplicationData8;		InitializeComponer	nt();	
private	System.Windows.Forms.TextBox txtApplicationData/;		ISEConstructor();		
private	System.Windows.Forms.Label lblApplicationMarker/;		}		
private	System.windows.Forms.Label iDiApplicationData/;				
private	System.windows.rorms.labol lblbooligationMarker(		/// <0110000000		
private	System Windows Forms Label 1blApplicationData6.		/// Summary>	None	
private	System Windows Forms TextBox tytloplicationData5.		/// Post-conditions:	NOIIC .	
private	System Windows Forms Label 1612001104010000005.		/// All of the me	mory and resources used in the frmMain h	ave been
private	System Windows Forms Label lblapplicationData5.		/// ALL OL LINE INC	Smory and resources used in the riumdin h	ave been
private	System. Windows. Forms. TextBox txtApplicationData4.		/// Parameters·		
private	System.Windows.Forms.Label lblApplicationMarker4.		/// TRUE to relea	ase both managed and unmanaged resources	and FALSE to
private	System.Windows.Forms.Label lblApplicationData4;		/// release only	unmanaged resources.	
-	/				

May 02, 04 2:03	frmMain.cs	Page 139/186	May 02, 04 2:03	frmMain.cs	Page 140/186
<pre>/// Return values: /// Function return /// Description: /// This function i /// instance of the /// implementation /// that require laboration</pre>	s void. s called when the application is Form is destroyed. It is not re of this method is recommended for rege amounts of data to ensure the	when the current quired, but .NET objects at all memory	<pre>this.menuItem6 = net this.menuAbout = net this.tabMain = new 3 this.tabConsol = net this.tabSubConsol = this.tabProject = net this.lblNotes = new this.thurbdePicture</pre>	<pre>System.Windows.Forms.MenuItem() System.Windows.Forms.MenuItem() System.Windows.Forms.TabControl() System.Windows.Forms.TabPage(); new System.Windows.Forms.TabPage() System.Windows.Forms.Label(); e = new System Windows.Forms.Label();</pre>	<pre>crol(); con():</pre>
<pre>/// allocated for t /// destroyed. ///  protected override voi</pre>	d Dispose( bool disposing )	the Form is	this.btnSavePicture this.btnLoadPicture this.lblFilePath = r this.txtProjectPath	<pre>= new System.Windows.Forms.Button = new System.Windows.Forms.Button new System.Windows.Forms.Label(); = new System.Windows.Forms.TextBo new System.Windows.Forms.TextBo</pre>	<pre>&gt;</pre>
<pre>if( disposing ) {     if (components !=     { }</pre>	null)		<pre>this.bthLoad = new S this.bthSave = new S this.bthNew = new S this.txtNotes = new this.tabFile = new</pre>	<pre>system.Windows.Forms.Button(); system.Windows.Forms.Button(); stem.Windows.Forms.Button(); System.Windows.Forms.TextBox(); system.Windows.Forms.TabPage();</pre>	
<pre>components.Dispo } base.Dispose( dispos</pre>	<pre>se(); ing );</pre>		this.txtManipulated this.lblComments = r this.txtComments = r this.txtFileSize = r	"ile = new System.Windows.Forms.Te new System.Windows.Forms.Label(); new System.Windows.Forms.TextBox() new System.Windows.Forms.TextBox()	extBox(); ); );
} #region Windows Form D	esigner generated code		this.lblFileSize = r this.lblManipulated this.lblOriginalFile this.txtOriginalFile	<pre>wew System.Windows.Forms.Label(); Tile = new System.Windows.Forms.Labe = new System.Windows.Forms.Labe = new System.Windows.Forms.TextI</pre>	abel(); l(); Box();
<pre>/// <summary> /// Pre-conditions: /// Post-conditions:</summary></pre>	None.		this.tabHeaders = ne this.lblComponents = this.lblNumberImage( this.lblNumberHuffma	w System.Windows.Forms.TabPage(), new System.Windows.Forms.Label( components = new System.Windows.Forms angles = new System.Windows.Forms	; ); prms.Label(); cms.Label();
/// All of the vari /// Designer have b /// Parameters: /// Return values:	ables created by the Visual Studi een initialized. None.	) .NET Form	this.lblNumberHuffma this.lblPrecision = this.lblStartHuffman this.lblStartHuffman	<pre>inLines = new System.Windows.Forms new System.Windows.Forms.Label(), ISize = new System.Windows.Forms.labe new System.Windows.Forms.Labe</pre>	s.Label(); ; Label(); L();
/// Description: /// This function i	s void. s required to be called by the Fo	rmM-^Rs constructor.	this.txtNumberImage( this.txtNumberHuffma this.txtNumberHuffma this.txtNumberHuffma	<pre>cmew System.Windows.Form.Windows.Fo components = new System.Windows.Fo nSamples = new System.Windows.Form. nLines = new System.Windows.Form. Sour Surtem Windows Forms ToutBour</pre>	<pre>xLBOx(); prms.TextBox(); rms.TextBox(); s.TextBox();</pre>
<pre>/// it initializes /// designer at the ///  private void Initializ /</pre>	eComponent()	on.	this.txtStartHuffmar this.txtStartHuffmar this.tabHuffmanl = r this.tabHuffmanl = r	System.Windows.Forms. a = new System.Windows.Forms.TabPage((	(); TextBox(); Box(); ); con():
this.components = ne System.Resources.Res System.R this menuFrmMain = n	<pre>w System.ComponentModel.Container ourceManager resources = new .esources.ResourceManager(typeof(f tew System Windows Forms MainMenu(</pre>	(); rmMain));	this.btnAddRandomHu this.btnClearHuffmar this.btnAddRandomHu this.btnClearHuffmar	<pre>incw System.Windows.Forms ifman4 = new System.Windows.Forms.Butt ifman2 = new System.Windows.Forms.Butt ifman2 = new System Windows.Forms 3 = new System Windows.Forms</pre>	Button(); con(); Button(); con():
this.menuFile = new this.menuOpen = new this.menuUpdate = ne this.menuItem1 = new	System.Windows.Forms.MenuItem(); System.Windows.Forms.MenuItem(); w System.Windows.Forms.MenuItem(); System.Windows.Forms.MenuItem();	;	this.btnAddRandomHu this.btnClearHuffmar this.btnAddRandomHu this.btnPatter	<pre>ind</pre>	Button(); con(); .Button();
this.menuNewProject this.menuOpenProject this.menuSaveProject this menuItem3 = new	<pre>= new System.Windows.Forms.MenuIt = new System.Windows.Forms.MenuI = new System.Windows.Forms.MenuI </pre>	<pre>em(); tem(); tem();</pre>	this.btnRestoreHuffr this.btnRestoreHuffr this.btnRestoreHuffr this.txtHuffmanOrig	aan3 = new System.Windows.Forms.B aan2 = new System.Windows.Forms.Bu aan1 = new System.Windows.Forms.B nal4 = new System.Windows.Forms.	<pre>itton(); itton(); itton(); FextBox();</pre>
this.menuExit = new this.menuEdit = new this.menuCopy = new this.menuCut = new S	System.Windows.Forms.MenuItem(); System.Windows.Forms.MenuItem(); System.Windows.Forms.MenuItem(); vstem.Windows.Forms.MenuItem();		this.lblHuffmanOrig this.lblHuffmanOrig this.lblHuffmanOrig this.txtHuffman4 = r	nalMarker4 = new System.Windows.I nal4 = new System.Windows.Forms.J ew System.Windows.Forms.TextBox(' r4 - new System Windows Forms.Lal	Forms.Label(); Label();
this.menuPaste = new this.menuView = new this.menuStretchMode	System.Windows.Forms.MenuItem(); System.Windows.Forms.MenuItem(); = new System.Windows.Forms.MenuI = new System.Windows.Forms.MenuI	<pre>tem(); uItem();</pre>	this.lblHuffman4 = r this.txtHuffmanOrig this.lblHuffmanOrig this.lblHuffmanOrig	<pre>iew System.Windows.Forms.Label(); nal2 = new System.Windows.Forms.' nalMarker2 = new System.Windows.Forms.' nal2 - new System Windows Forms.'</pre>	<pre>Set(); Forms.Label(); Sabel();</pre>
this.menuLargeManipu this.menuSmallOrigin this.menuSmallManipu	<pre>al = new System.Windows.Forms.Men al = new System.Windows.Forms.Men lated = new System.Windows.Forms. ustem Windows Forms Monutary).</pre>	<pre>MenuItem(); uItem(); MenuItem();</pre>	this.txtHuffman2 = r this.lblHuffmanMarkethis.lblHuffmanOrig	<pre>wew System.Windows.Forms.TextBox() eve System.Windows.Forms.TextBox() eve System.Windows.Forms.Label(); nal3 = new System Windows.Forms()</pre>	<pre>Subst(); bel(); SevtBox().</pre>
this.menuItem2 = new this.menuItem2 = new this.menuItorial = this.menuManual = ne	<pre>System.Windows.Forms.MenuItem(); new System.Windows.Forms.MenuItem() w System.Windows.Forms.MenuItem()</pre>	(); ;	this.lblHuffmanOrig: this.lblHuffmanOrig: this.txtHuffman3 = r	<pre>.nalMarker3 = new System.Windows.Forms. .nalMarker3 = new System.Windows.Forms. .nal3 = new System.Windows.Forms.TextBox()</pre>	Forms.Label(); Label();

May 02, 04 2:03	frmMain.cs	Page 141/186	May 02, 04 2:03	frmMain.cs	Page 142/186
this.lblHuffmanMark	er3 = new System.Windows.Forms.Labe	1();	this.txtQuantizer4	= new System.Windows.Forms.TextBox()	;
this.lblHuffman3 =	new System.Windows.Forms.Label();		this.lblQuantizerMa	arker4 = new System.Windows.Forms.Lak	cel();
this.txtHuffmanOrig	inal1 = new System.Windows.Forms.Te	xtBox();	this.lblQuantizer4	= new System.Windows.Forms.Label();	
this.lblHuffmanOrig	inalMarker1 = new System.Windows.Fo	rms.Label():	this.txtOuantizerO	riginal2 = new System.Windows.Forms.7	rextBox():
this lblHuffmanOrig	inall = new System Windows Forms La	bel():	this lblouantizer0	riginalMarker2 = new System Windows F	forms Label() ·
thic tytHuffman1 -	now System Windows Forms ToxtBox():		this lblouantizor0	riginal2 = now System Windows Forms I	abol():
this lalluffmanMank	and new System. Windows. Forms. Textbox(),	1 ( ) .	this.totQualitizero	rou Custer Windows Forms Tout Day ()	Jaber(),
this lbltuffmanMark	eri = new System.windows.Forms.Labe	1();	this.txtQuantizerz	= new System.windows.forms.lextbox()	<i>i</i>
this.lblHuffmanl =	new System.Windows.Forms.Label();		this.lblQuantizerMa	arker2 = new System.Windows.Forms.Lar	)el();
this.tabHuffman2 =	new System.Windows.Forms.TabPage();		this.lblQuantizer2	<pre>= new System.Windows.Forms.Label();</pre>	
this.btnClearHuffma	n8 = new System.Windows.Forms.Butto	n();	this.txtQuantizerO	riginal3 = new System.Windows.Forms.7	l'extBox();
this.btnAddRandomHu	ffman8 = new System.Windows.Forms.B	utton();	this.lblQuantizerO	riginalMarker3 = new System.Windows.F	<pre>'orms.Label();</pre>
this.btnClearHuffma	n7 = new System.Windows.Forms.Butto	n();	this.lblQuantizerO	riginal3 = new System.Windows.Forms.I	_abel();
this.btnAddRandomHu	ffman7 = new System.Windows.Forms.B	utton():	this.txtOuantizer3	= new System.Windows.Forms.TextBox()	:
this htnClearHuffma	n6 = new System Windows Forms Butto	n().	this lblOuantizerMa	arker3 = new System Windows Forms Lak	) ) ) ·
this htnAddBandomHu	ffman6 - new System Windows Forms B	utton().	this lblouantizer3	- new System Windows Forms Label():	,01 () <b>,</b>
this btnCloarUuffma	n5 - new System Windows Forms Butto	n().	this twtOuantigorO	riginall - new System Windows Forms	Fort Por().
LIIIS DUNCLEAR HUIIMA	no = new System.windows.Forms.Butto	();	this.txtQuantizeron	iginali = new system.windows.forms.i	.extbox();
this.btnAddRandomHu	<pre>IIman5 = new System.Windows.Forms.B</pre>	utton();	this.lblQuantizer0	ciginalMarker1 = new System.Windows.F	orms.Label();
this.btnRestoreHuff	man8 = new System.Windows.Forms.But	ton();	this.lblQuantizer0	riginall = new System.Windows.Forms.I	Jabel();
this.btnRestoreHuff	man7 = new System.Windows.Forms.But	ton();	this.txtQuantizer1	<pre>= new System.Windows.Forms.TextBox()</pre>	/ <b>;</b>
this.btnRestoreHuff	man6 = new System.Windows.Forms.But	ton();	this.lblQuantizerMa	arker1 = new System.Windows.Forms.Lak	cel();
this.btnRestoreHuff	man5 = new System.Windows.Forms.But	ton();	this.lblQuantizer1	= new System.Windows.Forms.Label();	
this.txtHuffmanOrig	inal8 = new System.Windows.Forms.Te	xtBox():	this.tabEncodedData	a = new System.Windows.Forms.TabPage/	():
this lblHuffmanOrig	inalMarker8 - new System Windows Fo	rms Label() ·	this lblOriginalHe	ader - new System Windows Forms Labe	().
this lbluuffmanOrig	inal - new System Windows Forma La	hol()	this twtOriginalWe	ader - new System. Windows. Forms. Haber	-()
this totutinanorig	Inalo = new System.windows.roims.La	Der();	this lbloss Header	der = new System.windows.rolms.lexte	SOX();
this.txtHuriman8 =	new System.windows.forms.lextBox();		this.ibiScanHeader	= new System.windows.Forms.Label();	
this.lblHuffmanMark	er8 = new System.Windows.Forms.Labe	1();	this.txtScanHeader	= new System.Windows.Forms.TextBox()	;
this.lblHuffman8 =	new System.Windows.Forms.Label();		this.txtOriginalEnd	<pre>codedData = new System.Windows.Forms.</pre>	.TextBox();
this.txtHuffmanOrig	inal6 = new System.Windows.Forms.Te	xtBox();	this.lblOriginalEnd	codedData = new System.Windows.Forms.	.Label();
this.lblHuffmanOrig	inalMarker6 = new System.Windows.Fo	rms.Label();	this.txtEncodedData	a = new System.Windows.Forms.TextBox(	();
this.lblHuffmanOrig	inal6 = new System.Windows.Forms.La	bel():	this.lblEncodedData	a = new System.Windows.Forms.Label();	
this txtHuffman6 =	new System Windows Forms TextBox().	() /	this tabApplication	nData = new System Windows Forms TabE	Page() ·
thic lblWuffmanMark	or6 = now System Windows Forms Labo	1().	this tythoplication	nDatal0 - new System Windows Forms To	age(),
this lalluffman(	ero - new System.windows.roims.labe	±(),	this lblapplication	Manham10 - new System.windows.roims.re	Lobol(),
UNIS.IDIHUIIMano =	new System.windows.Forms.Laber();		UNIS. IDIAPPIICALIO	Markerio = new System.windows.forms.	Laber();
this.txtHuffmanOrig	inal/ = new System.Windows.Forms.Te	xtBox();	this.lblApplication	.Datal0 = new System.Windows.Forms.La	ibel();
this.lblHuffmanOrig	<pre>inalMarker/ = new System.Windows.Fo</pre>	rms.Label();	this.txtApplication	<pre>nData9 = new System.Windows.Forms.Tex</pre>	(tBox();
this.lblHuffmanOrig	inal7 = new System.Windows.Forms.La	bel();	this.lblApplication	nMarker9 = new System.Windows.Forms.I	_abel();
this.txtHuffman7 = :	new System.Windows.Forms.TextBox();		this.lblApplication	nData9 = new System.Windows.Forms.Lak	pel();
this.lblHuffmanMark	er7 = new System.Windows.Forms.Labe	1();	this.txtApplication	nData8 = new System.Windows.Forms.Tex	<tbox();< td=""></tbox();<>
this.lblHuffman7 =	new System.Windows.Forms.Label();		this.lblApplication	nMarker8 = new System.Windows.Forms.I	Label();
this.txtHuffmanOrig	inal5 = new System.Windows.Forms.Te	xtBox():	this, lblApplication	nData8 = new System.Windows.Forms.Lak	pel():
this lblHuffmanOrig	inalMarker5 = new System Windows Fo	rms Label() ·	this tytApplication	nData7 = new System Windows Forms Tex	xtBox().
this lbl#uffmanOrig	inaliarkers new System.Windows Forms La	bol():	this lblApplication	nMarkor7 = now System Windows Forms I	abol();
this tytuffman5	now Swatom Windows Forms ToutBoy().	DC1(),	this lblapplication	nData7 - new System. Windows. Forma Lak	
this littleffmanMark	new System.windows.roims.lextbox();	1 () .	this totApplication	IData/ = new System.windows.Forms.Lac	
this.lblHuffmanMark	ers = new System.windows.forms.Labe	1();	this.txtApplication	.Datab = new System.windows.Forms.lex	();
this.lblHuffman5 =	new System.Windows.Forms.Label();		this.lblApplication	<pre>Marker6 = new System.Windows.Forms.1</pre>	label();
this.tabQuantizer =	new System.Windows.Forms.TabPage()	;	this.lblApplication	nData6 = new System.Windows.Forms.Lak	pel();
this.txtQuantizerTa	<pre>bleNum4 = new System.Windows.Forms.</pre>	Label();	this.txtApplication	nData5 = new System.Windows.Forms.Tex	(tBox();
this.lblQuantizerTa	bleNum4 = new System.Windows.Forms.	Label();	this.lblApplication	nMarker5 = new System.Windows.Forms.I	Label();
this.txtOuantizerTa	bleNum3 = new System.Windows.Forms.	Label();	this.lblApplication	nData5 = new System.Windows.Forms.Lak	<pre>&gt;el();</pre>
this lblouantizerTa	bleNum3 = new System Windows Forms.	Label():	this txtApplication	nData4 = new System Windows Forms Tex	():
this tytOuantizerTa	hleNum2 - new System Windows Forms	Label();	this lblApplication	nMarkerA = new System Windows Forms I	abel():
this lblousntizerTa	bloNum2 - new System.Windows.Forms.		this lblApplication	nData4 - new System Windows Forma Lak	
this totQualitizeria.	bleNuml and Gustem Windows.Forms.		this totApplication	IData4 - new System.Windows.Forms.Lac	
this.txtQuantizeria	bleNumi = new System.windows.Forms.	Label();	this.txtApplication	.nData3 = new System.windows.Forms.lex	();
this.lblQuantizerTa	bleNuml = new System.Windows.Forms.	Label();	this.lblApplication	<pre>Marker3 = new System.Windows.Forms.1</pre>	label();
this.btnClearQuanti	<pre>zer4 = new System.Windows.Forms.But</pre>	ton();	this.lblApplication	nData3 = new System.Windows.Forms.Lak	pel();
this.btnAddRandomQu	antizer4 = new System.Windows.Forms	.Button();	this.txtApplication	nData2 = new System.Windows.Forms.Tex	(tBox();
this.btnClearQuanti	zer3 = new System.Windows.Forms.But	ton();	this.lblApplication	nMarker2 = new System.Windows.Forms.I	Label();
this.btnAddRandomOu	antizer3 = new System.Windows.Forms	Button():	this, lblApplication	nData2 = new System.Windows.Forms.Lak	pel():
this htnClearOuanti	zer2 = new System Windows Forms But	ton():	this tytApplication	nDatal = new System Windows Forms Tex	xtBox().
this htnAddBandomOu	antizer2 - new System Windows Forms	Button():	this lblApplication	nMarkerl - new System Windows Forms I	abel():
this htp://www.andongu	ancizerz - new System. Windows .FOIMS	+on().	this lblamlistic	nDatal - new Creter Minders Former 1-1	
this bunclearQuanti	Zeri - new System.Windows.Forms.But	Dutter () :	this.iDiAppilcation	ubacai = new System.windows.forms.Lar	)e_();
this.btnAddRandomQu	antizer1 = new System.Windows.Forms	.Button();	this.tabMisc = new	System.Windows.Forms.TabPage();	
this.btnRestoreQuan	tizer4 = new System.Windows.Forms.B	utton();	this.lblExpandMarke	<pre>er = new System.Windows.Forms.Label()</pre>	;
this.btnRestoreQuan	tizer3 = new System.Windows.Forms.B	utton();	this.txtExpand = ne	ew System.Windows.Forms.TextBox();	
this.btnRestoreOuan	tizer2 = new System.Windows.Forms.B	utton();	this.lblExpand = ne	ew System.Windows.Forms.Label();	
this.btnRestoreOuan	tizer1 = new System.Windows.Forms.B	utton();	this.txtHierarchia	1 = new System.Windows.Forms.TextBox/	();
this txtOuantizerOr	iginal4 = new System Windows Forms	TextBox():	this blHierarchia	Marker = new System Windows Forms La	abel():
this lblouantizoror	iginalMarkerA - new System Windows	Forms Label().	this lblHierarchia	1 - new System Windows Forms Labol()	
this lblousstister.	iginal - new System Windows.	$1 \text{ orms} \cdot \text{ label}()$	this total Child.	<pre>i = new System.windows.forms.idDel();</pre>	().
LIIIS.IDIVUdILLZETUT	INTIGOUS - NEW SYSCEM.WINDOWS.FORMS.	LANCT ( ) i	LIIIS.LXLKESLALTMOQ	J – HEW BYSLEH, WINDOWS.FULHS.IEXTBOX	

```
frmMain.cs
May 02, 04 2:03
                                                                  Page 143/186
     this.lblRestartMod8 = new System.Windows.Forms.Label();
    this.txtError = new System.Windows.Forms.TextBox();
    this.lblError = new System.Windows.Forms.Label();
    this.lblNumberLinesMarker = new System.Windows.Forms.Label();
    this.lblRestartMarker = new System.Windows.Forms.Label();
    this.txtNumberLines = new System.Windows.Forms.TextBox();
    this.lblNumberLines = new System.Windows.Forms.Label();
    this.txtRestart = new System.Windows.Forms.TextBox();
    this.lblRestart = new System.Windows.Forms.Label();
    this.picManipulatedSmall = new System.Windows.Forms.PictureBox();
    this.picOriginalSmall = new System.Windows.Forms.PictureBox();
    this.tabOriginal = new System.Windows.Forms.TabPage();
    this.picOriginal = new System.Windows.Forms.PictureBox();
    this.tabManipulated = new System.Windows.Forms.TabPage();
    this.picManipulated = new System.Windows.Forms.PictureBox();
    this.openFileDialog = new System.Windows.Forms.OpenFileDialog();
     this.toolTips = new System.Windows.Forms.ToolTip(this.components);
    this.saveFileDialog1 = new System.Windows.Forms.SaveFileDialog();
    this.openFileDialog1 = new System.Windows.Forms.OpenFileDialog();
    this.timerSplash = new System.Windows.Forms.Timer(this.components);
    this.tabMain.SuspendLayout();
    this.tabConsol.SuspendLayout();
    this.tabSubConsole.SuspendLayout();
    this.tabProject.SuspendLayout();
    this.tabFile.SuspendLayout();
    this.tabHeaders.SuspendLayout();
    this.tabHuffman1.SuspendLayout();
    this.tabHuffman2.SuspendLayout();
    this.tabQuantizer.SuspendLayout();
    this.tabEncodedData.SuspendLayout();
    this.tabApplicationData.SuspendLayout();
    this.tabMisc.SuspendLayout();
    this.tabOriginal.SuspendLayout();
    this.tabManipulated.SuspendLayout();
    this.SuspendLayout();
    11
    // menuFrmMain
    11
    this.menuFrmMain.MenuItems.AddRange(new
                System.Windows.Forms.MenuItem[] {
                                           this.menuFile,
                                           this.menuEdit,
                                           this.menuView.
                                           this.menuItem2});
     // menuFile
    11
    this.menuFile.Index = 0;
    this.menuFile.MenuItems.AddRange(new
                 System.Windows.Forms.MenuItem[] {
                                          this.menuOpen,
                                          this.menuUpdate,
                                          this.menuItem1,
                                          this.menuNewProject,
                                          this.menuOpenProject,
                                          this.menuSaveProject,
                                          this.menuItem3,
                                          this.menuExit});
    this.menuFile.Text = "&File";
    11
    // menuOpen
    11
    this.menuOpen.Index = 0;
    this.menuOpen.Text = "Loa&d Picture";
    this.menuOpen.Click += new System.EventHandler(this.menuOpen_Click);
    // menuUpdate
    11
     this.menuUpdate.Index = 1;
```

```
frmMain.cs
May 02, 04 2:03
                                                                   Page 144/186
    this.menuUpdate.Text = "&Update Picture";
    this.menuUpdate.Click += new
                System.EventHandler(this.menuUpdate_Click);
    // menuItem1
    11
    this.menuItem1.Index = 2;
    this.menuItem1.Text = "-";
    11
    // menuNewProject
    11
    this.menuNewProject.Index = 3;
    this.menuNewProject.Text = "&New Project";
    this.menuNewProject.Click += new
                System.EventHandler(this.menuNewProject_Click);
    // menuOpenProject
    11
    this.menuOpenProject.Index = 4;
    this.menuOpenProject.Text = "Open & Project";
    this.menuOpenProject.Click += new
                System.EventHandler(this.menuOpenProject_Click);
    // menuSaveProject
    11
    this.menuSaveProject.Index = 5;
    this.menuSaveProject.Text = "&Save Project";
    this.menuSaveProject.Click += new
                System.EventHandler(this.menuSaveProject_Click);
    // menuItem3
    11
    this.menuItem3.Index = 6;
    this.menuItem3.Text = "-";
    11
    // menuExit
    11
    this.menuExit.Index = 7;
    this.menuExit.Text = "E&xit";
    this.menuExit.Click += new System.EventHandler(this.menuExit_Click);
    11
    // menuEdit
    11
    this.menuEdit.Index = 1;
    this.menuEdit.MenuItems.AddRange(new
                System.Windows.Forms.MenuItem[] {
                                          this.menuCopy,
                                          this.menuCut,
                                          this.menuPaste});
    this.menuEdit.Text = "&Edit";
    11
    // menuCopy
    11
    this.menuCopy.Index = 0;
    this.menuCopy.Text = "&Copy";
    this.menuCopy.Click += new System.EventHandler(this.menuCopy_Click);
    // menuCut
    11
    this.menuCut.Index = 1;
    this.menuCut.Text = "Cut";
    this.menuCut.Click += new System.EventHandler(this.menuCut_Click);
    11
    // menuPaste
    11
    this.menuPaste.Index = 2;
    this.menuPaste.Text = "Paste";
    this.menuPaste.Click += new System.EventHandler(this.menuPaste_Click);
    11
```

```
frmMain.cs
                                                                                                                       frmMain.cs
May 02, 04 2:03
                                                                  Page 145/186
                                                                                     May 02, 04 2:03
                                                                                                                                                        Page 146/186
    // menuView
                                                                                          this.menuTutorial.Click += new
    11
                                                                                                      System.EventHandler(this.menuTutorial_Click);
    this.menuView.Index = 2;
    this.menuView.MenuItems.AddRange(new
                                                                                          // menuManual
                System.Windows.Forms.MenuItem[] {
                                                                                          11
                                          this.menuStretchMode});
                                                                                          this.menuManual.Index = 1;
    this.menuView.Text = "&View";
                                                                                          this.menuManual.Text = "Manual";
                                                                                          this.menuManual.Click += new
    11
    // menuStretchMode
                                                                                                      System.EventHandler(this.menuManual_Click);
    11
                                                                                          11
    this.menuStretchMode.Index = 0;
                                                                                          // menuItem6
    this.menuStretchMode.MenuItems.AddRange(new
                                                                                          11
                 System.Windows.Forms.MenuItem[] {
                                                                                          this.menuItem6.Index = 2;
                                                                                          this.menuItem6.Text = "-";
                                             this.menuLargeOriginal,
                                             this.menuLargeManipulated,
                                                                                          11
                                             this.menuSmallOriginal,
                                                                                          // menuAbout
                                             this.menuSmallManipulated,
                                                                                          11
                                             this.menuAll});
                                                                                          this.menuAbout.Index = 3;
    this.menuStretchMode.Text = "S&tretch Mode";
                                                                                          this.menuAbout.Text = "About";
    11
                                                                                          this.menuAbout.Click += new System.EventHandler(this.menuAbout_Click);
    // menuLargeOriginal
                                                                                          11
    11
                                                                                          // tabMain
                                                                                          11
    this.menuLargeOriginal.Index = 0;
    this.menuLargeOriginal.Text = "Large Original";
                                                                                          this.tabMain.Controls.Add(this.tabConsol);
    this.menuLargeOriginal.Click += new
                                                                                          this.tabMain.Controls.Add(this.tabOriginal);
                 System.EventHandler(this.menuLargeOriginal_Click);
                                                                                          this.tabMain.Controls.Add(this.tabManipulated);
                                                                                          this.tabMain.Dock = System.Windows.Forms.DockStyle.Fill;
    // menuLargeManipulated
                                                                                          this.tabMain.Location = new System.Drawing.Point(0, 0);
    11
                                                                                          this.tabMain.Name = "tabMain";
    this.menuLargeManipulated.Index = 1;
                                                                                          this.tabMain.SelectedIndex = 0;
    this.menuLargeManipulated.Text = "Large Manipulated";
                                                                                          this.tabMain.Size = new System.Drawing.Size(904, 653);
    this.menuLargeManipulated.Click += new
                                                                                          this.tabMain.TabIndex = 0;
                System.EventHandler(this.menuLargeManipulated_Click);
                                                                                          11
                                                                                          // tabConsol
    11
    // menuSmallOriginal
                                                                                          11
    11
                                                                                          this.tabConsol.Controls.Add(this.tabSubConsole);
    this.menuSmallOriginal.Index = 2;
                                                                                          this.tabConsol.Controls.Add(this.picManipulatedSmall);
    this.menuSmallOriginal.Text = "Small Original";
                                                                                          this.tabConsol.Controls.Add(this.picOriginalSmall);
    this.menuSmallOriginal.Click += new
                                                                                          this.tabConsol.Location = new System.Drawing.Point(4, 22);
                System.EventHandler(this.menuSmallOriginal_Click);
                                                                                          this.tabConsol.Name = "tabConsol";
                                                                                          this.tabConsol.Size = new System.Drawing.Size(896, 627);
    // menuSmallManipulated
                                                                                          this.tabConsol.TabIndex = 0;
    11
                                                                                          this.tabConsol.Text = "Console";
    this.menuSmallManipulated.Index = 3;
                                                                                          11
                                                                                          // tabSubConsole
    this.menuSmallManipulated.Text = "Small Manipulated";
    this.menuSmallManipulated.Click += new
                                                                                          11
                System.EventHandler(this.menuSmallManipulated_Click);
                                                                                          this.tabSubConsole.Controls.Add(this.tabProject);
    11
                                                                                          this.tabSubConsole.Controls.Add(this.tabFile);
    // menuAll
                                                                                          this.tabSubConsole.Controls.Add(this.tabHeaders);
                                                                                          this.tabSubConsole.Controls.Add(this.tabHuffman1);
    11
    this.menuAll.Index = 4;
                                                                                          this.tabSubConsole.Controls.Add(this.tabHuffman2);
    this.menuAll.Text = "A&LL Pictures";
                                                                                          this.tabSubConsole.Controls.Add(this.tabQuantizer);
    this.menuAll.Click += new System.EventHandler(this.menuAll_Click);
                                                                                          this.tabSubConsole.Controls.Add(this.tabEncodedData);
                                                                                          this.tabSubConsole.Controls.Add(this.tabApplicationData);
    11
    // menuItem2
                                                                                          this.tabSubConsole.Controls.Add(this.tabMisc);
    11
                                                                                          this.tabSubConsole.Dock = System.Windows.Forms.DockStyle.Bottom;
    this.menuItem2.Index = 3;
                                                                                          this.tabSubConsole.ItemSize = new System.Drawing.Size(45, 18);
                                                                                          this.tabSubConsole.Location = new System.Drawing.Point(0, 355);
    this.menuItem2.MenuItems.AddRange(new
                 System.Windows.Forms.MenuItem[] {
                                                                                          this.tabSubConsole.Name = "tabSubConsole";
                                           this.menuTutorial,
                                                                                          this.tabSubConsole.SelectedIndex = 0;
                                           this.menuManual,
                                                                                          this.tabSubConsole.Size = new System.Drawing.Size(896, 272);
                                           this.menuItem6,
                                                                                          this.tabSubConsole.TabIndex = 2;
                                           this.menuAbout});
                                                                                          11
    this.menuItem2.Text = "&Help";
                                                                                          // tabProject
    11
    // menuTutorial
                                                                                          this.tabProject.Controls.Add(this.lblNotes);
                                                                                          this.tabProject.Controls.Add(this.btnUpdatePicture);
    11
    this.menuTutorial.Index = 0;
                                                                                          this.tabProject.Controls.Add(this.btnSavePicture);
    this.menuTutorial.Text = "Tutorial";
                                                                                          this.tabProject.Controls.Add(this.btnLoadPicture);
```

```
frmMain.cs
                                                                                                                        frmMain.cs
May 02, 04 2:03
                                                                   Page 147/186
                                                                                      May 02, 04 2:03
                                                                                                                                                          Page 148/186
     this.tabProject.Controls.Add(this.lblFilePath);
                                                                                           this.btnLoad.Name = "btnLoad";
    this.tabProject.Controls.Add(this.txtProjectPath);
                                                                                           this.btnLoad.Size = new System.Drawing.Size(88, 24);
    this.tabProject.Controls.Add(this.btnLoad);
                                                                                           this.btnLoad.TabIndex = 3;
    this.tabProject.Controls.Add(this.btnSave);
                                                                                           this.btnLoad.Text = "Open Project";
    this.tabProject.Controls.Add(this.btnNew);
                                                                                           this.btnLoad.Click += new System.EventHandler(this.btnLoad_Click);
    this.tabProject.Controls.Add(this.txtNotes);
                                                                                           11
    this.tabProject.Location = new System.Drawing.Point(4, 22);
                                                                                           // btnSave
    this.tabProject.Name = "tabProject";
                                                                                           11
    this.tabProject.Size = new System.Drawing.Size(888, 246);
                                                                                           this.btnSave.Location = new System.Drawing.Point(776, 80);
                                                                                           this.btnSave.Name = "btnSave";
this.btnSave.Size = new System.Drawing.Size(88, 24);
    this.tabProject.TabIndex = 10;
    this.tabProject.Text = "Project";
                                                                                           this.btnSave.TabIndex = 2;
    11
    // lblNotes
                                                                                           this.btnSave.Text = "Save Project";
                                                                                           this.btnSave.Click += new System.EventHandler(this.btnSave_Click);
    11
    this.lblNotes.Location = new System.Drawing.Point(16, 40);
                                                                                           11
    this.lblNotes.Name = "lblNotes";
this.lblNotes.Size = new System.Drawing.Size(80, 16);
                                                                                           // btnNew
                                                                                           11
    this.lblNotes.TabIndex = 9;
                                                                                           this.btnNew.Location = new System.Drawing.Point(776, 16);
    this.lblNotes.Text = "Project Notes:";
                                                                                           this.btnNew.Name = "btnNew";
                                                                                           this.btnNew.Size = new System.Drawing.Size(88, 24);
    11
    // btnUpdatePicture
                                                                                           this.btnNew.TabIndex = 1;
    11
                                                                                           this.btnNew.Text = "New Project";
                                                                                           this.btnNew.Click += new System.EventHandler(this.btnNew_Click);
    this.btnUpdatePicture.Location = new System.Drawing.Point(776, 208);
    this.btnUpdatePicture.Name = "btnUpdatePicture";
                                                                                           11
    this.btnUpdatePicture.Size = new System.Drawing.Size(88, 24);
                                                                                           // txtNotes
    this.btnUpdatePicture.TabIndex = 8;
                                                                                           11
    this.btnUpdatePicture.Text = "Update Picture";
                                                                                           this.txtNotes.AcceptsTab = true;
    this.btnUpdatePicture.Click += new
                                                                                           this.txtNotes.Location = new System.Drawing.Point(112, 40);
                 System.EventHandler(this.btnUpdatePicture_Click);
                                                                                           this.txtNotes.Multiline = true;
                                                                                           this.txtNotes.Name = "txtNotes";
    11
    // btnSavePicture
                                                                                           this.txtNotes.ScrollBars = System.Windows.Forms.ScrollBars.Vertical;
                                                                                           this.txtNotes.Size = new System.Drawing.Size(640, 192);
    11
                                                                                           this.txtNotes.TabIndex = 0;
    this.btnSavePicture.Location = new System.Drawing.Point(776, 160);
    this.btnSavePicture.Name = "btnSavePicture";
                                                                                           this.txtNotes.Text = "";
    this.btnSavePicture.Size = new System.Drawing.Size(88, 24);
                                                                                           this.toolTips.SetToolTip(this.txtNotes,
                                                                                                  "These are the SEP (Selective Encryption Project) notes.");
    this.btnSavePicture.TabIndex = 7;
    this.btnSavePicture.Text = "Save Picture";
                                                                                           11
                                                                                           // tabFile
    11
    // btnLoadPicture
                                                                                           11
                                                                                           this.tabFile.Controls.Add(this.txtManipulatedFile);
    this.btnLoadPicture.Location = new System.Drawing.Point(776, 128);
                                                                                           this.tabFile.Controls.Add(this.lblComments);
    this.btnLoadPicture.Name = "btnLoadPicture";
                                                                                           this.tabFile.Controls.Add(this.txtComments);
    this.btnLoadPicture.Size = new System.Drawing.Size(88, 24);
                                                                                           this.tabFile.Controls.Add(this.txtFileSize);
    this.btnLoadPicture.TabIndex = 6;
                                                                                           this.tabFile.Controls.Add(this.lblFileSize);
    this.btnLoadPicture.Text = "Load Picture";
                                                                                           this.tabFile.Controls.Add(this.lblManipulatedFile);
    this.btnLoadPicture.Click += new
                                                                                           this.tabFile.Controls.Add(this.lblOriginalFile);
                 System.EventHandler(this.btnLoadPicture_Click);
                                                                                           this.tabFile.Controls.Add(this.txtOriginalFile);
    11
                                                                                           this.tabFile.Location = new System.Drawing.Point(4, 22);
    // lblFilePath
                                                                                           this.tabFile.Name = "tabFile";
                                                                                           this.tabFile.Size = new System.Drawing.Size(888, 246);
    11
    this.lblFilePath.Location = new System.Drawing.Point(16, 8);
                                                                                           this.tabFile.TabIndex = 5;
    this.lblFilePath.Name = "lblFilePath";
                                                                                           this.tabFile.Text = "File Information";
    this.lblFilePath.Size = new System.Drawing.Size(96, 16);
                                                                                           11
    this.lblFilePath.TabIndex = 5;
                                                                                           // txtManipulatedFile
    this.lblFilePath.Text = "Project File Path:";
                                                                                           11
    11
                                                                                           this.txtManipulatedFile.Location = new System.Drawing.Point(128, 48);
    // txtProjectPath
                                                                                           this.txtManipulatedFile.Name = "txtManipulatedFile";
                                                                                           this.txtManipulatedFile.Size = new System.Drawing.Size(752, 20);
    11
    this.txtProjectPath.Location = new System.Drawing.Point(112, 8);
                                                                                           this.txtManipulatedFile.TabIndex = 0;
    this.txtProjectPath.Name = "txtProjectPath";
                                                                                           this.txtManipulatedFile.Text = "";
    this.txtProjectPath.Size = new System.Drawing.Size(640, 20);
                                                                                           this.toolTips.SetToolTip(this.txtManipulatedFile,
    this.txtProjectPath.TabIndex = 4;
                                                                                                "This is the Manipulated file.");
    this.txtProjectPath.Text = "";
                                                                                           this.txtManipulatedFile.TextChanged += new
    this.toolTips.SetToolTip(this.txtProjectPath,
                                                                                                       System.EventHandler(this.txtManipulatedFile_TextChanged);
         "Path to the SEP (Selective Encryption Project) name and path.");
                                                                                           1
                                                                                           // lblComments
    // btnLoad
                                                                                           11
    11
                                                                                           this.lblComments.Location = new System.Drawing.Point(8, 113);
    this.btnLoad.Location = new System.Drawing.Point(776, 48);
                                                                                           this.lblComments.Name = "lblComments";
```

```
frmMain.cs
                                                                                                                       frmMain.cs
May 02, 04 2:03
                                                                  Page 149/186
                                                                                     May 02, 04 2:03
                                                                                                                                                        Page 150/186
     this.lblComments.Size = new System.Drawing.Size(112, 39);
                                                                                          this.tabHeaders.Controls.Add(this.lblNumberHuffmanLines);
                                                                                          this.tabHeaders.Controls.Add(this.lblPrecision);
    this.lblComments.TabIndex = 9;
    this.lblComments.Text = "File Comments:
                                                                                          this.tabHeaders.Controls.Add(this.lblStartHuffmanSize);
                                                    (Not Saved)";
                                                                                          this.tabHeaders.Controls.Add(this.lblStartHuffman);
    11
    // txtComments
                                                                                          this.tabHeaders.Controls.Add(this.txtComponents);
                                                                                          this.tabHeaders.Controls.Add(this.txtNumberImageComponents);
    11
    this.txtComments.Location = new System.Drawing.Point(128, 113);
                                                                                          this.tabHeaders.Controls.Add(this.txtNumberHuffmanSamples);
    this.txtComments.Multiline = true;
                                                                                          this.tabHeaders.Controls.Add(this.txtNumberHuffmanLines);
    this.txtComments.Name = "txtComments";
                                                                                          this.tabHeaders.Controls.Add(this.txtPrecision);
    this.txtComments.ScrollBars =
                                                                                          this.tabHeaders.Controls.Add(this.txtStartHuffmanSize);
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                          this.tabHeaders.Controls.Add(this.txtStartHuffman);
    this.txtComments.Size = new System.Drawing.Size(752, 71);
                                                                                          this.tabHeaders.Location = new System.Drawing.Point(4, 22);
    this.txtComments.TabIndex = 8;
                                                                                          this.tabHeaders.Name = "tabHeaders";
    this.txtComments.Text = "";
                                                                                          this.tabHeaders.Size = new System.Drawing.Size(888, 246);
    this.toolTips.SetToolTip(this.txtComments,
                                                                                          this.tabHeaders.TabIndex = 11;
           "These are the comments contain within the original file.");
                                                                                          this.tabHeaders.Text = "Headers";
                                                                                          11
    // txtFileSize
                                                                                          // lblComponents
    11
                                                                                          11
    this.txtFileSize.Location = new System.Drawing.Point(128, 80);
                                                                                          this.lblComponents.Location = new System.Drawing.Point(168, 48);
    this.txtFileSize.Name = "txtFileSize";
                                                                                          this.lblComponents.Name = "lblComponents";
    this.txtFileSize.Size = new System.Drawing.Size(128, 20);
                                                                                          this.lblComponents.Size = new System.Drawing.Size(184, 16);
    this.txtFileSize.TabIndex = 6;
                                                                                          this.lblComponents.TabIndex = 27;
    this.txtFileSize.TabStop = false;
                                                                                          this.lblComponents.Text = "Components:";
    this.txtFileSize.Text = "0";
                                                                                          11
    this.toolTips.SetToolTip(this.txtFileSize,
                                                                                          // lblNumberImageComponents
           "This is the size of the original file.");
                                                                                          11
                                                                                          this.lblNumberImageComponents.Location = new
    // lblFileSize
                                                                                                      System.Drawing.Point(168, 16);
                                                                                          this.lblNumberImageComponents.Name = "lblNumberImageComponents";
    11
    this.lblFileSize.Location = new System.Drawing.Point(8, 80);
                                                                                          this.lblNumberImageComponents.Size = new
    this.lblFileSize.Name = "lblFileSize";
                                                                                                      System.Drawing.Size(120, 16);
    this.lblFileSize.Size = new System.Drawing.Size(96, 16);
                                                                                          this.lblNumberImageComponents.TabIndex = 26;
    this.lblFileSize.TabIndex = 7;
                                                                                          this.lblNumberImageComponents.Text = "Number Components:";
    this.lblFileSize.Text = "File Size:";
                                                                                          11
                                                                                          // lblNumberHuffmanSamples
    11
    // lblManipulatedFile
                                                                                          11
                                                                                          this.lblNumberHuffmanSamples.Location = new
    11
    this.lblManipulatedFile.Location = new System.Drawing.Point(8, 48);
                                                                                                      System.Drawing.Point(8, 176);
    this.lblManipulatedFile.Name = "lblManipulatedFile";
                                                                                          this.lblNumberHuffmanSamples.Name = "lblNumberHuffmanSamples";
    this.lblManipulatedFile.Size = new System.Drawing.Size(128, 16);
                                                                                          this.lblNumberHuffmanSamples.Size = new System.Drawing.Size(56, 16);
    this.lblManipulatedFile.TabIndex = 3;
                                                                                          this.lblNumberHuffmanSamples.TabIndex = 25;
    this.lblManipulatedFile.Text = "Manipulated File Name:";
                                                                                          this.lblNumberHuffmanSamples.Text = "Width:";
                                                                                          this.toolTips.SetToolTip(this.lblNumberHuffmanSamples,
    11
    // lblOriginalFile
                                                                                                "The number of samples per line in the Huffman.");
    11
                                                                                          1
    this.lblOriginalFile.Location = new System.Drawing.Point(8, 16);
                                                                                          // lblNumberHuffmanLines
    this.lblOriginalFile.Name = "lblOriginalFile";
                                                                                          11
    this.lblOriginalFile.Size = new System.Drawing.Size(104, 16);
                                                                                          this.lblNumberHuffmanLines.Location = new System.Drawing.Point(8, 136);
    this.lblOriginalFile.TabIndex = 1;
                                                                                          this.lblNumberHuffmanLines.Name = "lblNumberHuffmanLines";
    this.lblOriginalFile.Text = "Original File Name:";
                                                                                          this.lblNumberHuffmanLines.Size = new System.Drawing.Size(56, 16);
                                                                                          this.lblNumberHuffmanLines.TabIndex = 2\hat{4};
    11
    // txtOriginalFile
                                                                                          this.lblNumberHuffmanLines.Text = "Height:";
                                                                                          this.toolTips.SetToolTip(this.lblNumberHuffmanLines,
    11
    this.txtOriginalFile.Enabled = false;
                                                                                                "Number of lines in the source");
    this.txtOriginalFile.Location = new System.Drawing.Point(128, 16);
    this.txtOriginalFile.Name = "txtOriginalFile";
                                                                                          // lblPrecision
    this.txtOriginalFile.Size = new System.Drawing.Size(752, 20);
                                                                                          11
    this.txtOriginalFile.TabIndex = 0;
                                                                                          this.lblPrecision.Location = new System.Drawing.Point(8, 96);
    this.txtOriginalFile.TabStop = false;
                                                                                          this.lblPrecision.Name = "lblPrecision";
    this.txtOriginalFile.Text = "";
                                                                                          this.lblPrecision.Size = new System.Drawing.Size(56, 16);
    this.toolTips.SetToolTip(this.txtOriginalFile,
                                                                                          this.lblPrecision.TabIndex = 23;
           "This is the original file name.");
                                                                                          this.lblPrecision.Text = "Precision:";
                                                                                          this.toolTips.SetToolTip(this.lblPrecision,
    // tabHeaders
                                                                                                "Precision in the Huffman");
    11
    this.tabHeaders.Controls.Add(this.lblComponents);
                                                                                          // lblStartHuffmanSize
    this.tabHeaders.Controls.Add(this.lblNumberImageComponents);
                                                                                          11
    this.tabHeaders.Controls.Add(this.lblNumberHuffmanSamples);
                                                                                          this.lblStartHuffmanSize.Location = new System.Drawing.Point(8, 56);
```

```
frmMain.cs
                                   frmMain.cs
May 02, 04 2:03
                                                                      Page 151/186
                                                                                           May 02, 04 2:03
                                                                                                                                                                  Page 152/186
     this.lblStartHuffmanSize.Name = "lblStartHuffmanSize";
     this.lblStartHuffmanSize.Size = new System.Drawing.Size(56, 16);
                                                                                                this.txtStartHuffmanSize.Location = new System.Drawing.Point(80, 56);
     this.lblStartHuffmanSize.TabIndex = 2\overline{2};
                                                                                                this.txtStartHuffmanSize.MaxLength = 32;
     this.lblStartHuffmanSize.Text = "Size:";
                                                                                                this.txtStartHuffmanSize.Name = "txtStartHuffmanSize";
this.txtStartHuffmanSize.Size = new System.Drawing.Size(56, 20);
     this.toolTips.SetToolTip(this.lblStartHuffmanSize,
            "Size of the Huffman header size.");
                                                                                                this.txtStartHuffmanSize.TabIndex = 15;
                                                                                                this.txtStartHuffmanSize.Text = "";
     11
     // lblStartHuffman
                                                                                                11
                                                                                                // txtStartHuffman
     11
    this.lblStartHuffman.Location = new System.Drawing.Point(8, 16);
this.lblStartHuffman.Name = "lblStartHuffman";
this.lblStartHuffman.Size = new System.Drawing.Size(56, 16);
                                                                                                11
                                                                                                this.txtStartHuffman.Location = new System.Drawing.Point(80, 16);
                                                                                                this.txtStartHuffman.MaxLength = 32;
     this.lblStartHuffman.TabIndex = 21;
                                                                                                this.txtStartHuffman.Name = "txtStartHuffman";
     this.lblStartHuffman.Text = "Marker:";
                                                                                                this.txtStartHuffman.Size = new System.Drawing.Size(56, 20);
     this.toolTips.SetToolTip(this.lblStartHuffman,
                                                                                                this.txtStartHuffman.TabIndex = 14;
            "Value of the Huffman marker.");
                                                                                                this.txtStartHuffman.Text = "";
                                                                                                11
     // txtComponents
                                                                                                // tabHuffman1
     11
                                                                                                11
                                                                                                this.tabHuffman1.Controls.Add(this.btnClearHuffman4);
     this.txtComponents.AcceptsTab = true;
     this.txtComponents.Location = new System.Drawing.Point(168, 64);
                                                                                                this.tabHuffman1.Controls.Add(this.btnAddRandomHuffman4);
    this.txtComponents.MaxLength = 1024;
this.txtComponents.Name = "txtComponents";
this.txtComponents.ScrollBars =
                                                                                                this.tabHuffman1.Controls.Add(this.btnClearHuffman2);
                                                                                                this.tabHuffman1.Controls.Add(this.btnAddRandomHuffman2);
                                                                                                this.tabHuffman1.Controls.Add(this.btnClearHuffman3);
                  System.Windows.Forms.RichTextBoxScrollBars.Vertical;
                                                                                                this.tabHuffman1.Controls.Add(this.btnAddRandomHuffman3);
                                                                                                this.tabHuffman1.Controls.Add(this.btnClearHuffman1);
     this.txtComponents.Size = new System.Drawing.Size(208, 152);
     this.txtComponents.TabIndex = 20;
                                                                                                this.tabHuffman1.Controls.Add (this.btnAddRandomHuffman1);
     this.txtComponents.Text = "";
                                                                                                this.tabHuffman1.Controls.Add(this.btnRestoreHuffman4);
                                                                                                this.tabHuffman1.Controls.Add(this.btnRestoreHuffman3);
     11
                                                                                                this.tabHuffman1.Controls.Add(this.btnRestoreHuffman2);
     // txtNumberImageComponents
     11
                                                                                                this.tabHuffman1.Controls.Add(this.btnRestoreHuffman1);
     this.txtNumberImageComponents.Location = new
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffmanOriginal4);
                  System.Drawing.Point(296, 16);
                                                                                                this.tabHuffman1.Controls.Add (this.lblHuffmanOriginalMarker4);
    this.txtNumberImageComponents.MaxLength = 32;
this.txtNumberImageComponents.Name = "txtNumberImageComponents";
this.txtNumberImageComponents.Size = new System.Drawing.Size(56, 20);
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanOriginal4);
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffman4);
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanMarker4);
     this.txtNumberImageComponents.TabIndex = 19;
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffman4);
     this.txtNumberImageComponents.Text = "";
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffmanOriginal2);
                                                                                                this.tabHuffman1.Controls.Add (this.lblHuffmanOriginalMarker2);
     //
     // txtNumberHuffmanSamples
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanOriginal2);
     11
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffman2);
     this.txtNumberHuffmanSamples.Location = new
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanMarker2);
                 System.Drawing.Point(80, 176);
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffman2);
     this.txtNumberHuffmanSamples.MaxLength = 32;
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffmanOriginal3);
     this.txtNumberHuffmanSamples.Name = "txtNumberHuffmanSamples";
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanOriginalMarker3);
     this.txtNumberHuffmanSamples.Size = new System.Drawing.Size(56, 20);
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanOriginal3);
     this.txtNumberHuffmanSamples.TabIndex = 18;
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffman3);
     this.txtNumberHuffmanSamples.Text = "";
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanMarker3);
     11
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffman3);
     // txtNumberHuffmanLines
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffmanOriginal1);
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanOriginalMarker1);
     11
     this.txtNumberHuffmanLines.Location = new
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanOriginal1);
                  System.Drawing.Point(80, 136);
                                                                                                this.tabHuffman1.Controls.Add(this.txtHuffman1);
     this.txtNumberHuffmanLines.MaxLength = 32;
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffmanMarker1);
     this.txtNumberHuffmanLines.Name = "txtNumberHuffmanLines";
                                                                                                this.tabHuffman1.Controls.Add(this.lblHuffman1);
     this.txtNumberHuffmanLines.Size = new System.Drawing.Size(56, 20);
                                                                                                this.tabHuffman1.Location = new System.Drawing.Point(4, 22);
     this.txtNumberHuffmanLines.TabIndex = 1\overline{7};
                                                                                                this.tabHuffman1.Name = "tabHuffman1";
                                                                                                this.tabHuffman1.Size = new System.Drawing.Size(888, 246);
     this.txtNumberHuffmanLines.Text = "";
     11
                                                                                                this.tabHuffman1.TabIndex = 0;
     // txtPrecision
                                                                                                this.tabHuffman1.Text = "Huffman Tables 1";
                                                                                                11
     this.txtPrecision.Location = new System.Drawing.Point(80, 96);
                                                                                                // btnClearHuffman4
     this.txtPrecision.MaxLength = 2048;
     this.txtPrecision.Name = "txtPrecision";
                                                                                                this.btnClearHuffman4.Font = new
     this.txtPrecision.Size = new System.Drawing.Size(56, 20);
                                                                                                             System.Drawing.Font(
     this.txtPrecision.TabIndex = 16;
                                                                                                        "Microsoft Sans Serif", 7F, System.Drawing.FontStyle.Regular,
     this.txtPrecision.Text = "";
                                                                                                                System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                                this.btnClearHuffman4.Location = new System.Drawing.Point(448, 152);
     11
     // txtStartHuffmanSize
                                                                                                this.btnClearHuffman4.Name = "btnClearHuffman4";
```

```
frmMain.cs
                                                                                                                          frmMain.cs
May 02, 04 2:03
                                                                    Page 153/186
                                                                                        May 02, 04 2:03
                                                                                                                                                            Page 154/186
     this.btnClearHuffman4.Size = new System.Drawing.Size(40, 16);
                                                                                             this.btnAddRandomHuffman3.Location = new System.Drawing.Point(56,
                                                                                                                                                                 152);
     this.btnClearHuffman4.TabIndex = 63;
                                                                                             this.btnAddRandomHuffman3.Name = "btnAddRandomHuffman3";
     this.btnClearHuffman4.Text = "Clear";
                                                                                             this.btnAddRandomHuffman3.Size = new System.Drawing.Size(48, 16);
     this.btnClearHuffman4.Click += new
                                                                                            this.btnAddRandomHuffman3.TabIndex = 58;
                 System.EventHandler(this.btnClearHuffman4_Click);
                                                                                            this.btnAddRandomHuffman3.Text = "Random";
                                                                                            this.btnAddRandomHuffman3.Click += new
     11
     // btnAddRandomHuffman4
                                                                                                         System.EventHandler(this.btnAddRandomHuffman3 Click);
     11
     this.btnAddRandomHuffman4.Font = new
                                                                                             // btnClearHuffman1
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                             11
                    System.Drawing.FontStyle.Regular,
                                                                                            this.btnClearHuffman1.Font = new
                    System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                                         System.Drawing.Font("Microsoft Sans Serif", 7F,
     this.btnAddRandomHuffman4.Location = new
                                                                                                             System.Drawing.FontStyle.Regular,
                 System.Drawing.Point(496, 152);
                                                                                                             System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnAddRandomHuffman4.Name = "btnAddRandomHuffman4";
                                                                                            this.btnClearHuffman1.Location = new System.Drawing.Point(8, 32);
                                                                                            this.btnClearHuffman1.Name = "btnClearHuffman1";
this.btnClearHuffman1.Size = new System.Drawing.Size(40, 16);
     this.btnAddRandomHuffman4.Size = new System.Drawing.Size(48, 16);
     this.btnAddRandomHuffman4.TabIndex = 62;
    this.btnAddRandomHuffman4.Text = "Random";
                                                                                            this.btnClearHuffman1.TabIndex = 57;
     this.btnAddRandomHuffman4.Click += new
                                                                                            this.btnClearHuffman1.Text = "Clear";
                 System.EventHandler(this.btnAddRandomHuffman4_Click);
                                                                                             this.btnClearHuffman1.Click += new
                                                                                                         System.EventHandler(this.btnClearHuffman1_Click);
     //
     // btnClearHuffman2
                                                                                             11
                                                                                             // btnAddRandomHuffman1
     11
     this.btnClearHuffman2.Font = new
                                                                                             11
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                            this.btnAddRandomHuffman1.Font = new
                     System.Drawing.FontStyle.Regular,
                                                                                                         System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                                             System.Drawing.FontStyle.Regular,
     this.btnClearHuffman2.Location = new System.Drawing.Point(448, 32);
                                                                                                             System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnClearHuffman2.Name = "btnClearHuffman2";
this.btnClearHuffman2.Size = new System.Drawing.Size(40, 16);
                                                                                             this.btnAddRandomHuffman1.Location = new System.Drawing.Point(56, 32);
                                                                                            this.btnAddRandomHuffmanl.Name = "btnAddRandomHuffmanl";
this.btnAddRandomHuffmanl.Size = new System.Drawing.Size(48, 16);
     this.btnClearHuffman2.TabIndex = 61;
     this.btnClearHuffman2.Text = "Clear";
                                                                                             this.btnAddRandomHuffman1.TabIndex = 56;
     this.btnClearHuffman2.Click += new
                                                                                            this.btnAddRandomHuffman1.Text = "Random";
                 System.EventHandler(this.btnClearHuffman2_Click);
                                                                                            this.btnAddRandomHuffman1.Click += new
     11
                                                                                                         System.EventHandler(this.btnAddRandomHuffman1 Click);
     // btnAddRandomHuffman2
                                                                                             // btnRestoreHuffman4
     11
     this.btnAddRandomHuffman2.Font = new
                                                                                             11
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                            this.btnRestoreHuffman4.Font = new
                                                                                                         System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                                             System.Drawing.FontStyle.Regular,
     this.btnAddRandomHuffman2.Location = new System.Drawing.Point(496, 32);
                                                                                                             System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnAddRandomHuffman2.Name = "btnAddRandomHuffman2";
                                                                                             this.btnRestoreHuffman4.Location = new System.Drawing.Point(496, 208);
     this.btnAddRandomHuffman2.Size = new System.Drawing.Size(48, 16);
                                                                                            this.btnRestoreHuffman4.Name = "btnRestoreHuffman4";
     this.btnAddRandomHuffman2.TabIndex = 60;
                                                                                            this.btnRestoreHuffman4.Size = new System.Drawing.Size(48, 16);
     this.btnAddRandomHuffman2.Text = "Random";
                                                                                            this.btnRestoreHuffman4.TabIndex = 5\overline{5};
    this.btnAddRandomHuffman2.Click += new
                                                                                            this.btnRestoreHuffman4.Text = "Restore";
                 System.EventHandler(this.btnAddRandomHuffman2_Click);
                                                                                            this.btnRestoreHuffman4.Click += new
     11
                                                                                                         System.EventHandler(this.btnRestoreHuffman4_Click);
     // btnClearHuffman3
                                                                                            // btnRestoreHuffman3
     11
     this.btnClearHuffman3.Font = new
                                                                                             11
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                            this.btnRestoreHuffman3.Font = new
                     System.Drawing.FontStyle.Regular,
                                                                                                         System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                                             System.Drawing.FontStyle.Regular,
     this.btnClearHuffman3.Location = new System.Drawing.Point(8, 152);
                                                                                                             System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnClearHuffman3.Name = "btnClearHuffman3";
                                                                                             this.btnRestoreHuffman3.Location = new System.Drawing.Point(56, 208);
     this.btnClearHuffman3.Size = new System.Drawing.Size(40, 16);
                                                                                             this.btnRestoreHuffman3.Name = "btnRestoreHuffman3";
    this.btnClearHuffman3.TabIndex = 59;
                                                                                            this.btnRestoreHuffman3.Size = new System.Drawing.Size(48, 16);
     this.btnClearHuffman3.Text = "Clear";
                                                                                            this.btnRestoreHuffman3.TabIndex = 54;
     this.btnClearHuffman3.Click += new
                                                                                            this.btnRestoreHuffman3.Text = "Restore";
                 System.EventHandler(this.btnClearHuffman3_Click);
                                                                                             this.btnRestoreHuffman3.Click += new
                                                                                                         System.EventHandler(this.btnRestoreHuffman3_Click);
    // btnAddRandomHuffman3
     11
                                                                                            // btnRestoreHuffman2
     this.btnAddRandomHuffman3.Font = new
                                                                                             11
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                             this.btnRestoreHuffman2.Font = new
                     System.Drawing.FontStyle.Regular,
                                                                                                         System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                                             System.Drawing.FontStyle.Regular,
```

```
frmMain.cs
May 02, 04 2:03
                                                                    Page 155/186
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnRestoreHuffman2.Location = new System.Drawing.Point(496, 88);
    this.btnRestoreHuffman2.Name = "btnRestoreHuffman2";
this.btnRestoreHuffman2.Size = new System.Drawing.Size(48, 16);
     this.btnRestoreHuffman2.TabIndex = 53;
     this.btnRestoreHuffman2.Text = "Restore";
     this.btnRestoreHuffman2.Click += new
                 System.EventHandler(this.btnRestoreHuffman2_Click);
     11
     // btnRestoreHuffman1
     11
     this.btnRestoreHuffman1.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnRestoreHuffman1.Location = new System.Drawing.Point(56, 88);
    this.btnRestoreHuffmanl.Name = "btnRestoreHuffmanl";
this.btnRestoreHuffmanl.Size = new System.Drawing.Size(48, 16);
     this.btnRestoreHuffman1.TabIndex = 52;
     this.btnRestoreHuffman1.Text = "Restore";
     this.btnRestoreHuffman1.Click += new
                 System.EventHandler(this.btnRestoreHuffman1_Click);
     // txtHuffmanOriginal4
     11
     this.txtHuffmanOriginal4.AutoSize = false;
     this.txtHuffmanOriginal4.Enabled = false;
     this.txtHuffmanOriginal4.Location = new System.Drawing.Point(552, 184);
     this.txtHuffmanOriginal4.Multiline = true;
     this.txtHuffmanOriginal4.Name = "txtHuffmanOriginal4";
     this.txtHuffmanOriginal4.ScrollBars =
                 System.Windows.Forms.ScrollBars.Horizontal;
     this.txtHuffmanOriginal4.Size = new System.Drawing.Size(328, 48);
     this.txtHuffmanOriginal4.TabIndex = 26;
     this.txtHuffmanOriginal4.TabStop = false;
    this.txtHuffmanOriginal4.Text = "";
    11
     // lblHuffmanOriginalMarker4
     11
     this.lblHuffmanOriginalMarker4.BackColor =
                 System.Drawing.SystemColors.Window;
     this.lblHuffmanOriginalMarker4.Enabled = false;
     this.lblHuffmanOriginalMarker4.Location = new
                 System.Drawing.Point(512, 184);
     this.lblHuffmanOriginalMarker4.Name = "lblHuffmanOriginalMarker4";
     this.lblHuffmanOriginalMarker4.Size = new System.Drawing.Size(32, 16);
    this.lblHuffmanOriginalMarker4.TabIndex = 25;
    this.lblHuffmanOriginalMarker4.TextAlign =
                 System.Drawing.ContentAlignment.TopCenter;
     // lblHuffmanOriginal4
     this.lblHuffmanOriginal4.Location = new System.Drawing.Point(456, 184);
    this.lblHuffmanOriginal4.Name = "lblHuffmanOriginal4";
    this.lblHuffmanOriginal4.Size = new System.Drawing.Size(64, 16);
     this.lblHuffmanOriginal4.TabIndex = 24;
    this.lblHuffmanOriginal4.Text = "Original 4:";
    11
    // txtHuffman4
     11
     this.txtHuffman4.AutoSize = false;
     this.txtHuffman4.Location = new System.Drawing.Point(552, 128);
     this.txtHuffman4.Multiline = true;
     this.txtHuffman4.Name = "txtHuffman4";
     this.txtHuffman4.ScrollBars =
                 System.Windows.Forms.ScrollBars.Horizontal;
     this.txtHuffman4.Size = new System.Drawing.Size(328, 48);
     this.txtHuffman4.TabIndex = 4;
     this.txtHuffman4.Text = "";
```

```
frmMain.cs
May 02, 04 2:03
                                                                       Page 156/186
     this.txtHuffman4.GotFocus += new
                  System.EventHandler(this.txtHuffman4_GotFocus);
     // lblHuffmanMarker4
     11
    this.lblHuffmanMarker4.BackColor = System.Drawing.SystemColors.Window;
     this.lblHuffmanMarker4.Enabled = false;
     this.lblHuffmanMarker4.Location = new System.Drawing.Point(512, 128);
    this.lblHuffmanMarker4.Name = "lblHuffmanMarker4";
this.lblHuffmanMarker4.Size = new System.Drawing.Size(32, 16);
     this.lblHuffmanMarker4.TabIndex = 2\dot{2};
     this.lblHuffmanMarker4.TextAlign =
                  System.Drawing.ContentAlignment.TopCenter;
     // lblHuffman4
     11
    this.lblHuffman4.Location = new System.Drawing.Point(456, 128);
this.lblHuffman4.Name = "lblHuffman4";
this.lblHuffman4.Size = new System.Drawing.Size(64, 16);
     this.lblHuffman4.TabIndex = 21;
     this.lblHuffman4.Text = "Huffman 4:";
     11
     // txtHuffmanOriginal2
     11
    this.txtHuffmanOriginal2.AutoSize = false;
     this.txtHuffmanOriginal2.Enabled = false;
     this.txtHuffmanOriginal2.Location = new System.Drawing.Point(552, 64);
     this.txtHuffmanOriginal2.Multiline = true;
    this.txtHuffmanOriginal2.Name = "txtHuffmanOriginal2";
this.txtHuffmanOriginal2.ScrollBars =
                  System.Windows.Forms.ScrollBars.Horizontal;
     this.txtHuffmanOriginal2.Size = new System.Drawing.Size(328, 48);
     this.txtHuffmanOriginal2.TabIndex = 20;
     this.txtHuffmanOriginal2.TabStop = false;
     this.txtHuffmanOriginal2.Text = "";
     11
     // lblHuffmanOriginalMarker2
     11
     this.lblHuffmanOriginalMarker2.BackColor =
                  System.Drawing.SystemColors.Window;
     this.lblHuffmanOriginalMarker2.Enabled = false;
     this.lblHuffmanOriginalMarker2.Location = new
                  System.Drawing.Point(512, 64);
     this.lblHuffmanOriginalMarker2.Name = "lblHuffmanOriginalMarker2";
     this.lblHuffmanOriginalMarker2.Size = new System.Drawing.Size(32, 16);
     this.lblHuffmanOriginalMarker2.TabIndex = 19;
     this.lblHuffmanOriginalMarker2.TextAlign =
                  System.Drawing.ContentAlignment.TopCenter;
     // lblHuffmanOriginal2
     11
     this.lblHuffmanOriginal2.Location = new System.Drawing.Point(456, 64);
    this.lblHuffmanOriginal2.Name = "lblHuffmanOriginal2";
     this.lblHuffmanOriginal2.Size = new System.Drawing.Size(64, 16);
     this.lblHuffmanOriginal2.TabIndex = 18;
     this.lblHuffmanOriginal2.Text = "Original 2:";
     // txtHuffman2
     11
    this.txtHuffman2.AutoSize = false;
     this.txtHuffman2.Location = new System.Drawing.Point(552, 8);
     this.txtHuffman2.Multiline = true;
     this.txtHuffman2.Name = "txtHuffman2";
     this.txtHuffman2.ScrollBars =
                  System.Windows.Forms.ScrollBars.Horizontal;
     this.txtHuffman2.Size = new System.Drawing.Size(328, 48);
    this.txtHuffman2.TabIndex = 1;
     this.txtHuffman2.Text = "";
     this.txtHuffman2.GotFocus += new
```

May 02, 04 2:03	frmMain.cs	Page 157/186	May 02, 04 2:03	frmMain.cs	Page 158/186
System	a.EventHandler(this.txtHuffman2_Got	Focus);	11		
//			// lblHuffmanMarker3	3	
// IDIHUIIMAnMarke	erz		// this.lblHuffmanMarke	er3.BackColor = System.Drawing.Sy	stemColors Window:
this.lblHuffmanMar	<pre>cker2.BackColor = System.Drawing.Sy</pre>	stemColors.Window;	this.lblHuffmanMarke	er3.Enabled = false;	Scencorors.window,
this.lblHuffmanMar	<pre>:ker2.Enabled = false;</pre>		this.lblHuffmanMarke	er3.Location = new System.Drawing	.Point(72, 128);
this.lblHuffmanMar	<pre>cker2.Location = new System.Drawing</pre>	.Point(512, 8);	this.lblHuffmanMarke	er3.Name = "lblHuffmanMarker3";	- (22 16)
this lblHuffmanMar	<pre>cker2.Name = "IDIHUIIManMarker2"; cker2 Size = new System Drawing Siz</pre>	e(32 16).	this lblHuffmanMarke	er3.Size = new System.Drawing.Size er3 TabIndey = 10.	e(32, 16);
this.lblHuffmanMar	<pre>cker2.TabIndex = 16;</pre>	(32, 10),	this.lblHuffmanMarke	er3.TextAlign =	
this.lblHuffmanMar	<pre>cker2.TextAlign =</pre>		System.I	Drawing.ContentAlignment.TopCente	r;
System	.Drawing.ContentAlignment.TopCente	r;	11		
// // ]b]]]utffmon2			// lblHuffman3		
// IDIHUIIManz			// this lblHuffman3 Loc	cation = new System Drawing Point	(16 128) •
this.lblHuffman2.L	ocation = new System.Drawing.Point	(456, 8);	this.lblHuffman3.Nam	<pre>me = "lblHuffman3";</pre>	(10) 120))
this.lblHuffman2.N	<pre>Jame = "lblHuffman2";</pre>		this.lblHuffman3.Siz	ze = new System.Drawing.Size(64,	16);
this.lblHuffman2.S	Size = new System.Drawing.Size(64,	16);	this.lblHuffman3.Tak	oIndex = 9;	
this.lblHuffman2.T	ablndex = 15;		this.lblHuffman3.Tex	<pre>xt = "Huffman 3:";</pre>	
//	ext - Hullman 2: ;		// txtHuffmanOrigina	a]]	
// txtHuffmanOrigi	nal3		//		
//			this.txtHuffmanOrigi	inal1.AutoSize = false;	
this.txtHuffmanOri	ginal3.AutoSize = false;		this.txtHuffmanOrigi	inal1.Enabled = false;	D 1 1 (110 CA)
this txtHuffmanOri	ginal3.Enabled = Talse; ginal3 Location - new System Drawi	ng Point (112 184).	this txtHuffmanOrigi	inall.Location = new System.Drawi	ng.Point(112, 64);
this.txtHuffmanOri	ginal3.Multiline = true:	IIG.FOIIIC(IIZ, 104),	this.txtHuffmanOrigi	inall.Name = "txtHuffmanOriginall	".
this.txtHuffmanOri	.ginal3.Name = "txtHuffmanOriginal3	";	this.txtHuffmanOrigi	inall.ScrollBars =	,
this.txtHuffmanOri	ginal3.ScrollBars =		System.V	Nindows.Forms.ScrollBars.Horizont	al;
System	.Windows.Forms.ScrollBars.Horizont	al;	this.txtHuffmanOrigi	inall.Size = new System.Drawing.S	ize(328, 48);
this txtHuffmanOri	.ginal3.Size = new System.Drawing.S ginal3 TabIndex = 14.	1Ze(328, 48);	this tytHuffmanOrigi	inall.labindex = 8; inall TabStop = false:	
this.txtHuffmanOri	ginal3.TabStop = false;		this.txtHuffmanOrigi	inal1.Text = "";	
this.txtHuffmanOri	ginal3.Text = "";		//	· · · · · ·	
11			// lblHuffmanOrigina	alMarker1	
// lblHuffmanOrigi	.nalMarker3		//	inelMerker1 DeckCeler	
// this.lblHuffmanOri	ginalMarker3.BackColor =		System. [	Drawing SystemColors Window:	
System	.Drawing.SystemColors.Window;		this.lblHuffmanOrigi	inalMarker1.Enabled = false;	
this.lblHuffmanOri	ginalMarker3.Enabled = false;		this.lblHuffmanOrigi	inalMarker1.Location = new	
this.lblHuffmanOri	_ginalMarker3.Location = new		System.I	Drawing.Point(72, 64);	ainelMenken1".
this lblHuffmanOri	dinalMarker3 Name = "lblHuffmanOri	ginalMarker3".	this lblHuffmanOrigi	inalMarker1 Size = new System Dra	wing Size(32 16).
this.lblHuffmanOri	.ginalMarker3.Size = new System.Dra	wing.Size(32, 16);	this.lblHuffmanOrigi	<pre>inalMarker1.TabIndex = 7;</pre>	
this.lblHuffmanOri	ginalMarker3.TabIndex = 13;	-	this.lblHuffmanOrigi	inalMarker1.TextAlign =	
this.lblHuffmanOri	ginalMarker3.TextAlign =		System.I	Drawing.ContentAlignment.TopCente	r;
// System	.Drawing.ContentAlignment.TopCente	r;	// // lblHuffmanOrigina	<b>a</b> 11	
// lblHuffmanOrigi	.nal3		//	<u></u>	
//			this.lblHuffmanOrigi	inall.Location = new System.Drawi	ng.Point(16, 64);
this.lblHuffmanOri	ginal3.Location = new System.Drawi	ng.Point(16, 184);	this.lblHuffmanOrigi	<pre>inal1.Name = "lblHuffmanOriginal1</pre>	";
this.lblHuffmanOri	ginal3.Name = "lblHuffmanOriginal3	";	this.lblHuffmanOrigi	inall.Size = new System.Drawing.S	ize(64, 16);
this lblHuffmanOri	ginal3.TabIndex = 12:	120(04, 10);	this.lblHuffmanOrigi	inall.Text = "Original 1:":	
this.lblHuffmanOri	.ginal3.Text = "Original 3:";		//		
//			// txtHuffman1		
// txtHuffman3			//		
// this tytHuffman3 A	utosizo - falco:		this txtHuffmanl.Aut	cosize = Ialse;	(112 8).
this.txtHuffman3.L	ocation = new System.Drawing.Point	(112, 128):	this.txtHuffman1.Mul	ltiline = true:	(112, 0),
this.txtHuffman3.M	<pre>/ultiline = true;</pre>	(,, ,	this.txtHuffman1.Nam	<pre>me = "txtHuffman1";</pre>	
this.txtHuffman3.N	<pre>Jame = "txtHuffman3";</pre>		this.txtHuffman1.Scr	rollBars =	
this.txtHuffman3.S	ScrollBars =	<u>_</u> ].	System.V	Nindows.Forms.ScrollBars.Horizont	al;
System this tytHuffman3 S	i.windows.Forms.ScrollBars.Horizont	a⊥; 48)•	this tytHuffmanl.Slz	<pre>2e = new System.Drawing.Size(328, bindex = 0.</pre>	40);
this.txtHuffman3.T	TabIndex = 3;	10/ /	this.txtHuffman1.Tex	xt = "";	
this.txtHuffman3.T	<pre>Cext = "";</pre>		this.txtHuffman1.Got	tFocus += new	
this.txtHuffman3.G	GotFocus += new		System.E	EventHandler(this.txtHuffman1_Got	Focus);
System	n.EventHandler(this.txtHuffman3_Got	Focus);	//		

```
frmMain.cs
May 02, 04 2:03
                                                                   Page 159/186
    // lblHuffmanMarker1
    11
    this.lblHuffmanMarker1.BackColor = System.Drawing.SystemColors.Window;
this.lblHuffmanMarker1.Enabled = false;
    this.lblHuffmanMarker1.Location = new System.Drawing.Point(72, 8);
    this.lblHuffmanMarker1.Name = "lblHuffmanMarker1";
    this.lblHuffmanMarker1.Size = new System.Drawing.Size(32, 16);
    this.lblHuffmanMarker1.TabIndex = 1;
    this.lblHuffmanMarker1.TextAlign =
                 System.Drawing.ContentAlignment.TopCenter;
    11
    // lblHuffman1
    11
    this.lblHuffman1.Location = new System.Drawing.Point(16, 8);
    this.lblHuffman1.Name = "lblHuffman1";
    this.lblHuffman1.Size = new System.Drawing.Size(64, 16);
    this.lblHuffman1.TabIndex = 0;
    this.lblHuffman1.Text = "Huffman 1:";
    11
    // tabHuffman2
    11
    this.tabHuffman2.Controls.Add(this.btnClearHuffman8);
    this.tabHuffman2.Controls.Add(this.btnAddRandomHuffman8);
    this.tabHuffman2.Controls.Add(this.btnClearHuffman7);
    this.tabHuffman2.Controls.Add(this.btnAddRandomHuffman7);
    this.tabHuffman2.Controls.Add(this.btnClearHuffman6);
    this.tabHuffman2.Controls.Add(this.btnAddRandomHuffman6);
    this.tabHuffman2.Controls.Add(this.btnClearHuffman5);
    this.tabHuffman2.Controls.Add(this.btnAddRandomHuffman5);
    this.tabHuffman2.Controls.Add(this.btnRestoreHuffman8);
    this.tabHuffman2.Controls.Add(this.btnRestoreHuffman7);
    this.tabHuffman2.Controls.Add(this.btnRestoreHuffman6);
    this.tabHuffman2.Controls.Add(this.btnRestoreHuffman5);
    this.tabHuffman2.Controls.Add(this.txtHuffmanOriginal8);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginalMarker8);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginal8);
    this.tabHuffman2.Controls.Add(this.txtHuffman8);
    this.tabHuffman2.Controls.Add(this.lblHuffmanMarker8);
    this.tabHuffman2.Controls.Add(this.lblHuffman8);
    this.tabHuffman2.Controls.Add(this.txtHuffmanOriginal6);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginalMarker6);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginal6);
    this.tabHuffman2.Controls.Add(this.txtHuffman6);
    this.tabHuffman2.Controls.Add(this.lblHuffmanMarker6);
    this.tabHuffman2.Controls.Add(this.lblHuffman6);
    this.tabHuffman2.Controls.Add(this.txtHuffmanOriginal7);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginalMarker7);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginal7);
    this.tabHuffman2.Controls.Add(this.txtHuffman7);
    this.tabHuffman2.Controls.Add(this.lblHuffmanMarker7);
    this.tabHuffman2.Controls.Add(this.lblHuffman7);
    this.tabHuffman2.Controls.Add(this.txtHuffmanOriginal5);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginalMarker5);
    this.tabHuffman2.Controls.Add(this.lblHuffmanOriginal5);
    this.tabHuffman2.Controls.Add(this.txtHuffman5);
    this.tabHuffman2.Controls.Add(this.lblHuffmanMarker5);
    this.tabHuffman2.Controls.Add(this.lblHuffman5);
    this.tabHuffman2.Location = new System.Drawing.Point(4, 22);
    this.tabHuffman2.Name = "tabHuffman2";
    this.tabHuffman2.Size = new System.Drawing.Size(888, 246);
    this.tabHuffman2.TabIndex = 7;
    this.tabHuffman2.Text = "Huffman Tables 2";
    11
    // btnClearHuffman8
    this.btnClearHuffman8.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
```

```
frmMain.cs
May 02, 04 2:03
                                                                    Page 160/186
     this.btnClearHuffman8.Location = new System.Drawing.Point(448, 152);
     this.btnClearHuffman8.Name = "btnClearHuffman8";
     this.btnClearHuffman8.Size = new System.Drawing.Size(40, 16);
    this.btnClearHuffman8.TabIndex = 65;
    this.btnClearHuffman8.Text = "Clear";
     this.btnClearHuffman8.Click += new
                 System.EventHandler(this.btnClearHuffman8 Click);
     // btnAddRandomHuffman8
     11
    this.btnAddRandomHuffman8.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnAddRandomHuffman8.Location = new
                 System.Drawing.Point(496, 152);
    this.btnAddRandomHuffman8.Name = "btnAddRandomHuffman8";
this.btnAddRandomHuffman8.Size = new System.Drawing.Size(48, 16);
    this.btnAddRandomHuffman8.TabIndex = 6\overline{4};
     this.btnAddRandomHuffman8.Text = "Random";
    this.btnAddRandomHuffman8.Click += new
                 System.EventHandler(this.btnAddRandomHuffman8_Click);
     // btnClearHuffman7
     11
    this.btnClearHuffman7.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnClearHuffman7.Location = new System.Drawing.Point(8, 152);
    this.btnClearHuffman7.Name = "btnClearHuffman7";
    this.btnClearHuffman7.Size = new System.Drawing.Size(40, 16);
    this.btnClearHuffman7.TabIndex = 63;
     this.btnClearHuffman7.Text = "Clear";
    this.btnClearHuffman7.Click += new
                 System.EventHandler(this.btnClearHuffman7_Click);
    // btnAddRandomHuffman7
     11
    this.btnAddRandomHuffman7.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnAddRandomHuffman7.Location = new System.Drawing.Point(56, 152);
    this.btnAddRandomHuffman7.Name = "btnAddRandomHuffman7";
     this.btnAddRandomHuffman7.Size = new System.Drawing.Size(48, 16);
    this.btnAddRandomHuffman7.TabIndex = 6\hat{2};
    this.btnAddRandomHuffman7.Text = "Random";
    this.btnAddRandomHuffman7.Click += new
                 System.EventHandler(this.btnAddRandomHuffman7_Click);
     // btnClearHuffman6
    this.btnClearHuffman6.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                      System.Drawing.FontStyle.Regular,
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnClearHuffman6.Location = new System.Drawing.Point(448, 32);
    this.btnClearHuffman6.Name = "btnClearHuffman6";
    this.btnClearHuffman6.Size = new System.Drawing.Size(40, 16);
    this.btnClearHuffman6.TabIndex = 6\hat{1};
    this.btnClearHuffman6.Text = "Clear";
    this.btnClearHuffman6.Click += new
                 System.EventHandler(this.btnClearHuffman6_Click);
     // btnAddRandomHuffman6
     11
     this.btnAddRandomHuffman6.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
```

```
frmMain.cs
May 02, 04 2:03
                                                                     Page 161/186
                      System.Drawing.FontStyle.Regular,
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnAddRandomHuffman6.Location = new System.Drawing.Point(496, 32);
    this.btnAddRandomHuffman6.Name = "btnAddRandomHuffman6";
this.btnAddRandomHuffman6.Size = new System.Drawing.Size(48, 16);
     this.btnAddRandomHuffman6.TabIndex = 60;
     this.btnAddRandomHuffman6.Text = "Random";
     this.btnAddRandomHuffman6.Click += new
                 System.EventHandler(this.btnAddRandomHuffman6_Click);
     11
     // btnClearHuffman5
     11
     this.btnClearHuffman5.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                      System.Drawing.FontStyle.Regular,
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnClearHuffman5.Location = new System.Drawing.Point(8, 32);
    this.btnClearHuffman5.Name = "btnClearHuffman5";
this.btnClearHuffman5.Size = new System.Drawing.Size(40, 16);
     this.btnClearHuffman5.TabIndex = 59;
     this.btnClearHuffman5.Text = "Clear";
     this.btnClearHuffman5.Click += new
                 System.EventHandler(this.btnClearHuffman5_Click);
     11
    // btnAddRandomHuffman5
     11
     this.btnAddRandomHuffman5.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                      System.Drawing.FontStyle.Regular,
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnAddRandomHuffman5.Location = new System.Drawing.Point(56, 32);
     this.btnAddRandomHuffman5.Name = "btnAddRandomHuffman5";
     this.btnAddRandomHuffman5.Size = new System.Drawing.Size(48, 16);
     this.btnAddRandomHuffman5.TabIndex = 5\overline{8};
     this.btnAddRandomHuffman5.Text = "Random";
    this.btnAddRandomHuffman5.Click += new
                 System.EventHandler(this.btnAddRandomHuffman5_Click);
     // btnRestoreHuffman8
     11
     this.btnRestoreHuffman8.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                      System.Drawing.FontStyle.Regular,
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnRestoreHuffman8.Location = new System.Drawing.Point(496, 208);
     this.btnRestoreHuffman8.Name = "btnRestoreHuffman8";
    this.btnRestoreHuffman8.Size = new System.Drawing.Size(48, 16);
    this.btnRestoreHuffman8.TabIndex = 5\overline{5};
     this.btnRestoreHuffman8.Text = "Restore";
     this.btnRestoreHuffman8.Click += new
                 System.EventHandler(this.btnRestoreHuffman8_Click);
    // btnRestoreHuffman7
     11
     this.btnRestoreHuffman7.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                      System.Drawing.FontStyle.Regular,
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnRestoreHuffman7.Location = new System.Drawing.Point(56, 208);
    this.btnRestoreHuffman7.Name = "btnRestoreHuffman7";
     this.btnRestoreHuffman7.Size = new System.Drawing.Size(48, 16);
     this.btnRestoreHuffman7.TabIndex = 54;
     this.btnRestoreHuffman7.Text = "Restore";
     this.btnRestoreHuffman7.Click += new
                 System.EventHandler(this.btnRestoreHuffman7_Click);
     // btnRestoreHuffman6
     11
     this.btnRestoreHuffman6.Font = new
```

```
frmMain.cs
May 02, 04 2:03
                                                                     Page 162/186
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                      System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnRestoreHuffman6.Location = new System.Drawing.Point(496, 88);
this.btnRestoreHuffman6.Name = "btnRestoreHuffman6";
this.btnRestoreHuffman6.Size = new System.Drawing.Size(48, 16);
    this.btnRestoreHuffman6.TabIndex = 53;
     this.btnRestoreHuffman6.Text = "Restore";
    this.btnRestoreHuffman6.Click += new
                 System.EventHandler(this.btnRestoreHuffman6 Click);
    // btnRestoreHuffman5
     11
    this.btnRestoreHuffman5.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     this.btnRestoreHuffman5.Location = new System.Drawing.Point(56, 88);
    this.btnRestoreHuffman5.Name = "btnRestoreHuffman5";
     this.btnRestoreHuffman5.Size = new System.Drawing.Size(48, 16);
     this.btnRestoreHuffman5.TabIndex = 52;
     this.btnRestoreHuffman5.Text = "Restore";
    this.btnRestoreHuffman5.Click += new
                 System.EventHandler(this.btnRestoreHuffman5_Click);
    // txtHuffmanOriginal8
     11
    this.txtHuffmanOriginal8.AutoSize = false;
    this.txtHuffmanOriginal8.Enabled = false;
    this.txtHuffmanOriginal8.Location = new System.Drawing.Point(552, 187);
    this.txtHuffmanOriginal8.Multiline = true;
     this.txtHuffmanOriginal8.Name = "txtHuffmanOriginal8";
    this.txtHuffmanOriginal8.ScrollBars =
                 System.Windows.Forms.ScrollBars.Horizontal;
     this.txtHuffmanOriginal8.Size = new System.Drawing.Size(328, 48);
    this.txtHuffmanOriginal8.TabIndex = 50;
    this.txtHuffmanOriginal8.TabStop = false;
    this.txtHuffmanOriginal8.Text = "";
     11
     // lblHuffmanOriginalMarker8
    this.lblHuffmanOriginalMarker8.BackColor =
                 System.Drawing.SystemColors.Window;
    this.lblHuffmanOriginalMarker8.Enabled = false;
    this.lblHuffmanOriginalMarker8.Location = new
                 System.Drawing.Point(512, 184);
    this.lblHuffmanOriginalMarker8.Name = "lblHuffmanOriginalMarker8";
    this.lblHuffmanOriginalMarker8.Size = new System.Drawing.Size(32, 16);
     this.lblHuffmanOriginalMarker8.TabIndex = 49;
    this.lblHuffmanOriginalMarker8.TextAlign =
                 System.Drawing.ContentAlignment.TopCenter;
    // lblHuffmanOriginal8
     11
    this.lblHuffmanOriginal8.Location = new System.Drawing.Point(456, 184);
    this.lblHuffmanOriginal8.Name = "lblHuffmanOriginal8";
    this.lblHuffmanOriginal8.Size = new System.Drawing.Size(64, 16);
     this.lblHuffmanOriginal8.TabIndex = 4\hat{8};
    this.lblHuffmanOriginal8.Text = "Original 8:";
     11
    // txtHuffman8
    this.txtHuffman8.AutoSize = false;
     this.txtHuffman8.Location = new System.Drawing.Point(552, 131);
    this.txtHuffman8.Multiline = true;
    this.txtHuffman8.Name = "txtHuffman8";
    this.txtHuffman8.ScrollBars =
```

Page 164/186

frmMain.cs

```
frmMain.cs
May 02, 04 2:03
                                                                    Page 163/186
                                                                                        May 02, 04 2:03
     this.txtHuffman8.TabIndex = 32;
                                                                                             this.txtHuffman6.Text = "";
     this.txtHuffman8.Text = "";
                                                                                             this.txtHuffman6.GotFocus += new
     this.txtHuffman8.GotFocus += new
                                                                                                          System.EventHandler(this.txtHuffman6_GotFocus);
                 System.EventHandler(this.txtHuffman8 GotFocus);
                                                                                             // lblHuffmanMarker6
    // lblHuffmanMarker8
                                                                                             11
                                                                                             this.lblHuffmanMarker6.BackColor = System.Drawing.SystemColors.Window;
    11
     this.lblHuffmanMarker8.BackColor = System.Drawing.SystemColors.Window;
                                                                                             this.lblHuffmanMarker6.Enabled = false;
                                                                                             this.lblHuffmanMarker6.Location = new System.Drawing.Point(512, 8);
    this.lblHuffmanMarker8.Enabled = false;
                                                                                             this.lblHuffmanMarker6.Name = "lblHuffmanMarker6";
this.lblHuffmanMarker6.Size = new System.Drawing.Size(32, 16);
     this.lblHuffmanMarker8.Location = new System.Drawing.Point(512, 128);
    this.lblHuffmanMarker8.Name = "lblHuffmanMarker8";
this.lblHuffmanMarker8.Size = new System.Drawing.Size(32, 16);
                                                                                             this.lblHuffmanMarker6.TabIndex = 4\hat{2};
     this.lblHuffmanMarker8.TabIndex = 47;
                                                                                             this.lblHuffmanMarker6.TextAlign =
     this.lblHuffmanMarker8.TextAlign =
                                                                                                          System.Drawing.ContentAlignment.TopCenter;
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                             // lblHuffman6
     1
     // lblHuffman8
                                                                                             11
     11
                                                                                             this.lblHuffman6.Location = new System.Drawing.Point(456, 8);
     this.lblHuffman8.Location = new System.Drawing.Point(456, 128);
                                                                                             this.lblHuffman6.Name = "lblHuffman6";
     this.lblHuffman8.Name = "lblHuffman8";
                                                                                             this.lblHuffman6.Size = new System.Drawing.Size(64, 16);
     this.lblHuffman8.Size = new System.Drawing.Size(64, 16);
                                                                                             this.lblHuffman6.TabIndex = 41;
     this.lblHuffman8.TabIndex = 46;
                                                                                             this.lblHuffman6.Text = "Huffman 6:";
     this.lblHuffman8.Text = "Huffman 8:";
                                                                                             11
                                                                                             // txtHuffmanOriginal7
    11
     // txtHuffmanOriginal6
                                                                                             11
     11
                                                                                             this.txtHuffmanOriginal7.AutoSize = false;
                                                                                             this.txtHuffmanOriginal7.Enabled = false;
     this.txtHuffmanOriginal6.AutoSize = false;
     this.txtHuffmanOriginal6.Enabled = false;
                                                                                             this.txtHuffmanOriginal7.Location = new System.Drawing.Point(112, 187);
     this.txtHuffmanOriginal6.Location = new System.Drawing.Point(552, 67);
                                                                                             this.txtHuffmanOriginal7.Multiline = true;
     this.txtHuffmanOriginal6.Multiline = true;
                                                                                             this.txtHuffmanOriginal7.Name = "txtHuffmanOriginal7";
     this.txtHuffmanOriginal6.Name = "txtHuffmanOriginal6";
                                                                                             this.txtHuffmanOriginal7.ScrollBars =
     this.txtHuffmanOriginal6.ScrollBars =
                                                                                                          System.Windows.Forms.ScrollBars.Horizontal;
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                             this.txtHuffmanOriginal7.Size = new System.Drawing.Size(328, 48);
     this.txtHuffmanOriginal6.Size = new System.Drawing.Size(328, 48);
                                                                                             this.txtHuffmanOriginal7.TabIndex = 40;
     this.txtHuffmanOriginal6.TabIndex = 45;
                                                                                             this.txtHuffmanOriginal7.TabStop = false;
this.txtHuffmanOriginal7.Text = "";
    this.txtHuffmanOriginal6.TabStop = false;
this.txtHuffmanOriginal6.Text = "";
                                                                                             11
                                                                                             // lblHuffmanOriginalMarker7
     11
     // lblHuffmanOriginalMarker6
                                                                                             11
                                                                                             this.lblHuffmanOriginalMarker7.BackColor =
     11
     this.lblHuffmanOriginalMarker6.BackColor =
                                                                                                          System.Drawing.SystemColors.Window;
                 System.Drawing.SystemColors.Window;
                                                                                             this.lblHuffmanOriginalMarker7.Enabled = false;
     this.lblHuffmanOriginalMarker6.Enabled = false;
                                                                                             this.lblHuffmanOriginalMarker7.Location = new
     this.lblHuffmanOriginalMarker6.Location = new
                                                                                                          System.Drawing.Point(72, 184);
                 System.Drawing.Point(512, 64);
                                                                                             this.lblHuffmanOriginalMarker7.Name = "lblHuffmanOriginalMarker7";
     this.lblHuffmanOriginalMarker6.Name = "lblHuffmanOriginalMarker6";
                                                                                             this.lblHuffmanOriginalMarker7.Size = new System.Drawing.Size(32, 16);
    this.lblHuffmanOriginalMarker6.Size = new System.Drawing.Size(32, 16);
                                                                                             this.lblHuffmanOriginalMarker7.TabIndex = 39;
     this.lblHuffmanOriginalMarker6.TabIndex = 44;
                                                                                             this.lblHuffmanOriginalMarker7.TextAlign =
    this.lblHuffmanOriginalMarker6.TextAlign =
                                                                                                          System.Drawing.ContentAlignment.TopCenter;
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                             // lblHuffmanOriginal7
     // lblHuffmanOriginal6
                                                                                             this.lblHuffmanOriginal7.Location = new System.Drawing.Point(16, 184);
     11
    this.lblHuffmanOriginal6.Location = new System.Drawing.Point(456, 64);
                                                                                             this.lblHuffmanOriginal7.Name = "lblHuffmanOriginal7";
    this.lblHuffmanOriginal6.Name = "lblHuffmanOriginal6";
                                                                                             this.lblHuffmanOriginal7.Size = new System.Drawing.Size(64, 16);
     this.lblHuffmanOriginal6.Size = new System.Drawing.Size(64, 16);
                                                                                             this.lblHuffmanOriginal7.TabIndex = 38;
     this.lblHuffmanOriginal6.TabIndex = 43;
                                                                                             this.lblHuffmanOriginal7.Text = "Original 7:";
    this.lblHuffmanOriginal6.Text = "Original 6:";
                                                                                             11
    11
                                                                                             // txtHuffman7
     // txtHuffman6
                                                                                             11
                                                                                             this.txtHuffman7.AutoSize = false;
     this.txtHuffman6.AutoSize = false;
                                                                                             this.txtHuffman7.Location = new System.Drawing.Point(112, 131);
                                                                                             this.txtHuffman7.Multiline = true;
     this.txtHuffman6.Location = new System.Drawing.Point(552, 11);
     this.txtHuffman6.Multiline = true;
                                                                                             this.txtHuffman7.Name = "txtHuffman7";
     this.txtHuffman6.Name = "txtHuffman6";
                                                                                             this.txtHuffman7.ScrollBars =
     this.txtHuffman6.ScrollBars =
                                                                                                          System.Windows.Forms.ScrollBars.Horizontal;
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                             this.txtHuffman7.Size = new System.Drawing.Size(328, 48);
     this.txtHuffman6.Size = new System.Drawing.Size(328, 48);
                                                                                             this.txtHuffman7.TabIndex = 3\hat{1};
     this.txtHuffman6.TabIndex = 29;
                                                                                             this.txtHuffman7.Text = "";
```

May 02, 04 2:03	frmMain.cs	Page 165/186	May 02, 04 2:03	frmMain.cs	Page 166/186
this.txtHuffman7.Go	tFocus += new		System.	EventHandler(this.txtHuffman5_GotE	/ocus);
//	EventHandler(this.txtHuffman/_Got	Focus);	// // lblHuffmanMarker	5	
// lblHuffmanMarker	7		// this lblHuffmanMark	er5 BackColor - System Drawing Sys	stemColors Window.
// this.lblHuffmanMark	er7.BackColor = System.Drawing.Sy	stemColors.Window;	this.lblHuffmanMark	er5.Enabled = false;	, temcorors.window,
this.lblHuffmanMark	er7.Enabled = false;	Deint (72 128)	this.lblHuffmanMark	er5.Location = new System.Drawing.	.Point(72, 8);
this.lblHuffmanMark	er7.Name = "lblHuffmanMarker7";	.Point(72, 128);	this.lblHuffmanMark	er5.Name = "IDIHUIIManMarker5"; er5.Size = new System.Drawing.Size	e(32, 16);
this.lblHuffmanMark	er7.Size = new System.Drawing.Siz	e(32, 16);	this.lblHuffmanMark	er5.TabIndex = 30;	
this.lblHuffmanMark	er/.Tabindex = 3/; er7.TextAlign =		this.lblHuttmanMark System.	er5.TextAlign = Drawing.ContentAlignment.TopCenter	^ <b>:</b>
System.	Drawing.ContentAlignment.TopCente	er;	//	2144119.0010011119	
// // lblHuffman7			// lblHuffman5 //		
//			this.lblHuffman5.Lo	cation = new System.Drawing.Point	(16, 8);
this.lblHuffman7.Lo	<pre>cation = new System.Drawing.Point mo = "lbl#uffmap7".</pre>	(16, 128);	this.lblHuffman5.Na	<pre>me = "lblHuffman5"; zo = now System Drawing Size(64, 1</pre>	6).
this.lblHuffman7.Si	.ze = new System.Drawing.Size(64,	16);	this.lblHuffman5.Ta	bIndex = 28;	.0),
this.lblHuffman7.Ta	bIndex = 36;		this.lblHuffman5.Te	<pre>xt = "Huffman 5:";</pre>	
//	xt = "Hullman /:";		// // tabQuantizer		
// txtHuffmanOrigin	al5		//		
// this.txtHuffmanOrig	inal5.AutoSize = false:		this.tabQuantizer.C	ontrols.Add(this.txtQuantizerTable ontrols.Add(this.lblQuantizerTable	:Num4); :Num4);
this.txtHuffmanOrig	inal5.Enabled = false;		this.tabQuantizer.C	ontrols.Add(this.txtQuantizerTable	eNum3);
this.txtHuffmanOrig	inal5.Location = new System.Drawi	ng.Point(112, 67);	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerTable	•Num3); •Num2):
this.txtHuffmanOrig	inal5.Name = "txtHuffmanOriginal5	";	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerTable	eNum2);
this.txtHuffmanOrig	inal5.ScrollBars = Windows Forms ScrollBars Horizont	al·	this.tabQuantizer.C	ontrols.Add(this.txtQuantizerTable	Num1);
this.txtHuffmanOrig	<pre>inal5.Size = new System.Drawing.S</pre>	ize(328, 48);	this.tabQuantizer.C	ontrols.Add(this.btnClearQuantizer	:4);
this.txtHuffmanOrig	<pre>inal5.TabIndex = 35; inal5_TabStop = false;</pre>		this.tabQuantizer.C	ontrols.Add(this.btnAddRandomQuant	:izer4);
this.txtHuffmanOrig	<pre>inal5.Text = "";</pre>		this.tabQuantizer.C	ontrols.Add(this.btnAddRandomQuant	:izer3);
//	al Markars		this.tabQuantizer.C	ontrols.Add(this.btnClearQuantizer	:2);
// ibinulimanorigin	la IMai Kei J		this.tabQuantizer.C	ontrols.Add(this.btnClearQuantizer	112012); 1);
this.lblHuffmanOrig	inalMarker5.BackColor =		this.tabQuantizer.C	ontrols.Add(this.btnAddRandomQuant	<pre>:izer1); cor4);</pre>
this.lblHuffmanOrig	<pre>inalMarker5.Enabled = false;</pre>		this.tabQuantizer.C	ontrols.Add(this.btnRestoreQuantiz	2er3);
this.lblHuffmanOrig	pinalMarker5.Location = new		this.tabQuantizer.C	ontrols.Add(this.btnRestoreQuantiz	<pre>ser2);</pre>
this.lblHuffmanOrig	<pre>inalMarker5.Name = "lblHuffmanOri</pre>	ginalMarker5";	this.tabQuantizer.C	ontrols.Add(this.txtQuantizerOrigi	inal4);
this.lblHuffmanOrig	inalMarker5.Size = new System.Dra	wing.Size(32, 16);	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerOrigi	<pre>inalMarker4);</pre>
this.lblHuffmanOrig	<pre>inalMarker5.labindex = 34; inalMarker5.TextAlign =</pre>		this.tabQuantizer.C	ontrols.Add(this.ibiQuantizerOrigi ontrols.Add(this.txtOuantizer4);	.na14);
System.	Drawing.ContentAlignment.TopCente	er;	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerMarke	er4);
// // lblHuffmanOrigin	al5		this.tabQuantizer.C	ontrols.Add(this.lblQuantizer4); ontrols.Add(this.txtOuantizerOrigi	inal2);
//			this.tabQuantizer.C	ontrols.Add(this.lblQuantizerOrigi	<pre>inalMarker2);</pre>
this.lblHuffmanOrig	inal5.Location = new System.Drawi	ng.Point(16, 64);	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerOrigi	.nal2);
this.lblHuffmanOrig	inal5.Size = new System.Drawing.S	ize(64, 16);	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerMarke	er2);
this.lblHuffmanOrig	<pre>inal5.TabIndex = 33; inal5 Text = "Original 5.".</pre>		this.tabQuantizer.C	<pre>ontrols.Add(this.lblQuantizer2); ontrols Add(this txtQuantizerOrigi</pre>	inal3).
//	inaio.ieke originar 5. /		this.tabQuantizer.C	ontrols.Add(this.lblQuantizerOrigi	LnalMarker3);
// txtHuffman5			this.tabQuantizer.C	ontrols.Add(this.lblQuantizerOrigi	_nal3);
this.txtHuffman5.Au	toSize = false;		this.tabQuantizer.C	ontrols.Add(this.lblQuantizerMarke	er3);
this.txtHuffman5.Lo	cation = new System.Drawing.Point	(112, 11);	this.tabQuantizer.C	<pre>ontrols.Add(this.lblQuantizer3); optrols.Add(this.tutQuantizer3);</pre>	inall).
this.txtHuffman5.Na	<pre>ime = "txtHuffman5";</pre>		this.tabQuantizer.C	ontrols.Add(this.lblQuantizerOrigi	inalMarker1);
this.txtHuffman5.Sc	rollBars = Windows Forms SamallBans Newigers	21.	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerOrigi	.nal1);
.system this.txtHuffman5.Si	<pre>windows.forms.ScrollBars.Horl20nt ze = new System.Drawing.Size(328,</pre>	48);	this.tabQuantizer.C	ontrols.Add(this.lblQuantizerI);	erl);
this.txtHuffman5.Ta	bIndex = 27;		this.tabQuantizer.C	ontrols.Add(this.lblQuantizer1);	
this.txtHuffman5.Go	xt = ""; tFocus += new		this.tabQuantizer.L	<pre>ocation = new System.Drawing.Point ame = "tabOuantizer";</pre>	(4, 22);
				· · · · · · · · · · · · · · · · · · ·	

```
frmMain.cs
                                  frmMain.cs
May 02, 04 2:03
                                                                   Page 167/186
                                                                                      May 02, 04 2:03
                                                                                                                                                          Page 168/186
     this.tabOuantizer.Size = new System.Drawing.Size(888, 246);
    this.tabQuantizer.TabIndex = 1;
                                                                                           this.txtQuantizerTableNum1.BackColor =
    this.tabQuantizer.Text = "Quantizer Table";
                                                                                                        System.Drawing.SystemColors.Window;
                                                                                           this.txtOuantizerTableNum1.Enabled = false;
    11
    // txtQuantizerTableNum4
                                                                                           this.txtQuantizerTableNum1.Location = new System.Drawing.Point(72, 32);
                                                                                           this.txtQuantizerTableNum1.Name = "txtQuantizerTableNum1";
    11
    this.txtOuantizerTableNum4.BackColor =
                                                                                           this.txtQuantizerTableNum1.Size = new System.Drawing.Size(32, 16);
                 System.Drawing.SystemColors.Window;
                                                                                           this.txtQuantizerTableNum1.TabIndex = 67;
    this.txtQuantizerTableNum4.Enabled = false;
                                                                                           this.txtQuantizerTableNum1.TextAlign =
    this.txtQuantizerTableNum4.Location = new
System.Drawing.Point(512, 152);
                                                                                                        System.Drawing.ContentAlignment.TopCenter;
    this.txtQuantizerTableNum4.Name = "txtQuantizerTableNum4";
                                                                                           // lblQuantizerTableNum1
    this.txtQuantizerTableNum4.Size = new System.Drawing.Size(32, 16);
                                                                                           11
    this.txtQuantizerTableNum4.TabIndex = 73;
                                                                                           this.lblQuantizerTableNum1.Location = new System.Drawing.Point(8, 32);
    this.txtQuantizerTableNum4.TextAlign =
                                                                                           this.lblQuantizerTableNum1.Name = "lblQuantizerTableNum1";
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                           this.lblQuantizerTableNum1.Size = new System.Drawing.Size(56, 16);
                                                                                           this.lblQuantizerTableNum1.TabIndex = 66;
    // lblQuantizerTableNum4
                                                                                           this.lblQuantizerTableNum1.Text = "Table #:";
    11
                                                                                           11
                                                                                           // btnClearQuantizer4
    this.lblQuantizerTableNum4.Location = new
                 System.Drawing.Point(448, 152);
                                                                                           11
    this.lblQuantizerTableNum4.Name = "lblQuantizerTableNum4";
this.lblQuantizerTableNum4.Size = new System.Drawing.Size(56, 16);
                                                                                           this.btnClearQuantizer4.Font = new
                                                                                                        System.Drawing.Font("Microsoft Sans Serif", 7F,
    this.lblQuantizerTableNum4.TabIndex = 72;
                                                                                                            System.Drawing.FontStyle.Regular,
    this.lblQuantizerTableNum4.Text = "Table #:";
                                                                                                            System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    11
                                                                                           this.btnClearQuantizer4.Location = new System.Drawing.Point(448, 176);
    // txtQuantizerTableNum3
                                                                                           this.btnClearQuantizer4.Name = "btnClearQuantizer4";
                                                                                           this.btnClearQuantizer4.Size = new System.Drawing.Size(40, 16);
    11
    this.txtQuantizerTableNum3.BackColor =
                                                                                           this.btnClearQuantizer4.TabIndex = 65;
                 System.Drawing.SystemColors.Window;
                                                                                           this.btnClearQuantizer4.Text = "Clear";
    this.txtQuantizerTableNum3.Enabled = false;
                                                                                           this.btnClearQuantizer4.Click += new
    this.txtQuantizerTableNum3.Location = new System.Drawing.Point(72, 152);
                                                                                                        System.EventHandler(this.btnClearQuantizer4_Click);
    this.txtQuantizerTableNum3.Name = "txtQuantizerTableNum3";
    this.txtQuantizerTableNum3.Size = new System.Drawing.Size(32, 16);
                                                                                           // btnAddRandomQuantizer4
    this.txtQuantizerTableNum3.TabIndex = 71;
                                                                                           11
                                                                                           this.btnAddRandomQuantizer4.Font = new
    this.txtQuantizerTableNum3.TextAlign =
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                                        System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                                            System.Drawing.FontStyle.Regular,
    // lblQuantizerTableNum3
                                                                                                            System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    11
                                                                                           this.btnAddRandomQuantizer4.Location = new
    this.lblQuantizerTableNum3.Location = new System.Drawing.Point(8, 152);
                                                                                                        System.Drawing.Point(496, 176);
    this.lblQuantizerTableNum3.Name = "lblQuantizerTableNum3";
                                                                                           this.btnAddRandomQuantizer4.Name = "btnAddRandomQuantizer4";
    this.lblQuantizerTableNum3.Size = new System.Drawing.Size(56, 16);
                                                                                           this.btnAddRandomQuantizer4.Size = new System.Drawing.Size(48, 16);
    this.lblQuantizerTableNum3.TabIndex = 70;
                                                                                           this.btnAddRandomQuantizer4.TabIndex = 6\overline{4};
    this.lblQuantizerTableNum3.Text = "Table #:";
                                                                                           this.btnAddRandomQuantizer4.Text = "Random";
    11
                                                                                           this.btnAddRandomQuantizer4.Click += new
    // txtQuantizerTableNum2
                                                                                                        System.EventHandler(this.btnAddRandomQuantizer4_Click);
    11
    this.txtOuantizerTableNum2.BackColor =
                                                                                           // btnClearQuantizer3
                 System.Drawing.SystemColors.Window;
    this.txtQuantizerTableNum2.Enabled = false;
                                                                                           this.btnClearQuantizer3.Font = new
    this.txtQuantizerTableNum2.Location = new
                                                                                                        System.Drawing.Font("Microsoft Sans Serif", 7F,
                 System.Drawing.Point(512, 32);
                                                                                                            System.Drawing.FontStyle.Regular,
    this.txtQuantizerTableNum2.Name = "txtQuantizerTableNum2";
                                                                                                            System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.txtQuantizerTableNum2.Size = new System.Drawing.Size(32, 16);
                                                                                           this.btnClearQuantizer3.Location = new System.Drawing.Point(8, 176);
    this.txtQuantizerTableNum2.TabIndex = 69;
                                                                                           this.btnClearQuantizer3.Name = "btnClearQuantizer3";
    this.txtQuantizerTableNum2.TextAlign =
                                                                                           this.btnClearQuantizer3.Size = new System.Drawing.Size(40, 16);
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                           this.btnClearQuantizer3.TabIndex = 63;
                                                                                           this.btnClearQuantizer3.Text = "Clear";
    // lblQuantizerTableNum2
                                                                                           this.btnClearQuantizer3.Click += new
    11
                                                                                                        System.EventHandler(this.btnClearQuantizer3_Click);
    this.lblQuantizerTableNum2.Location = new
                 System.Drawing.Point(448, 32);
                                                                                           // btnAddRandomQuantizer3
    this.lblQuantizerTableNum2.Name = "lblQuantizerTableNum2";
                                                                                           11
    this.lblQuantizerTableNum2.Size = new System.Drawing.Size(56, 16);
                                                                                           this.btnAddRandomQuantizer3.Font = new
    this.lblQuantizerTableNum2.TabIndex = 68;
                                                                                                        System.Drawing.Font("Microsoft Sans Serif", 7F,
    this.lblQuantizerTableNum2.Text = "Table #:";
                                                                                                            System.Drawing.FontStyle.Regular,
                                                                                                            System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
     // txtQuantizerTableNum1
                                                                                           this.btnAddRandomQuantizer3.Location = new
```

Page 170/186

```
frmMain.cs
                                                                                                                        frmMain.cs
May 02, 04 2:03
                                                                   Page 169/186
                                                                                      May 02, 04 2:03
                 System.Drawing.Point(56, 176);
                                                                                                       System.Drawing.Font("Microsoft Sans Serif", 7F,
    this.btnAddRandomQuantizer3.Name = "btnAddRandomQuantizer3";
                                                                                                            System.Drawing.FontStyle.Regular,
    this.btnAddRandomQuantizer3.Size = new System.Drawing.Size(48, 16);
                                                                                                            System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnAddRandomQuantizer3.TabIndex = 62;
this.btnAddRandomQuantizer3.Text = "Random";
                                                                                           this.btnRestoreQuantizer4.Location = new
                                                                                                       System.Drawing.Point(496, 224);
    this.btnAddRandomQuantizer3.Click += new
                                                                                           this.btnRestoreQuantizer4.Name = "btnRestoreQuantizer4";
                 System.EventHandler(this.btnAddRandomQuantizer3 Click);
                                                                                           this.btnRestoreQuantizer4.Size = new System.Drawing.Size(48, 16);
                                                                                           this.btnRestoreQuantizer4.TabIndex = 54;
     11
    // btnClearQuantizer2
                                                                                           this.btnRestoreQuantizer4.Text = "Restore";
                                                                                           this.btnRestoreQuantizer4.Click += new
    11
    this.btnClearQuantizer2.Font = new
                                                                                                       System.EventHandler(this.btnRestoreQuantizer4_Click);
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                     System.Drawing.FontStyle.Regular,
                                                                                           // btnRestoreOuantizer3
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                           11
    this.btnClearQuantizer2.Location = new System.Drawing.Point(448, 56);
                                                                                           this.btnRestoreQuantizer3.Font = new
    this.btnClearQuantizer2.Name = "btnClearQuantizer2";
this.btnClearQuantizer2.Size = new System.Drawing.Size(40, 16);
                                                                                                       System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                                           System.Drawing.FontStyle.Regular,
    this.btnClearQuantizer2.TabIndex = 61;
                                                                                                            System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnClearQuantizer2.Text = "Clear";
                                                                                           this.btnRestoreQuantizer3.Location = new System.Drawing.Point(56, 224);
    this.btnClearQuantizer2.Click += new
                                                                                           this.btnRestoreQuantizer3.Name = "btnRestoreQuantizer3";
                 System.EventHandler(this.btnClearQuantizer2_Click);
                                                                                           this.btnRestoreQuantizer3.Size = new System.Drawing.Size(48, 16);
    11
                                                                                           this.btnRestoreQuantizer3.TabIndex = 53;
    // btnAddRandomQuantizer2
                                                                                           this.btnRestoreQuantizer3.Text = "Restore";
    11
                                                                                           this.btnRestoreQuantizer3.Click += new
    this.btnAddRandomQuantizer2.Font = new
                                                                                                       System.EventHandler(this.btnRestoreQuantizer3_Click);
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                           // btnRestoreQuantizer2
                      System.Drawing.FontStyle.Regular,
                      System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                           11
    this.btnAddRandomQuantizer2.Location = new
                                                                                           this.btnRestoreQuantizer2.Font = new
                 System.Drawing.Point(496, 56);
                                                                                                       System.Drawing.Font("Microsoft Sans Serif", 7F,
    this.btnAddRandomQuantizer2.Name = "btnAddRandomQuantizer2";
                                                                                                            System.Drawing.FontStyle.Regular,
    this.btnAddRandomQuantizer2.Size = new System.Drawing.Size(48, 16);
                                                                                                             System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnAddRandomQuantizer2.TabIndex = 60;
                                                                                           this.btnRestoreQuantizer2.Location = new System.Drawing.Point (496, 104);
    this.btnAddRandomQuantizer2.Text = "Random";
                                                                                           this.btnRestoreQuantizer2.Name = "btnRestoreQuantizer2";
    this.btnAddRandomQuantizer2.Click += new
                                                                                           this.btnRestoreQuantizer2.Size = new System.Drawing.Size(48, 16);
                                                                                           this.btnRestoreQuantizer2.TabIndex = 52;
                 System.EventHandler(this.btnAddRandomQuantizer2_Click);
                                                                                           this.btnRestoreQuantizer2.Text = "Restore";
    11
    // btnClearQuantizer1
                                                                                           this.btnRestoreQuantizer2.Click += new
                                                                                                       System.EventHandler(this.btnRestoreQuantizer2_Click);
    11
    this.btnClearOuantizer1.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                           // btnRestoreQuantizer1
                     System.Drawing.FontStyle.Regular,
                                                                                           11
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                           this.btnRestoreOuantizer1.Font = new
    this.btnClearQuantizer1.Location = new System.Drawing.Point(8, 56);
                                                                                                       System.Drawing.Font("Microsoft Sans Serif", 7F,
    this.btnClearQuantizer1.Name = "btnClearQuantizer1";
                                                                                                            System.Drawing.FontStyle.Regular,
    this.btnClearQuantizer1.Size = new System.Drawing.Size(40, 16);
                                                                                                             System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
    this.btnClearQuantizer1.TabIndex = 59;
                                                                                           this.btnRestoreQuantizer1.Location = new System.Drawing.Point(56, 104);
    this.btnClearQuantizer1.Text = "Clear";
                                                                                           this.btnRestoreQuantizer1.Name = "btnRestoreQuantizer1";
    this.btnClearOuantizer1.Click += new
                                                                                           this.btnRestoreQuantizer1.Size = new System.Drawing.Size(48, 16);
                 System.EventHandler(this.btnClearQuantizer1_Click);
                                                                                           this.btnRestoreQuantizer1.TabIndex = 51;
                                                                                           this.btnRestoreQuantizer1.Text = "Restore";
    // btnAddRandomQuantizer1
                                                                                           this.btnRestoreQuantizer1.Click += new
    11
                                                                                                       System.EventHandler(this.btnRestoreQuantizer1_Click);
    this.btnAddRandomQuantizer1.Font = new
                 System.Drawing.Font("Microsoft Sans Serif", 7F,
                                                                                           // txtQuantizerOriginal4
                     System.Drawing.FontStyle.Regular,
                                                                                           11
                     System.Drawing.GraphicsUnit.Point, ((System.Byte)(0)));
                                                                                           this.txtQuantizerOriginal4.AutoSize = false;
    this.btnAddRandomQuantizer1.Location = new
                                                                                           this.txtQuantizerOriginal4.Enabled = false;
                 System.Drawing.Point(56, 56);
                                                                                           this.txtQuantizerOriginal4.Location = new
    this.btnAddRandomQuantizer1.Name = "btnAddRandomQuantizer1";
                                                                                                       System.Drawing.Point(552, 187);
    this.btnAddRandomQuantizer1.Size = new System.Drawing.Size(48, 16);
                                                                                           this.txtQuantizerOriginal4.Multiline = true;
    this.btnAddRandomQuantizer1.TabIndex = 58;
                                                                                           this.txtQuantizerOriginal4.Name = "txtQuantizerOriginal4";
    this.btnAddRandomQuantizer1.Text = "Random";
                                                                                           this.txtQuantizerOriginal4.ScrollBars =
    this.btnAddRandomQuantizer1.Click += new
                                                                                                       System.Windows.Forms.ScrollBars.Horizontal;
                 System.EventHandler(this.btnAddRandomQuantizer1_Click);
                                                                                           this.txtQuantizerOriginal4.Size = new System.Drawing.Size(328, 48);
                                                                                           this.txtQuantizerOriginal4.TabIndex = 50;
    // btnRestoreQuantizer4
                                                                                           this.txtQuantizerOriginal4.TabStop = false;
    11
                                                                                           this.txtQuantizerOriginal4.Text = "";
    this.btnRestoreQuantizer4.Font = new
                                                                                           11
```

May 02, 04 2:03	frmMain.cs	Page 171/186	May 02, 04 2:03	frmMain.cs	Page 172/186
// lblQuantizerOrig	inalMarker4		System	.Windows.Forms.ScrollBars.Horizontal;	
//			this.txtQuantizer0	riginal2.Size = new System.Drawing.Size	(328, 48);
this.lblQuantizerOr	iginalMarker4.BackColor =		this.txtQuantizer0	riginal2.Tablndex = 44;	
this lblouantizerOr	jginalMarker4 Enabled - false.		this tytOuantizerO	riginal2.labStop = laise;	
this.lblQuantizerOr	iginalMarker4.Location = new		//		
System.	Drawing.Point(512, 200);		// lblQuantizerOri	ginalMarker2	
this.lblQuantizerOr	iginalMarker4.Name = "lblQuantizer0:	riginalMarker4";	11	5	
this.lblQuantizerOr	iginalMarker4.Size = new		this.lblQuantizer0	riginalMarker2.BackColor =	
System.	Drawing.Size(32, 16);		System	.Drawing.SystemColors.Window;	
this.lblQuantizerOr	iginalMarker4.Tablndex = 49;		this.lblQuantizer0	riginalMarker2.Enabled = false;	
System.	Drawing.ContentAlignment.TopCenter:		System	Drawing Point (512, 80):	
//			this.lblQuantizer0	riginalMarker2.Name = "lblQuantizerOrig	inalMarker2";
// lblQuantizerOrig	inal4		this.lblQuantizer0	riginalMarker2.Size = new	
//			System	.Drawing.Size(32, 16);	
this.lblQuantizerOr	iginal4.Location = new		this.lblQuantizer0	riginalMarker2.TabIndex = 43;	
this lblouantizerOr	iginal4 Name - "lbl0uantizerOrigina"	14".	CHIS.IDIQUANCIZEIO	Drawing Contentalignment TonCenter:	
this.lblOuantizerOr	iginal4.Size = new System.Drawing.Si	ize(72, 16);	//	.brawing.concentrigiment.ropeenter,	
this.lblQuantizerOr	iginal4.TabIndex = 48;	, - , ,	// lblQuantizerOri	ginal2	
this.lblQuantizerOr	iginal4.Text = "Original 4:";		//		
			this.lblQuantizer0	riginal2.Location = new	
// txtQuantizer4			System	.Drawing.Point(448, 80);	
// this tytOuantizer4	Autosizo - falso.		this lblouantizer0	riginal2.Name = "IDIQUantIZerOriginal2"	; (72 16) ·
this.txtQuantizer4.	Location = new System.Drawing.Point	(552, 131);	this.lblQuantizer0	riginal2.TabIndex = 42:	(12, 10),
this.txtQuantizer4.1	Multiline = true;	(,, ,	this.lblQuantizer0	riginal2.Text = "Original 2:";	
this.txtQuantizer4.	Name = "txtQuantizer4";		//		
this.txtQuantizer4.	ScrollBars =		// txtQuantizer2		
System.	Windows.Forms.ScrollBars.Horizontal	19).	//	AutoSigo - folgo:	
this.txtQuantizer4.	TabIndex = 3:	40),	this.txtOuantizer2	.Location = new System.Drawing.Point(55	2. 11):
this.txtQuantizer4.	Text = "";		this.txtQuantizer2	.Multiline = true;	_,,
this.txtQuantizer4.	GotFocus += new		this.txtQuantizer2	.Name = "txtQuantizer2";	
System.	EventHandler(this.txtQuantizer4_Clic	ck);	this.txtQuantizer2	.ScrollBars =	
this.txtQuantizer4.	Click += new Eventuendler (this totOventicen( Cli	~!~ ) .	System	.Windows.Forms.ScrollBars.Horizontal;	\ <b>.</b>
//	Evenchandler (chils.cxcguancizer4_cii)	JK);	this tytOuantizer2	-312e = 1ew System.Drawing.Size(320, 40) TabIndex = 1.	) <i>i</i>
// lblQuantizerMark	er4		this.txtQuantizer2	.Text = "";	
//			this.txtQuantizer2	.GotFocus += new	
this.lblQuantizerMa	rker4.BackColor =		System	.EventHandler(this.txtQuantizer2_Click)	;
System.	Drawing.SystemColors.Window;		this.txtQuantizer2	.Click += new	
this lblOuantizerMa	rker4 Location = new		//	.Evenchandler (chils.txtQuantizerz_click)	,
System.	Drawing.Point(512, 128);		// lblOuantizerMar	ker2	
this.lblQuantizerMa	<pre>rker4.Name = "lblQuantizerMarker4";</pre>		11		
this.lblQuantizerMa	<pre>rker4.Size = new System.Drawing.Size</pre>	e(32, 16);	this.lblQuantizerM	arker2.BackColor =	
this.lblQuantizerMa	rker4.TabIndex = 46;		System	.Drawing.SystemColors.Window;	
this.ibiQuantizerMa	<pre>rker4.lextAllgn = Drawing ContentAlignment TonCenter.</pre>		this lblOuantizerM	arker2.Lnabled = Ialse;	
//	brawing.concentratignment.topeenter,		System	.Drawing.Point(512, 8):	
// lblQuantizer4			this.lblQuantizerM	arker2.Name = "lblQuantizerMarker2";	
11			this.lblQuantizerM	arker2.Size = new System.Drawing.Size(3	2, 16);
this.lblQuantizer4.	Location = new System.Drawing.Point	(448, 128);	this.lblQuantizerM	arker2.TabIndex = 40;	
this.lblQuantizer4.	Name = "IblQuantizer4";		this.lblQuantizerM	arker2.TextAlign =	
this lblOuantizer4	Size = new System, Drawing, Size(72, .)	10);	//	.Drawing.concentarignment.topcenter;	
this.lblQuantizer4.	Text = "Quantizer 4:";		// lblOuantizer2		
//			11		
// txtQuantizerOrig	inal2		this.lblQuantizer2	.Location = new System.Drawing.Point(44	8, 8);
	ininglo puteding 5 l		this.lblQuantizer2	.Name = "lblQuantizer2";	
this txtQuantizerOr	<pre>iginal2.AutoSize = false; iginal2 Enabled = false;</pre>		this.lblQuantizer2	.size = new System.Drawing.Size(/2, 16)	;
this tytouantizeror	iginal2.Location = new		this lblouantizer2	.Text = "Ouantizer 2.".	
System.	Drawing.Point(552, 67);		//		
this.txtQuantizerOr	iginal2.Multiline = true;		// txtQuantizerOri	ginal3	
this.txtQuantizerOr	iginal2.Name = "txtQuantizerOriginal	12";	//	-	
this.txtQuantizerOr	iginal2.ScrollBars =		this.txtQuantizer0	riginal3.AutoSize = false;	

May 02, 04 2:03	frmMain.cs	Page 173/186	May 02, 04 2:03	frmMain.cs	Page 174/186
this.txtQuantizerOr	riginal3.Enabled = false;		//		
this.txtQuantizerOr	riginal3.Location = new		// txtQuantizerOrig	jinall	
System.	.Drawing.Point(112, 187);		//	deviaell Autopies felees	
this.txtQuantizerOr	riginal3.Multiline = true;	no12".	this.txtQuantizerOr	riginali.Autosize = raise;	
this tytOuantizerOr	riginal3 ScrollBars -	illato ;	this tytOuantizerOr	riginall Location - new	
System.	Windows Forms ScrollBars Horizont	al:	System.	Drawing Point (112, 67):	
this.txtOuantizerOr	riginal3.Size = new System.Drawing	.Size(328, 48);	this.txtOuantizerOr	riginal1.Multiline = true;	
this.txtQuantizerOr	riginal3.TabIndex = 38;		this.txtQuantizerOr	iginal1.Name = "txtQuantizerOrigin	nall";
this.txtQuantizerOr	riginal3.TabStop = false;		this.txtQuantizerOr	iginal1.ScrollBars =	
this.txtQuantizerOr	riginal3.Text = "";		System.	Windows.Forms.ScrollBars.Horizonta	al;
//			this.txtQuantizerOr	iginall.Size = new System.Drawing.	.Size(328, 48);
// lblQuantizerOrig	ginalMarker3		this.txtQuantizerOr	riginall.TabIndex = 32;	
//	niginalMarker? PackColor -		this.txtQuantizerOr	riginali.labStop = raise;	
CHIIS.IDIQUALICIZEIOI	Drawing SystemColors Window:		//	iginali.lext – ,	
this.lblOuantizerOr	riginalMarker3.Enabled = false:		// lblOuantizerOric	vinalMarker1	
this.lblQuantizerOr	riginalMarker3.Location = new		//	,	
System.	.Drawing.Point(72, 200);		this.lblQuantizerOr	iginalMarker1.BackColor =	
this.lblQuantizerOr	riginalMarker3.Name = "lblQuantize	rOriginalMarker3";	System.	Drawing.SystemColors.Window;	
this.lblQuantizerOr	riginalMarker3.Size = new		this.lblQuantizerOr	iginalMarker1.Enabled = false;	
System.	.Drawing.Size(32, 16);		this.lblQuantizerOr	iginalMarkerl.Location = new	
this.lblQuantizerOr	riginalMarker3.Tabindex = 3/;		System.	Drawing.Point(/2, 80);	nomininal Mankan1".
CHIIS.IDIQUALICIZEIOI	Drawing ContentAlignment TonCente		this lblouantizerOr	iginalMarker1 Size - new	. OliginalMarkeri ;
//	.Drawing.concentratignment.topeente	· /	System.	Drawing.Size(32, 16):	
// lblQuantizerOric	ginal3		this.lblQuantizerOr	riginalMarker1.TabIndex = 31;	
//	-		this.lblQuantizerOr	iginalMarker1.TextAlign =	
this.lblQuantizerOr	riginal3.Location = new System.Dra	wing.Point(8, 200);	System.	Drawing.ContentAlignment.TopCenter	-;
this.lblQuantizerOr	riginal3.Name = "lblQuantizerOrigi	nal3";	//		
this.lblQuantizerOr	riginal3.Size = new System.Drawing	.Size(/2, 16);	// lblQuantizerOrig	jinall	
this lblouantizerOr	riginal3 Text = "Original 3.".		// this lblouantizerOr	riginall Location - new System Dray	ving Point (8 80).
//	iginais.iekt - originai 5.,		this.lblouantizerOr	iginall.Name = "lblOuantizerOrigin	nall":
// txtQuantizer3			this.lblQuantizerOr	iginall.Size = new System.Drawing.	.Size(72, 16);
11			this.lblQuantizerOr	riginal1.TabIndex = 30;	
this.txtQuantizer3.	.AutoSize = false;		this.lblQuantizerOr	iginal1.Text = "Original 1:";	
this.txtQuantizer3.	.Location = new System.Drawing.Poi	nt(112, 131);			
this txtQuantizer3.	.Multiline = true;		// txtQuantizeri		
this.txtQuantizer3.	ScrollBars =		this.txtOuantizer1.	AutoSize = false:	
System.	.Windows.Forms.ScrollBars.Horizont	al;	this.txtQuantizer1.	Location = new System.Drawing.Poir	nt(112, 11);
this.txtQuantizer3.	.Size = new System.Drawing.Size(32	8, 48);	this.txtQuantizer1.	Multiline = true;	
this.txtQuantizer3.	.TabIndex = 2;		this.txtQuantizer1.	Name = "txtQuantizer1";	
this.txtQuantizer3.	.Text = "";		this.txtQuantizer1.	ScrollBars =	_
this.txtQuantizer3.	.GotFocus += new		System.	Windows.Forms.ScrollBars.Horizonta	11;
system.	.EventHandler(this.txtQuantizer3_C	lick);	this.txtQuantizer1.	Size = new System.Drawing.Size(32)	3, 48);
Sustem	EventHandler(this tytOuantizer3 (	lick).	this tytOuantizer1	Text = "".	
//	. L'enenanaier (enib. excguanerzero_e	(iten)	this.txtOuantizer1.	GotFocus += new	
// lblQuantizerMark	ker3		System.	EventHandler(this.txtQuantizer1_CI	Lick);
//			this.txtQuantizer1.	Click += new	
this.lblQuantizerMa	arker3.BackColor =		System.	EventHandler(this.txtQuantizer1_C	Lick);
System.	.Drawing.SystemColors.Window;			1	
this.lblQuantizerMa	arker3.Enabled = Ialse;	ng Doint (72 129).	// ibiQuantizerMark	teri	
this lblOuantizerMa	arker3 Name = "lblOuantizerMarker3	".	// this lblOuantizerMa	erkerl BackColor =	
this.lblOuantizerMa	arker3.Size = new System.Drawing.S	ize(32, 16);	System.	Drawing.SystemColors.Window;	
this.lblQuantizerMa	arker3.TabIndex = 34;		this.lblQuantizerMa	arker1.Enabled = false;	
this.lblQuantizerMa	arker3.TextAlign =		this.lblQuantizerMa	arker1.Location = new System.Drawin	ng.Point(72, 8);
System.	.Drawing.ContentAlignment.TopCente	r;	this.lblQuantizerMa	arker1.Name = "lblQuantizerMarker1'	';
//			this.lblQuantizerMa	arker1.Size = new System.Drawing.Si	Lze(32, 16);
// 1blQuantizer3			this.lblQuantizerMa	arkeri.Tabindex = 28;	
// this lblouantizor?	Location - new System Drawing Poi	nt (8 128)	LIIIS.IDIQUANTIZETMA	Drawing Contentalignment TonContor	· ·
this.lblOuantizer3	Name = "lblouantizer3":		//	stawing.concentratignment.topcenter	. /
this.lblQuantizer3.	.Size = new System.Drawing.Size(72	, 16);	// lblQuantizer1		
this.lblQuantizer3.	.TabIndex = 33;		// ~		
this.lblQuantizer3.	.Text = "Quantizer 3:";		this.lblQuantizer1.	Location = new System.Drawing.Poir	it(8, 8);

```
frmMain.cs
                                                                                                                         frmMain.cs
May 02, 04 2:03
                                                                   Page 175/186
                                                                                       May 02, 04 2:03
                                                                                                                                                          Page 176/186
     this.lblQuantizer1.Name = "lblQuantizer1";
                                                                                                        System.Windows.Forms.ScrollBars.Horizontal;
     this.lblQuantizer1.Size = new System.Drawing.Size(72, 16);
                                                                                            this.txtOriginalEncodedData.Size = new System.Drawing.Size(864, 64);
     this.lblQuantizer1.TabIndex = 27;
                                                                                            this.txtOriginalEncodedData.TabIndex = 10;
     this.lblQuantizer1.Text = "Quantizer 1:";
                                                                                            this.txtOriginalEncodedData.TabStop = false;
                                                                                            this.txtOriginalEncodedData.Text = "";
     // tabEncodedData
                                                                                            this.toolTips.SetToolTip(this.txtOriginalEncodedData,
                                                                                                  "This is the original entropy encoded data stream.");
     11
     this.tabEncodedData.Controls.Add(this.lblOriginalHeader);
     this.tabEncodedData.Controls.Add(this.txtOriginalHeader);
                                                                                            // lblOriginalEncodedData
     this.tabEncodedData.Controls.Add(this.lblScanHeader);
                                                                                            11
     this.tabEncodedData.Controls.Add(this.txtScanHeader);
                                                                                            this.lblOriginalEncodedData.Location = new
     this.tabEncodedData.Controls.Add(this.txtOriginalEncodedData);
                                                                                                        System.Drawing.Point(8, 120);
     this.tabEncodedData.Controls.Add(this.lblOriginalEncodedData);
                                                                                            this.lblOriginalEncodedData.Name = "lblOriginalEncodedData";
     this.tabEncodedData.Controls.Add(this.txtEncodedData);
                                                                                            this.lblOriginalEncodedData.Size = new System.Drawing.Size(128, 16);
     this.tabEncodedData.Controls.Add(this.lblEncodedData);
                                                                                            this.lblOriginalEncodedData.TabIndex = 9;
     this.tabEncodedData.Location = new System.Drawing.Point(4, 22);
                                                                                            this.lblOriginalEncodedData.Text = "Original Encoded Data:";
    this.tabEncodedData.Name = "tabEncodedData";
this.tabEncodedData.Size = new System.Drawing.Size(888, 246);
                                                                                            11
                                                                                            // txtEncodedData
     this.tabEncodedData.TabIndex = 2;
                                                                                            11
     this.tabEncodedData.Text = "Encoded Data";
                                                                                            this.txtEncodedData.Location = new System.Drawing.Point(8, 32);
                                                                                            this.txtEncodedData.MaxLength = 10240;
    //
    // lblOriginalHeader
                                                                                            this.txtEncodedData.Multiline = true;
                                                                                            this.txtEncodedData.Name = "txtEncodedData";
     11
                                                                                            this.txtEncodedData.ScrollBars =
     this.lblOriginalHeader.Location = new System.Drawing.Point(312, 112);
     this.lblOriginalHeader.Name = "lblOriginalHeader";
                                                                                                        System.Windows.Forms.ScrollBars.Horizontal;
     this.lblOriginalHeader.Size = new System.Drawing.Size(88, 16);
                                                                                            this.txtEncodedData.Size = new System.Drawing.Size(864, 64);
     this.lblOriginalHeader.TabIndex = 14;
                                                                                            this.txtEncodedData.TabIndex = 0;
     this.lblOriginalHeader.Text = "Original Header:";
                                                                                            this.txtEncodedData.Text = "";
                                                                                            this.toolTips.SetToolTip(this.txtEncodedData,
     11
                                                                                                  "This is the entropy encoded data stream.");
     // txtOriginalHeader
     11
                                                                                            11
     this.txtOriginalHeader.Enabled = false;
                                                                                            // lblEncodedData
     this.txtOriginalHeader.Location = new System.Drawing.Point(408, 112);
                                                                                            11
     this.txtOriginalHeader.Name = "txtOriginalHeader";
                                                                                            this.lblEncodedData.Location = new System.Drawing.Point(8, 16);
     this.txtOriginalHeader.Size = new System.Drawing.Size(464, 20);
                                                                                           this.lblEncodedData.Name = "lblEncodedData";
this.lblEncodedData.Size = new System.Drawing.Size(248, 16);
     this.txtOriginalHeader.TabIndex = 13;
    this.txtOriginalHeader.TabStop = false;
this.txtOriginalHeader.Text = "";
                                                                                            this.lblEncodedData.TabIndex = 6;
                                                                                            this.lblEncodedData.Text = "Encoded Data:";
     this.toolTips.SetToolTip(this.txtOriginalHeader,
           "This is the original Scan Header for the encoded data.");
                                                                                            // tabApplicationData
                                                                                            11
     // lblScanHeader
                                                                                           this.tabApplicationData.Controls.Add(this.txtApplicationData10);
     11
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker10);
     this.lblScanHeader.Location = new System.Drawing.Point(320, 8);
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData10);
     this.lblScanHeader.Name = "lblScanHeader";
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData9);
     this.lblScanHeader.Size = new System.Drawing.Size(80, 16);
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker9);
    this.lblScanHeader.TabIndex = 12;
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData9);
     this.lblScanHeader.Text = "Scan Header:";
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData8);
     11
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker8);
     // txtScanHeader
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData8);
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData7);
     11
     this.txtScanHeader.Location = new System.Drawing.Point(408, 8);
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker7);
     this.txtScanHeader.Name = "txtScanHeader";
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData7);
     this.txtScanHeader.Size = new System.Drawing.Size(464, 20);
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData6);
     this.txtScanHeader.TabIndex = 1;
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker6);
     this.txtScanHeader.Text = "";
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData6);
     this.toolTips.SetToolTip(this.txtScanHeader,
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData5);
           "This is the Scan Header desrcibing this particular "+
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker5);
                   "encoded stream.");
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData5);
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData4);
     // txtOriginalEncodedData
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker4);
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData4);
     this.txtOriginalEncodedData.Enabled = false;
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData3);
     this.txtOriginalEncodedData.Location = new
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker3);
                 System.Drawing.Point(8, 136);
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData3);
     this.txtOriginalEncodedData.MaxLength = 10240;
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData2);
     this.txtOriginalEncodedData.Multiline = true;
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationMarker2);
     this.txtOriginalEncodedData.Name = "txtOriginalEncodedData";
                                                                                            this.tabApplicationData.Controls.Add(this.lblApplicationData2);
     this.txtOriginalEncodedData.ScrollBars =
                                                                                            this.tabApplicationData.Controls.Add(this.txtApplicationData1);
```

```
frmMain.cs
                                                                                                                         frmMain.cs
May 02, 04 2:03
                                                                   Page 177/186
                                                                                       May 02, 04 2:03
                                                                                                                                                           Page 178/186
     this.tabApplicationData.Controls.Add(this.lblApplicationMarker1);
     this.tabApplicationData.Controls.Add(this.lblApplicationData1);
                                                                                            this.lblApplicationData9.Location = new System.Drawing.Point(0, 208);
     this.tabApplicationData.Location = new System.Drawing.Point(4, 22);
                                                                                            this.lblApplicationData9.Name = "lblApplicationData9";
    this.tabApplicationData.Name = "tabApplicationData";
this.tabApplicationData.Size = new System.Drawing.Size(888, 246);
                                                                                            this.lblApplicationData9.Size = new System.Drawing.Size(64, 16);
                                                                                            this.lblApplicationData9.TabIndex = 60;
     this.tabApplicationData.TabIndex = 6;
                                                                                            this.lblApplicationData9.Text = "App Data 9:";
     this.tabApplicationData.Text = "Application Data";
                                                                                            11
                                                                                            // txtApplicationData8
     //
     // txtApplicationData10
                                                                                            11
     11
                                                                                            this.txtApplicationData8.AutoSize = false;
     this.txtApplicationData10.AutoSize = false;
                                                                                            this.txtApplicationData8.Location = new
     this.txtApplicationData10.Location = new
                                                                                                        System.Drawing.Point(552, 152);
                 System.Drawing.Point(552, 200);
                                                                                            this.txtApplicationData8.Multiline = true;
     this.txtApplicationData10.Multiline = true;
                                                                                            this.txtApplicationData8.Name = "txtApplicationData8";
     this.txtApplicationData10.Name = "txtApplicationData10";
                                                                                            this.txtApplicationData8.ScrollBars =
                                                                                                        System.Windows.Forms.ScrollBars.Horizontal;
     this.txtApplicationData10.ScrollBars =
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                            this.txtApplicationData8.Size = new System.Drawing.Size(328, 37);
                                                                                            this.txtApplicationData8.TabIndex = 7;
     this.txtApplicationData10.Size = new System.Drawing.Size(328, 37);
                                                                                            this.txtApplicationData8.Text = "";
     this.txtApplicationData10.TabIndex = 9;
     this.txtApplicationData10.Text = "";
                                                                                            11
                                                                                            // lblApplicationMarker8
    //
     // lblApplicationMarker10
                                                                                            11
     11
                                                                                            this.lblApplicationMarker8.BackColor =
     this.lblApplicationMarker10.BackColor =
                                                                                                        System.Drawing.SystemColors.Window;
                 System.Drawing.SystemColors.Window;
                                                                                            this.lblApplicationMarker8.Enabled = false;
     this.lblApplicationMarker10.Enabled = false;
                                                                                            this.lblApplicationMarker8.Location = new
     this.lblApplicationMarker10.Location = new
                                                                                                        System.Drawing.Point(512, 160);
                 System.Drawing.Point(512, 208);
                                                                                            this.lblApplicationMarker8.Name = "lblApplicationMarker8";
    this.lblApplicationMarker10.Name = "lblApplicationMarker10";
this.lblApplicationMarker10.Size = new System.Drawing.Size(32, 16);
                                                                                            this.lblApplicationMarker8.Size = new System.Drawing.Size(32, 16);
                                                                                            this.lblApplicationMarker8.TabIndex = 58;
     this.lblApplicationMarker10.TabIndex = 64;
                                                                                            this.lblApplicationMarker8.TextAlign =
     this.lblApplicationMarker10.TextAlign =
                                                                                                        System.Drawing.ContentAlignment.TopCenter;
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                            // lblApplicationData8
     // lblApplicationData10
                                                                                            11
     11
                                                                                            this.lblApplicationData8.Location = new System.Drawing.Point(448, 160);
     this.lblApplicationData10.Location = new
                                                                                            this.lblApplicationData8.Name = "lblApplicationData8";
                 System.Drawing.Point(440, 208);
                                                                                            this.lblApplicationData8.Size = new System.Drawing.Size(64, 16);
     this.lblApplicationData10.Name = "lblApplicationData10";
                                                                                            this.lblApplicationData8.TabIndex = 57;
     this.lblApplicationData10.Size = new System.Drawing.Size(72, 16);
                                                                                            this.lblApplicationData8.Text = "App Data 8:";
     this.lblApplicationData10.TabIndex = 63;
     this.lblApplicationData10.Text = "App Data 10:";
                                                                                            // txtApplicationData7
                                                                                            11
     11
     // txtApplicationData9
                                                                                            this.txtApplicationData7.AutoSize = false;
                                                                                            this.txtApplicationData7.Location = new System.Drawing.Point(104, 152);
     11
     this.txtApplicationData9.AutoSize = false;
                                                                                            this.txtApplicationData7.Multiline = true;
     this.txtApplicationData9.Location = new System.Drawing.Point(104, 200);
                                                                                            this.txtApplicationData7.Name = "txtApplicationData7";
     this.txtApplicationData9.Multiline = true;
                                                                                            this.txtApplicationData7.ScrollBars =
     this.txtApplicationData9.Name = "txtApplicationData9";
                                                                                                        System.Windows.Forms.ScrollBars.Horizontal;
     this.txtApplicationData9.ScrollBars =
                                                                                            this.txtApplicationData7.Size = new System.Drawing.Size(328, 37);
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                            this.txtApplicationData7.TabIndex = 6;
     this.txtApplicationData9.Size = new System.Drawing.Size(328, 37);
                                                                                            this.txtApplicationData7.Text = "";
     this.txtApplicationData9.TabIndex = 8;
                                                                                            11
     this.txtApplicationData9.Text = "";
                                                                                            // lblApplicationMarker7
                                                                                            11
    11
     // lblApplicationMarker9
                                                                                            this.lblApplicationMarker7.BackColor =
                                                                                                        System.Drawing.SystemColors.Window;
     11
     this.lblApplicationMarker9.BackColor =
                                                                                            this.lblApplicationMarker7.Enabled = false;
                 System.Drawing.SystemColors.Window;
                                                                                            this.lblApplicationMarker7.Location = new
     this.lblApplicationMarker9.Enabled = false;
                                                                                                        System.Drawing.Point(64, 160);
     this.lblApplicationMarker9.Location = new
                                                                                            this.lblApplicationMarker7.Name = "lblApplicationMarker7";
                 System.Drawing.Point(64, 208);
                                                                                            this.lblApplicationMarker7.Size = new System.Drawing.Size(32, 16);
     this.lblApplicationMarker9.Name = "lblApplicationMarker9";
                                                                                            this.lblApplicationMarker7.TabIndex = 55;
     this.lblApplicationMarker9.Size = new System.Drawing.Size(32, 16);
                                                                                            this.lblApplicationMarker7.TextAlign =
     this.lblApplicationMarker9.TabIndex = 61;
                                                                                                        System.Drawing.ContentAlignment.TopCenter;
     this.lblApplicationMarker9.TextAlign =
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                            // lblApplicationData7
                                                                                            11
     // lblApplicationData9
                                                                                            this.lblApplicationData7.Location = new System.Drawing.Point(0, 160);
```

```
frmMain.cs
                                                                                                                         frmMain.cs
May 02, 04 2:03
                                                                   Page 179/186
                                                                                      May 02, 04 2:03
                                                                                                                                                          Page 180/186
     this.lblApplicationData7.Name = "lblApplicationData7";
                                                                                           this.lblApplicationData5.Text = "App Data 5:";
    this.lblApplicationData7.Size = new System.Drawing.Size(64, 16);
    this.lblApplicationData7.TabIndex = 54;
                                                                                           // txtApplicationData4
    this.lblApplicationData7.Text = "App Data 7:";
                                                                                           11
                                                                                           this.txtApplicationData4.AutoSize = false;
                                                                                           this.txtApplicationData4.Location = new System.Drawing.Point(552, 56);
    // txtApplicationData6
                                                                                           this.txtApplicationData4.Multiline = true;
    11
    this.txtApplicationData6.AutoSize = false;
                                                                                           this.txtApplicationData4.Name = "txtApplicationData4";
                                                                                           this.txtApplicationData4.ScrollBars =
    this.txtApplicationData6.Location = new System.Drawing.Point(552, 105);
                                                                                                        System.Windows.Forms.ScrollBars.Horizontal;
    this.txtApplicationData6.Multiline = true;
    this.txtApplicationData6.Name = "txtApplicationData6";
                                                                                           this.txtApplicationData4.Size = new System.Drawing.Size(328, 37);
    this.txtApplicationData6.ScrollBars =
                                                                                           this.txtApplicationData4.TabIndex = 3;
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                           this.txtApplicationData4.Text = "";
    this.txtApplicationData6.Size = new System.Drawing.Size(328, 37);
    this.txtApplicationData6.TabIndex = 5;
                                                                                           // lblApplicationMarker4
    this.txtApplicationData6.Text = "";
                                                                                           11
                                                                                           this.lblApplicationMarker4.BackColor =
    // lblApplicationMarker6
                                                                                                        System.Drawing.SystemColors.Window;
    11
                                                                                           this.lblApplicationMarker4.Enabled = false;
    this.lblApplicationMarker6.BackColor =
                                                                                           this.lblApplicationMarker4.Location = new System.Drawing.Point(512, 64);
                                                                                           this.lblApplicationMarker4.Name = "lblApplicationMarker4";
                 System.Drawing.SystemColors.Window;
    this.lblApplicationMarker6.Enabled = false;
                                                                                           this.lblApplicationMarker4.Size = new System.Drawing.Size(32, 16);
    this.lblApplicationMarker6.Location = new
        System.Drawing.Point(512, 112);
                                                                                           this.lblApplicationMarker4.TabIndex = 46;
                                                                                           this.lblApplicationMarker4.TextAlign =
    this.lblApplicationMarker6.Name = "lblApplicationMarker6";
                                                                                                        System.Drawing.ContentAlignment.TopCenter;
    this.lblApplicationMarker6.Size = new System.Drawing.Size(32, 16);
                                                                                           // lblApplicationData4
    this.lblApplicationMarker6.TabIndex = 52;
    this.lblApplicationMarker6.TextAlign =
                                                                                           11
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                           this.lblApplicationData4.Location = new System.Drawing.Point(448, 64);
                                                                                           this.lblApplicationData4.Name = "lblApplicationData4";
this.lblApplicationData4.Size = new System.Drawing.Size(64, 16);
    // lblApplicationData6
                                                                                           this.lblApplicationData4.TabIndex = 45;
    11
    this.lblApplicationData6.Location = new System.Drawing.Point(448, 112);
                                                                                           this.lblApplicationData4.Text = "App Data 4:";
    this.lblApplicationData6.Name = "lblApplicationData6";
                                                                                           11
    this.lblApplicationData6.Size = new System.Drawing.Size(64, 16);
                                                                                           // txtApplicationData3
    this.lblApplicationData6.TabIndex = 51;
                                                                                           11
    this.lblApplicationData6.Text = "App Data 6:";
                                                                                           this.txtApplicationData3.AutoSize = false;
                                                                                           this.txtApplicationData3.Location = new System.Drawing.Point(104, 56);
    11
    // txtApplicationData5
                                                                                           this.txtApplicationData3.Multiline = true;
                                                                                           this.txtApplicationData3.Name = "txtApplicationData3";
    this.txtApplicationData5.AutoSize = false;
                                                                                           this.txtApplicationData3.ScrollBars =
    this.txtApplicationData5.Location = new System.Drawing.Point(104, 105);
                                                                                                        System.Windows.Forms.ScrollBars.Horizontal;
    this.txtApplicationData5.Multiline = true;
                                                                                           this.txtApplicationData3.Size = new System.Drawing.Size(328, 37);
    this.txtApplicationData5.Name = "txtApplicationData5";
                                                                                           this.txtApplicationData3.TabIndex = 2;
    this.txtApplicationData5.ScrollBars =
                                                                                           this.txtApplicationData3.Text = "";
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                           11
    this.txtApplicationData5.Size = new System.Drawing.Size(328, 37);
                                                                                           // lblApplicationMarker3
    this.txtApplicationData5.TabIndex = 4;
                                                                                           11
    this.txtApplicationData5.Text = "";
                                                                                           this.lblApplicationMarker3.BackColor =
                                                                                                        System.Drawing.SystemColors.Window;
    11
    // lblApplicationMarker5
                                                                                           this.lblApplicationMarker3.Enabled = false;
                                                                                           this.lblApplicationMarker3.Location = new System.Drawing.Point(64, 64);
    1
    this.lblApplicationMarker5.BackColor =
                                                                                           this.lblApplicationMarker3.Name = "lblApplicationMarker3";
                                                                                           this.lblApplicationMarker3.Size = new System.Drawing.Size(32, 16);
                 System.Drawing.SystemColors.Window;
    this.lblApplicationMarker5.Enabled = false;
                                                                                           this.lblApplicationMarker3.TabIndex = 43;
    this.lblApplicationMarker5.Location = new
                                                                                           this.lblApplicationMarker3.TextAlign =
                 System.Drawing.Point(64, 112);
                                                                                                        System.Drawing.ContentAlignment.TopCenter;
    this.lblApplicationMarker5.Name = "lblApplicationMarker5";
    this.lblApplicationMarker5.Size = new System.Drawing.Size(32, 16);
                                                                                           // lblApplicationData3
    this.lblApplicationMarker5.TabIndex = 49;
    this.lblApplicationMarker5.TextAlign =
                                                                                           this.lblApplicationData3.Location = new System.Drawing.Point(0, 64);
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                           this.lblApplicationData3.Name = "lblApplicationData3";
                                                                                           this.lblApplicationData3.Size = new System.Drawing.Size(64, 16);
    // lblApplicationData5
                                                                                           this.lblApplicationData3.TabIndex = 42;
                                                                                           this.lblApplicationData3.Text = "App Data 3:";
    this.lblApplicationData5.Location = new System.Drawing.Point(0, 112);
    this.lblApplicationData5.Name = "lblApplicationData5";
                                                                                           // txtApplicationData2
    this.lblApplicationData5.Size = new System.Drawing.Size(64, 16);
                                                                                           11
     this.lblApplicationData5.TabIndex = 48;
                                                                                           this.txtApplicationData2.AutoSize = false;
```

```
frmMain.cs
                                                                                                                           frmMain.cs
May 02, 04 2:03
                                                                    Page 181/186
                                                                                        May 02, 04 2:03
                                                                                                                                                             Page 182/186
     this.txtApplicationData2.Location = new System.Drawing.Point(552, 11);
                                                                                             this.tabMisc.Controls.Add(this.lblHierarchial);
     this.txtApplicationData2.Multiline = true;
                                                                                             this.tabMisc.Controls.Add(this.txtRestartMod8);
     this.txtApplicationData2.Name = "txtApplicationData2";
                                                                                             this.tabMisc.Controls.Add(this.lblRestartMod8);
     this.txtApplicationData2.ScrollBars =
                                                                                             this.tabMisc.Controls.Add(this.txtError);
                                                                                             this.tabMisc.Controls.Add(this.lblError);
                 System.Windows.Forms.ScrollBars.Horizontal;
     this.txtApplicationData2.Size = new System.Drawing.Size(328, 37);
                                                                                             this.tabMisc.Controls.Add(this.lblNumberLinesMarker);
     this.txtApplicationData2.TabIndex = 1;
                                                                                             this.tabMisc.Controls.Add(this.lblRestartMarker);
                                                                                             this.tabMisc.Controls.Add(this.txtNumberLines);
     this.txtApplicationData2.Text = "";
                                                                                             this.tabMisc.Controls.Add(this.lblNumberLines);
     11
     // lblApplicationMarker2
                                                                                             this.tabMisc.Controls.Add(this.txtRestart);
                                                                                             this.tabMisc.Controls.Add(this.lblRestart);
     11
     this.lblApplicationMarker2.BackColor =
                                                                                             this.tabMisc.Location = new System.Drawing.Point(4, 22);
                                                                                             this.tabMisc.Name = "tabMisc";
                 System.Drawing.SystemColors.Window;
     this.lblApplicationMarker2.Enabled = false;
                                                                                             this.tabMisc.Size = new System.Drawing.Size(888, 246);
     this.lblApplicationMarker2.Location = new
                                                                                             this.tabMisc.TabIndex = 4;
                 System.Drawing.Point(512, 16);
                                                                                             this.tabMisc.Text = "Misc";
    this.lblApplicationMarker2.Name = "lblApplicationMarker2";
this.lblApplicationMarker2.Size = new System.Drawing.Size(32, 16);
                                                                                             11
                                                                                             // lblExpandMarker
     this.lblApplicationMarker2.TabIndex = 40;
                                                                                             11
     this.lblApplicationMarker2.TextAlign =
                                                                                             this.lblExpandMarker.BackColor = System.Drawing.SystemColors.Window;
                                                                                             this.lblExpandMarker.Enabled = false;
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                             this.lblExpandMarker.Location = new System.Drawing.Point(112, 80);
this.lblExpandMarker.Name = "lblExpandMarker";
this.lblExpandMarker.Size = new System.Drawing.Size(32, 16);
     11
     // lblApplicationData2
     11
     this.lblApplicationData2.Location = new System.Drawing.Point(448, 16);
                                                                                             this.lblExpandMarker.TabIndex = 3\overline{4};
     this.lblApplicationData2.Name = "lblApplicationData2";
                                                                                             this.lblExpandMarker.TextAlign =
                                                                                                          System.Drawing.ContentAlignment.TopCenter;
     this.lblApplicationData2.Size = new System.Drawing.Size(64, 16);
     this.lblApplicationData2.TabIndex = 39;
     this.lblApplicationData2.Text = "App Data 2:";
                                                                                             // txtExpand
                                                                                             11
     // txtApplicationData1
                                                                                             this.txtExpand.Location = new System.Drawing.Point(152, 80);
                                                                                             this.txtExpand.Name = "txtExpand";
     11
                                                                                             this.txtExpand.Size = new System.Drawing.Size(208, 20);
     this.txtApplicationData1.AutoSize = false;
     this.txtApplicationDatal.Location = new System.Drawing.Point(104, 11);
                                                                                             this.txtExpand.TabIndex = 32;
     this.txtApplicationData1.Multiline = true;
                                                                                             this.txtExpand.Text = "";
     this.txtApplicationData1.Name = "txtApplicationData1";
                                                                                             11
     this.txtApplicationData1.ScrollBars =
                                                                                             // lblExpand
                 System.Windows.Forms.ScrollBars.Horizontal;
                                                                                             11
     this.txtApplicationData1.Size = new System.Drawing.Size(328, 37);
                                                                                             this.lblExpand.Location = new System.Drawing.Point(16, 80);
     this.txtApplicationData1.TabIndex = 0;
                                                                                             this.lblExpand.Name = "lblExpand";
     this.txtApplicationData1.Text = "";
                                                                                             this.lblExpand.Size = new System.Drawing.Size(96, 16);
                                                                                             this.lblExpand.TabIndex = 33;
     // lblApplicationMarker1
                                                                                             this.lblExpand.Text = "Expand Image";
     11
                                                                                             11
     this.lblApplicationMarker1.BackColor =
                                                                                             // txtHierarchial
                 System.Drawing.SystemColors.Window;
                                                                                             11
     this.lblApplicationMarker1.Enabled = false;
                                                                                             this.txtHierarchial.AutoSize = false;
     this.lblApplicationMarker1.Location = new System.Drawing.Point(64, 16);
                                                                                             this.txtHierarchial.Location = new System.Drawing.Point(416, 64);
     this.lblApplicationMarker1.Name = "lblApplicationMarker1";
                                                                                             this.txtHierarchial.Multiline = true;
     this.lblApplicationMarker1.Size = new System.Drawing.Size(32, 16);
                                                                                             this.txtHierarchial.Name = "txtHierarchial";
     this.lblApplicationMarker1.TabIndex = 28;
                                                                                             this.txtHierarchial.ScrollBars =
     this.lblApplicationMarker1.TextAlign =
                                                                                                          System.Windows.Forms.ScrollBars.Horizontal;
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                             this.txtHierarchial.Size = new System.Drawing.Size(464, 56);
                                                                                             this.txtHierarchial.TabIndex = 29;
     // lblApplicationData1
                                                                                             this.txtHierarchial.Text = "";
     11
                                                                                             11
     this.lblApplicationDatal.Location = new System.Drawing.Point(0, 16);
                                                                                             // lblHierarchialMarker
     this.lblApplicationData1.Name = "lblApplicationData1";
                                                                                             11
     this.lblApplicationData1.Size = new System.Drawing.Size(64, 16);
                                                                                             this.lblHierarchialMarker.BackColor =
     this.lblApplicationData1.TabIndex = 27;
                                                                                                          System.Drawing.SystemColors.Window;
     this.lblApplicationData1.Text = "App Data 1:";
                                                                                             this.lblHierarchialMarker.Enabled = false;
    11
                                                                                             this.lblHierarchialMarker.Location = new System.Drawing.Point(552, 40);
     // tabMisc
                                                                                             this.lblHierarchialMarker.Name = "lblHierarchialMarker";
                                                                                             this.lblHierarchialMarker.Size = new System.Drawing.Size(32, 16);
     11
     this.tabMisc.Controls.Add(this.lblExpandMarker);
                                                                                             this.lblHierarchialMarker.TabIndex = 3\hat{1};
     this.tabMisc.Controls.Add(this.txtExpand);
                                                                                             this.lblHierarchialMarker.TextAlign =
     this.tabMisc.Controls.Add(this.lblExpand);
                                                                                                          System.Drawing.ContentAlignment.TopCenter;
     this.tabMisc.Controls.Add(this.txtHierarchial);
                                                                                             // lblHierarchial
     this.tabMisc.Controls.Add(this.lblHierarchialMarker);
```

```
frmMain.cs
                                                                                                                           frmMain.cs
May 02, 04 2:03
                                                                    Page 183/186
                                                                                        May 02, 04 2:03
                                                                                                                                                             Page 184/186
                                                                                             this.txtNumberLines.TabIndex = 1;
     this.lblHierarchial.Location = new System.Drawing.Point(416, 40);
                                                                                             this.txtNumberLines.Text = "";
    this.lblHierarchial.Name = "lblHierarchial";
                                                                                             11
    this.lblHierarchial.Size = new System.Drawing.Size(128, 16);
this.lblHierarchial.TabIndex = 30;
                                                                                             // lblNumberLines
                                                                                             11
    this.lblHierarchial.Text = "Hierarchial Progression:";
                                                                                             this.lblNumberLines.Location = new System.Drawing.Point(16, 48);
                                                                                             this.lblNumberLines.Name = "lblNumberLines";
    11
     // txtRestartMod8
                                                                                             this.lblNumberLines.Size = new System.Drawing.Size(96, 16);
                                                                                             this.lblNumberLines.TabIndex = 2;
     11
     this.txtRestartMod8.Location = new System.Drawing.Point(624, 16);
                                                                                             this.lblNumberLines.Text = "Number of Lines:";
    this.txtRestartMod8.Name = "txtRestartMod8";
this.txtRestartMod8.Size = new System.Drawing.Size(72, 20);
                                                                                             11
                                                                                             // txtRestart
     this.txtRestartMod8.TabIndex = 8;
                                                                                             11
     this.txtRestartMod8.Text = "";
                                                                                             this.txtRestart.Location = new System.Drawing.Point(152, 16);
                                                                                             this.txtRestart.Name = "txtRestart";
     11
     // lblRestartMod8
                                                                                             this.txtRestart.Size = new System.Drawing.Size(208, 20);
     11
                                                                                             this.txtRestart.TabIndex = 0;
    this.lblRestartMod8.Location = new System.Drawing.Point(416, 16);
                                                                                             this.txtRestart.Text = "";
     this.lblRestartMod8.Name = "lblRestartMod8";
                                                                                             11
     this.lblRestartMod8.Size = new System.Drawing.Size(208, 16);
                                                                                             // lblRestart
     this.lblRestartMod8.TabIndex = 7;
                                                                                             11
     this.lblRestartMod8.Text = "Restart Modulo 8 occured at byte index:";
                                                                                             this.lblRestart.Location = new System.Drawing.Point(16, 16);
                                                                                             this.lblRestart.Name = "lblRestart";
this.lblRestart.Size = new System.Drawing.Size(96, 16);
     11
    // txtError
     11
                                                                                             this.lblRestart.TabIndex = 0;
     this.txtError.Location = new System.Drawing.Point(8, 128);
                                                                                             this.lblRestart.Text = "Restart Interval:";
    this.txtError.Multiline = true;
                                                                                             11
     this.txtError.Name = "txtError";
                                                                                             // picManipulatedSmall
     this.txtError.ScrollBars = System.Windows.Forms.ScrollBars.Horizontal;
                                                                                             11
     this.txtError.Size = new System.Drawing.Size(872, 112);
                                                                                             this.picManipulatedSmall.BackColor =
     this.txtError.TabIndex = 2;
                                                                                                         System.Drawing.SystemColors.Window;
     this.txtError.Text = "";
                                                                                             this.picManipulatedSmall.Location = new System.Drawing.Point(456, 8);
                                                                                             this.picManipulatedSmall.Name = "picManipulatedSmall";
    11
    // lblError
                                                                                             this.picManipulatedSmall.Size = new System.Drawing.Size(432, 344);
                                                                                             this.picManipulatedSmall.TabIndex = 1;
this.picManipulatedSmall.TabStop = false;
     11
     this.lblError.Location = new System.Drawing.Point(16, 104);
    this.lblError.Name = "lblError";
                                                                                             this.toolTips.SetToolTip(this.picManipulatedSmall,
     this.lblError.Size = new System.Drawing.Size(96, 16);
                                                                                                    "Manipulated Picture");
     this.lblError.TabIndex = 6;
     this.lblError.Text = "Program Errors:";
                                                                                             // picOriginalSmall
     11
     // lblNumberLinesMarker
                                                                                             this.picOriginalSmall.BackColor = System.Drawing.SystemColors.Window;
     11
                                                                                             this.picOriginalSmall.Location = new System.Drawing.Point(8, 8);
     this.lblNumberLinesMarker.BackColor =
                                                                                             this.picOriginalSmall.Name = "picOriginalSmall";
                 System.Drawing.SystemColors.Window;
                                                                                             this.picOriginalSmall.Size = new System.Drawing.Size(432, 344);
     this.lblNumberLinesMarker.Enabled = false;
                                                                                             this.picOriginalSmall.TabIndex = 0;
    this.lblNumberLinesMarker.Location = new System.Drawing.Point(112, 48);
                                                                                             this.picOriginalSmall.TabStop = false;
     this.lblNumberLinesMarker.Name = "lblNumberLinesMarker";
                                                                                             this.toolTips.SetToolTip(this.picOriginalSmall, "Orignal Picture");
     this.lblNumberLinesMarker.Size = new System.Drawing.Size(32, 16);
                                                                                             11
     this.lblNumberLinesMarker.TabIndex = 5;
                                                                                             // tabOriginal
     this.lblNumberLinesMarker.TextAlign =
                                                                                             11
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                             this.tabOriginal.Controls.Add(this.picOriginal);
                                                                                             this.tabOriginal.Location = new System.Drawing.Point(4, 22);
     // lblRestartMarker
                                                                                             this.tabOriginal.Name = "tabOriginal";
     11
                                                                                             this.tabOriginal.Size = new System.Drawing.Size(896, 627);
     this.lblRestartMarker.BackColor = System.Drawing.SystemColors.Window;
                                                                                             this.tabOriginal.TabIndex = 1;
     this.lblRestartMarker.Enabled = false;
                                                                                             this.tabOriginal.Text = "Original Picture";
     this.lblRestartMarker.Location = new System.Drawing.Point(112, 16);
                                                                                             11
    this.lblRestartMarker.Name = "lblRestartMarker";
                                                                                             // picOriginal
    this.lblRestartMarker.Size = new System.Drawing.Size(32, 16);
     this.lblRestartMarker.TabIndex = 4;
                                                                                             this.picOriginal.BackColor = System.Drawing.SystemColors.Window;
     this.lblRestartMarker.TextAlign =
                                                                                             this.picOriginal.Dock = System.Windows.Forms.DockStyle.Fill;
                 System.Drawing.ContentAlignment.TopCenter;
                                                                                             this.picOriginal.Location = new System.Drawing.Point(0, 0);
                                                                                             this.picOriginal.Name = "picOriginal";
     // txtNumberLines
                                                                                             this.picOriginal.Size = new System.Drawing.Size(896, 627);
     11
                                                                                             this.picOriginal.TabIndex = 0;
    this.txtNumberLines.Location = new System.Drawing.Point(152, 48);
                                                                                             this.picOriginal.TabStop = false;
     this.txtNumberLines.Name = "txtNumberLines";
                                                                                             11
     this.txtNumberLines.Size = new System.Drawing.Size(208, 20);
                                                                                             // tabManipulated
```

May 02, 04 2:03	frmMain.cs	Page 185/186	May 02, 04 2:03	frmMain.cs	Page 186/186
<pre>May 02, 04 2:03 // this.tabManipulated this.tabManipulated this.tabManipulated this.tabManipulated this.tabManipulated this.tabManipulated // // picManipulated // this.picManipulated // // openFileDialog // this.openFileDialod this.openFileDialod // // saveFileDialod</pre>	<pre>frmMain.cs d.Controls.Add(this.picManipulated d.Location = new System.Drawing.Pc d.Name = "tabManipulated"; d.Size = new System.Drawing.Size(8 d.TabIndex = 2; d.Text = "Manipulated Picture"; d.BackColor = System.Drawing.Syste d.Dock = System.Windows.Forms.Dock d.Location = new System.Drawing.Pc d.Mame = "picManipulated"; d.Size = new System.Drawing.Size(8 d.TabIndex = 1; d.TabStop = false; g.Filter = *.*) *.* JPEG files (*.jpeg)" + G files (*.jpg) *.jpg"; g.FilterIndex = 3; g.Title = "Open JPEG File";</pre>	<pre>Page 185/186 ); iint(4, 22); 96, 627); mColors.Window; Style.Fill; int(0, 0); 96, 627);</pre>	<pre>May 02, 04 2:03 this.tabMisc.Resu this.tabOriginal. this.tabManipulat this.ResumeLayout } #endregion /// <summary> /// Pre-conditions: /// Post-conditions /// The Windows /// Parameters: /// Return values: /// Function ret /// Description: /// This functio /// application. /// </summary> [STAThread] static void Main() {     Application.Run(n } </pre>	<pre>frmMain.cs meLayout(false); ResumeLayout(false); ed.ResumeLayout(false); ed(false);  None.  Form has been invoked. Form has been inv</pre>	lows based .NET on.Run form of the
<pre>// SaveFileDialogi // this.saveFileDialog     "All files (' this.saveFileDialog this.saveFileDialog // this.openFileDialog this.openFileDialog this.openFileDialog this.openFileDialog this.timerSplash // this.timerSplash.En this.timerSplash.e</pre>	<pre>gl.Filter = *.*) *.* Project files (*.SEP) *.S gl.FilterIndex = 2; gl.Title = "Save SEP File"; gl.Filter = *.*) *.* Project files (*.SEP) *.S gl.FilterIndex = 2; gl.Title = "Open SEP File"; nabled = true; ick += new EventHandler(this timerSplash Tick </pre>	EP"; EP";	; #endregion Standard } }	Windows From Application Methods	
<pre>// frmMain // this.AutoScaleBase! this.ClientSize = n this.Controls.Add(' this.Icon = ((Syste' this.Manu = this.met this.Name = "frmMa: this.StartPosition this.Text = "ISE JI this.tabMain.Resumet this.tabEonsol.Resut this.tabFile.Resumet this.tabHeaders.Resut this.tabHuffmanl.Ret this.tabHuffmanl.Ret this.tabAuffmanl.Ret this.tabAuffmanl.</pre>	<pre>Size = new System.Drawing.Size(5, new System.Drawing.Size(904, 653); this.tabMain); em.Drawing.Icon)(resources.GetObje enuFrmMain; in"; = .Windows.Forms.FormStartPosition.C PEG Manipulator"; ystem.EventHandler(this.frmMain_Lc eLayout(false); .ResumeLayout(false); sumeLayout(false); esumeLayout(false); esumeLayout(false); esumeLayout(false); esumeLayout(false); a.ResumeLayout(false); a.ResumeLayout(false); a.ResumeLayout(false); a.ResumeLayout(false); a.ResumeLayout(false); a.ResumeLayout(false); a.ResumeLayout(false);</pre>	13); ect("\$this.Icon"))); enterScreen; ad);			

Ма	y 02, 04 2:04	frmSplash.cs	Page 1/2	Ma	y 02, 04 2:04	frmSplash.cs	Page 2/2
///- /// ///	File Name:	frmSplash.cs			components.D: } } base.Dispose( dis	ispose(); sposing );	
	File Description:	JPEG Manipulator application.			} #region Windows Fo	rm Designer generated code	
                   	Project Name:	Selective Encryption for JPEG Images CSCI 4308-4318: Senior Project August 2003 to May 2004 Department of Computer Science University of Colorado at Boulder			/// <summary> /// Required method /// the contents of /// </summary> private void Initia {	d for Designer support - do not modify E this method with the code editor. alizeComponent()	
             	Project Sponsor:	Tom Lookabaugh Assistant Professor of Computer Science University of Colorado at Boulder			System.Resources System.Resource // // frmSplash	ResourceManager resources = new es.ResourceManager(typeof(frmSplash));	
             	Project Manager:	Bruce Sanders Assistant Professor of Computer Science University of Colorado at Boulder			// this.AutoScaleBas this.BackgroundIr (resources.Get(	<pre>seSize = new System.Drawing.Size(5, 13); mage = ((System.Drawing.Image) bbject("\$this.BackgroundImage")));</pre>	
/// /// /// /// ///	Team ISE Members:	Shinya Daigaku Geoffrey Griffith Joe Jarchow Joseph Kadhim Andrew Pouzeshi			<pre>this.ClientSize = this.FormBorderSt this.Icon = ((Sys (resources.GetC this.Name = "frm this.StartPositic System.Windows</pre>	<pre>= new System.Drawing.Size(512, 280); cyle = System.Windows.Forms.FormBorderStyle.None stem.Drawing.Icon) Object("\$this.Icon"))); Splash"; on = .Forms.FormStartPosition.CenterScreen;</pre>	;
/// /// /// /// /// ///	This code is open The authors are i from the usage of no warranties, pr support. The aut for good purposes	source and may be used with no cost. n no way responsible for any effects this code. It is provided as is with otections, promises or any form of hors would hope it would only be used . Thank you.		}	<pre>this.Text = "frm: this.TopMost = tr } #endregion</pre>	Splash"; rue;	
usin usin usin usin	ng System; ng System.Drawing; ng System.Collectio ng System.Component ng System.Windows.F	ns; Model; orms;					
name {	space JPEG_Manipul	ator					
// // pu	/ <summary> / Summary descript / </summary> ublic class frmSpla	ion for frmSplash. sh : System.Windows.Forms.Form					
ı	/// <summary> /// Required desig /// </summary> private System.Com	ner variable. ponentModel.Container components = null;					
	<pre>public frmSplash() {</pre>						
	InitializeCompon }	ent();					
	/// <summary> /// Clean up any r /// </summary> protected override	esources being used. void Dispose( bool disposing )					
	<pre>' if( disposing ) {     if(components     {</pre>	!= null)					
-							

## ISE Website Code Files

May 02, 04 2:32	Index.html	Page 1/3	May 02, 04 2:32	Index.html	Page 2/3
HTML PUBLIC "-//W</td <th>3C//DTD HTML 4.01 Transitional//EN"</th> <td></td> <td><pre><img alt="Manipulator" src="images/Manipu&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;latorShot1.jpg"/></pre></td> <td></td>	3C//DTD HTML 4.01 Transitional//EN"		<pre><img alt="Manipulator" src="images/Manipu&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;latorShot1.jpg"/></pre>		
<pre>"http://www.w3.org</pre>	g/TR/html4/loose.dtd">				
<pre><head></head></pre>	-Type" content="text/html; charset=ISO- -Language" content="en-US"> cryption Home	8859-1">	<pre><p align="center">    <font <b="" color="#006600" f="">Site Navigation    </font></p></pre>	ace="Verdana" size="6">	
<body bgcolor="#FFFFFF" tex<br="">vlink="#FF0000" alink="#FF0</body>	t="#000000" link="#0000FF" 000" >		<p><pre><font size="4"></font></pre></p>	to nine sections.	
<table al.<="" bgcolor="#003300" td=""><th>ign="center" width="100%"&gt;</th><td></td><td><pre></pre></td><td></td><td></td></table>	ign="center" width="100%">		<pre></pre>		
<p></p>			<a ch.<="" href="index.html" td=""><td>arset="ISO-8859-1"&gt;<b>Home</b></td></a>	arset="ISO-8859-1"> <b>Home</b>	
<font color:<br="" size="+1"><b>University of Colo: 2003-2004 </b></font>	="#FFFFFF"> rado Computer Science Department <br< th=""><td>&gt;</td><td>The Home page, which is relays information about</td><td>the current page, provides an introducti t the website.</td><td>on and</td></br<>	>	The Home page, which is relays information about	the current page, provides an introducti t the website.	on and
<pre><font color:<br="" size="+2"><b>ISE JPEG Selective </b></font></pre>	="#FFFFFF"> Encryption Home Page		<pre><a href="documents/Proj&lt;br&gt;charset=" iso-8859-1"=""><b </b </a></pre>	ectProposal.pdf" target="_blank" >Project Proposal	

  |  | The project proposal is the Team ISE project. | a PDF document giving a high level propo | sal for || Xsara Menu Maker. This to | import the menu bar. The bar was creat ool generated the javascript files used | ed using |  | html" charset="ISO-8859-1">**Documentati** | on |
| > | ten by the team. |  | ```  This section contains a ``` | II of the documentation produced for the | project. |
|  |

May 02, 04 2:32	Index.html	Page 3/3			
Users will find site a navigation with the exception of the message different sections of the website on the top level button will direc menu buttons, such as the Document will display a submenu when higlit to jump directly to a specific doc using the submenu to jump to docum opened. If you want to download t Documentation button. This will d can choose to either view or downl new users visit the pages rather t The pages contain additional info 	menu under the header board. The user can r by clicking on these k t the user to that pag ation Button and the I ed by the cursor. Thi ument or download. <f ents a PDF file of the he document, click on irect you to the docum oad the documents. Tea han using the quick li rmation which is usefu</f 	on every page, navigate to the puttons. Clicking ge. Some of the Download Button, is allows the user 3>Note: When a document will be the top-level ment page where you am ISE recommends inks in the submenu. all to new users.			
<p> <font size="2"> This project was done by <a <br="" href="http://www.colorado.edu">Colorado</a> students under the supervision of <a _blank"="" href="http://www.cs.colorado.ed&lt;br&gt;Department&lt;/a&gt;.&lt;br&gt;&lt;/FONT&gt;&lt;br&gt;&lt;/P&gt;&lt;/td&gt;&lt;td&gt;target=">Univer the u" target="_blank"&gt;Com</a></font></p>	sity of				
<p> <font size="2"> This website is located on a sever Boulder. Questions: Contact <a _blank"="" href="http://www.cs.colorado.ed&lt;br&gt;target=">Tom Lookabaugh</a> or <a href="mailto:TeamISE@hotmail.co&lt;br&gt;&lt;/FONT&gt;&lt;br&gt;&lt;/P&gt;&lt;/td&gt;&lt;td&gt;at the University of&lt;br&gt;u/people/tom_lookabauc&lt;br&gt;m">TeamISE@hotmail.com</a></font></p>	Colorado at gh.html" a.				
<table #fffff"="" align="cent&lt;br&gt;&lt;FONT color=" bgcolor="#003300"> <b>Team Image Selective Encr Lookabaugh</b> <bdepartment computer="" of="" sc<br=""><b>Duniversity of Colorado at <b>Boulder, CO 80309-0430<b>ENTML 4.01 Transitional<b>Copyright © 2003-2004   <p align="right"> <font color="#FFFFF"> Last Updated: <b>5/1/04</b> </font></p></b></b></b></b></bdepartment></table>	er" width="100%"> yption Sponsored by To ience Boulder > 	m			
May 02, 04 2:40	DocumentIndex.html	Page 1/5	May 02, 04 2:40	DocumentIndex.html	Page 2/5
---	--	----------	--	---	------------
DOCTYPE HTML PUBLIC "-/</td <td>//W3C//DTD HTML 4.01 Transitional//EN"</td> <td></td> <td><p></p></td> <td></td> <td></td>	//W3C//DTD HTML 4.01 Transitional//EN"		<p></p>		
"http://www.w3.	.org/TR/htm14/loose.dtd">		<font size="4"> This paper outlin</font>	es team ISE's research and development of	the ISE
			Selective Encrypt	ion for JPEG Baseline Compressed Images A	lgorithm.
<head> <meta content="text/html: charset=ISO-&lt;/td&gt;&lt;td&gt;8859-1" http-equiv="Conte&lt;/td&gt;&lt;td&gt;ent-Type"/></head>	 				
<meta content="en-US" http-equiv="Conte&lt;/td&gt;&lt;td&gt;ent-Language"/>	00009 1 9				
<title>Image Selective</title>	Encryption Documentation				
(/ HEAD>					
<body bgcolor="#FFFFFF" t<="" td=""><td><pre>text="#000000" link="#0000FF"</pre></td><td></td><td><a href="documents/&lt;/td&gt;&lt;td&gt;ISEreference.zip"></a></td><td></td></body>	<pre>text="#000000" link="#0000FF"</pre>		<a href="documents/&lt;/td&gt;&lt;td&gt;ISEreference.zip"></a>		
Vlink="#FF00000" alink="#F	2F0000">		<pre></pre>	e Manual">	
<table <="" bgcolor="#003300" td=""><td>align="center" width="100%"&gt;</td><td></td><td></td><td></td><td></td></table>	align="center" width="100%">				
<p></p>			<p></p>		
<pre><font col<br="" size="+1"><p>University of Col <p>University of Col <p>University</p></p></p></font></pre>	lor="#FFFFFF"> alarada Computar Saianaa Dapartments/B>sbr		<font size="4"></font>	once manual for the ISE class which imple	monts Toom
2003-2004	Siorado computer Scrence Department(/B/OF	-	ISE's Baseline JP	EG Selective Encryption algorithm. Team	ISE
			recommends that	you read this reference before using the	class.
<pre><foni col<br="" size="+2"><b>ISE JPEG Selection</b></foni></pre>	ive Encryption Documentation Page			ton to download the reference document.	
	· · · · · · · · · · · · · · · · · · ·				

					``` cl Those Seriet Command ```	ds import the menu har . The har was creat	od using		TCEmannagos zin"	
Xsara Menu Maker. This	s tool generated the javascript files used	• using	``` ISE ma</td><td>n pages ```							
These files were not wr	ritten by the team.									
>										

May 02, 04 2:40	DocumentIndex.html	Page 3/5	May 02, 04 2:40	DocumentIndex.html	Page 4/5
			prototype for the	project.	
<p></p>			<a href="documents&lt;/td&gt;&lt;td&gt;/ISEPrototypePlan.zip"></a>		
<font size="4"></font>		-	Download		
This is the refere	ence document for the JPEG Manipulator.	Any		he Duct them a Diler	
components can be	found in this document	ILS	C/FONTS	ne Prococype Pian.	
<a be<="" can="" href="documents" td=""><td>ManipulatorReference.zip"&gt;</td><td></td><td></td><td></td><td></td></a>	ManipulatorReference.zip">				
Download	, nanipalacoincicience. Lip >				
the .ZIP file of t	the Manipulator Reference.				
			<a href="documents/l&lt;/td&gt;&lt;td&gt;SESystemArchitectureDesign.pdf" target="&lt;/td"><td>"_blank"&gt;</td></a>	"_blank">	
			border="0" alt="Sv	satton_System_Architecture_Design.jpg	
				5 mon 200 /	
<a <="" href="documents/I&lt;/td&gt;&lt;td&gt;ISE_Developers_Reference/developer.htm" td=""><td></td><td></td><td></td><td></td></a>					
target="_blank">	Desidence Defenses des la des 101		<p></p>		
<img <="" border="0" src="images/L&lt;/td&gt;&lt;td&gt;Developers_Reference.jpg" td=""/> <td></td> <td><foni size="4"></foni></td> <td>a detailed explanation of the System 7</td> <td>rchitocturo</td>		<foni size="4"></foni>	a detailed explanation of the System 7	rchitocturo	
	Vererence >		Design of the ISE	project. The document outlines the arc	hitecture
			of the C++ class,	the JPEG Manipulator, and the website.	This
			document is helpfu	l to anyone wishing to extend the C++ o	lass to
<p></p>			perform Selective	Encryption methods on other types of fi	le formats.
<font size="4"></font>		P. Duridant	<a href="documents&lt;/td&gt;&lt;td&gt;/ISESystemArchitectureDesign.zip"></a>		
Inis is the develo	per's reference document for the leam is	L Project.	Download		
this document befo	a in excending of modifying the code shot	itu ieau	the ZIP file of t	he System Architecuture document.	
<a href="documents&lt;/td&gt;&lt;td&gt;s/ISE_Developers_Reference.zip"></a>					
Download					
the .ZIP file of t	the Developer's Reference.				
			<a href="documents/D&lt;/td&gt;&lt;td&gt;esignSpecFinal.pdf" target="_blank"></a>		
			<img alt="Des&lt;/td&gt;&lt;td&gt;ign Doc" border="0" src="images/d&lt;/td&gt;&lt;td&gt;lesign_document.jpg"/>		
<a <="" href="documents/I&lt;/td&gt;&lt;td&gt;[SEFinalRequirements ndf" target=" blank" td=""><td>``</td><td></td><td></td><td></td></a>	``				
<imq <="" border="0" src="images/&lt;/td&gt;&lt;td&gt;Button_Requirements.jpg" td=""><td>·</td><td><p></p></td><td></td><td></td></imq>	·	<p></p>			
alt="Requirements	s Doc">		<font size="4"></font>		
			This document cont	ains very detailed information about th	e design of
			the C++ class, the	JPEG Manipulator, and the website. The	is document
			Team ISE Anvone	interested in the design of the class of	vered by vr modifying
<font size="4"></font>			the class in anywa	y should read this document.	i modilying
This document outl	lines the requirements put foreward by Pr	oject	<a href="documents&lt;/td&gt;&lt;td&gt;/ISEDesign.zip"></a>		
Sponsor Tom Lookab	baugh. Team ISE followed these requireme	ents as	Download		
guidelines when de	eveloping the C++ class, the JPEG Manipul	ator		he Design desument	
and the website.	/ISFFinalRequirements zin">		The .ZIP file of t	ne Design document.	
Download	s/ibbrinainequirements.zip >				
the .ZIP file of t	the Requirements document.				
			<ta><ta><ta><ta><ta><ta><ta><ta><ta><ta></ta></ta></ta></ta></ta></ta></ta></ta></ta></ta>	'estPlan ndf" target-" hlank">	
			<pre><img alt="Test&lt;/td&gt;&lt;td&gt;Plan Doc" border="0" src="images/T&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;'est plan doc.jpg"/></pre>		
<a href="documents/I&lt;/td&gt;&lt;td&gt;[SEPrototypePlan.pdf" target="_blank"></a>					
<pre><li>src="images/b alt="Prototype Plane"</li></pre>	an">		<font size="4"></font>		
	···· /		This document cont	ains detailed information about the tes	st plan
			executed by Team I	SE on all of its software products. Th	le document
			also reports the s	tatus of the performed tests.	
<p></p>			<a href="documents&lt;/td&gt;&lt;td&gt;/ISETestPlan.zip"></a>		
This document outl	lines the plan devised by Team ISE to cre	ate a			

```
DocumentIndex.html
 May 02, 04 2:40
  Page 5/5
      the .ZIP file of the Test Plan document.
      </FONT>
    </P>
 </TABLE>
<P>
 <FONT size="2">
 This project was done by <a href="http://www.colorado.edu" target="_blank">University of
 Colorado</a>
 students under the supervision of the
 <a href="http://www.cs.colorado.edu" target="_blank">Computer Science
 Department</a>.
 </FONT>
</P>
<P>
 <FONT size="2">
 This website is located on a sever at the University of Colorado at
 Boulder. Questions: Contact
 <a href="http://www.cs.colorado.edu/people/tom_lookabaugh.html"
 target="_blank">Tom Lookabaugh</a>
 or
 <a href="mailto:TeamISE@hotmail.com">TeamISE@hotmail.com</a>.
 </FONT>
</P>
<TABLE bgcolor="#003300" align="center" width="100%">
 <P align="center">
      <FONT color="#FFFFFF">
        <B>Team Image Selective Encryption Sponsored by Tom
        Lookabaugh</B><br>
        <B>Department of Computer Science</B><br><B>University of Colorado at Boulder</B><br>
        <B>Boulder, CO 80309-0430</B><br>
        <B>HTML 4.01 Transitional</B><br>
        <B>Copyright & copy 2003-2004</B>
      </FONT>
    </P>
    <P align="right">
      <FONT color="#FFFFFF">
        Last Updated: <B>5/1/04</B>
      </FONT>
    </P>
 </TABLE>
</BODY>
</HTML>
```

May 02, 04 2:30	Download.html	Page 1/4	May 02, 04 2:30	Download.html	Page 2/4
HTML PUBLIC "-/<br "http://www.w3. <html></html>	/W3C//DTD HTML 4.01 Transitional//EN" org/TR/html4/loose.dtd">		the JPEG Baseline Comp any other methods could developers would (in m	ression Standard. The ISE class was de d easily inherit from the ISE class, th ost cases) only have to devise and impl	eveloped so uus .ement the
<head> <meta conte<="" http-equiv="Conte&lt;br&gt;&lt;META http-equiv=" td=""/><td>ent-Type" content="text/html; charset=I ent-Language" content="en-US"&gt;</td><td>SO-8859-1"&gt;</td><td><pre></pre></td><td>Igorithms for different file formats.</td><td></td></head>	ent-Type" content="text/html; charset=I ent-Language" content="en-US">	SO-8859-1">	<pre></pre>	Igorithms for different file formats.	
<title>ISE Downloads</title>	ITLE>		<p align="center"> <font :<br="" color="#006600"><b>Licensing</b></font></p>	face="Verdana" size="6">	
<body bgcolor="#FFFFFF" t<br="">vlink="#FF0000" alink="#F</body>	ext="#000000" link="#0000FF" F0000">		 		
<table <br="" bgcolor="#003300"><p> <font col<br="" size="+1"><b>University of Cc 2003-2004 <conts< td=""><td>align="center" width="100%"&gt; or="#FFFFFF"&gt; blorado Computer Science Department</td></conts<></b></font></p></table>	align="center" width="100%"> or="#FFFFFF"> blorado Computer Science Department		<pre><p> <font size="4"> All of the code provid to modify the code in a encryption algorithm the Any changes made to the agreement. Users may appear to be a size of the provide from any appear to the size of th</font></p></pre>	ed by Team ISE is open source. Users a any way. Some users may want to use a han the AES algorithm included in the I e ISE code must adhere to the Team ISE not remove the ISE name, the team membe barged code Mayney users may add th	ire allowed different SE class. licensing r's names,
<pre><font <b="" col="" size="+2">ISE JPEG Selecti </font> </pre>	or="#FFFFFF"> ve Encryption Download Page		<pre>nor receive row any changes made in the Rijndael code, v </pre>	which was not written by Team ISE.	ig agreement
			All ISE code is provide code in commercial app <a _blank"="" href="http://www.cs&lt;/td&gt;&lt;td&gt;ed for non-commercial use. If you wish&lt;br&gt;lications, you must first contact&lt;br&gt;.colorado.edu/people/tom lookabaugh htm&lt;/td&gt;&lt;td&gt;to use ISE&lt;/td&gt;&lt;/tr&gt;&lt;tr&gt;&lt;td&gt;&lt;! These Script Command&lt;br&gt;Xsara Menu Maker. This&lt;br&gt;These files were not wr&lt;br&gt;&gt;&lt;/td&gt;&lt;td&gt;Is import the menu bar. The bar was cr&lt;br&gt;tool generated the javascript files u&lt;br&gt;itten by the team.&lt;/td&gt;&lt;td&gt;eated using sed.&lt;/td&gt;&lt;td&gt;&lt;pre&gt;target=">Dr. Tor and receive his permiss <a href="http://www.col&lt;br&gt;&lt;/a&gt;&lt;/PONT&gt;&lt;br&gt;&lt;/P&gt;&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;n Lookabaugh&lt;/a&gt;&lt;br&gt;sion and the permission of the&lt;br&gt;lorado.edu/" target="_blank">University</a></a>	<sup>,</sup> of Colorado	
<p align="center"> <script images="" men<br="" src="images/men&lt;br&gt;&lt;SCRIPT src="></script></p>					

May 02, 04 2:30	Download.html	Page 3/4	May 02, 04 2:30	Download.html	Page 4/4
<pre>                    <p>  <font size="4">   <p>       Click on this lin refer to the user Manipulator. The research tool to   selective encrypt the JPEG Manipula    <b>Note:</b> The Framework to run.   <a href="code/dot you can download &lt;/fonts&lt;/a&gt;                                                                                                  &lt;img src=" images="" manipulator"=""></a></p></font></p></pre>	<pre>k to download the ISE JPEG Manipulator. Please tutorial for information on how to use the ISE JPEG Manipulator was developed as a aid Team ISE in creating a JPEG Baseline ion method. However, Team ISE is providing tor to interested users. JPEG Manipulator requires the Microsoft .NET If you do not have the netfx.exe"&gt;.NET Framework, it by clicking on the link below.</pre>		< <p> <font size="4"> Click on this link the software. This release, however it available above. T interested. </font></p>	<pre>to download Team ISE's Beta Test version o code is less buggy than the Alpha Test t still differs from the completed ISE code Team ISE is providing it for those who are by lorado.edu" target="_blank"&gt;University of C ervision of the .colorado.edu" target="_blank"&gt;Computer Sci</pre>	of ; Colorado Lence
<pre><lmg 100%"="" bor<="" src="images/Man1&lt;br&gt;&lt;/P&gt;&lt;br&gt;&lt;TABLE WIDTH=" td=""><td><pre>pulatorSnot2.jpg" alt="ISE Manipulator"&gt; DER="0" CELLSPACING="0" CELLPADDING="20"&gt; tfx.exe"&gt; Button_dot_net.jpg" ork" border="0"&gt;</pre></td><td></td><td>  <p> <font size="2"> This website is located Boulder. Questions: Cor <a _blank"="" href="http://www.cs.&lt;br&gt;target=">Tom Loc</a></font></p></td><td>d on a sever at the University of Colorado ntact .colorado.edu/people/tom_lookabaugh.html" okabaugh</td><td>at</td></lmg></pre>	<pre>pulatorSnot2.jpg" alt="ISE Manipulator"&gt; DER="0" CELLSPACING="0" CELLPADDING="20"&gt; tfx.exe"&gt; Button_dot_net.jpg" ork" border="0"&gt;</pre>		  <p> <font size="2"> This website is located Boulder. Questions: Cor <a _blank"="" href="http://www.cs.&lt;br&gt;target=">Tom Loc</a></font></p>	d on a sever at the University of Colorado ntact .colorado.edu/people/tom_lookabaugh.html" okabaugh	at
 <p> <font size="4"> Click on this lin The .NET framewor You must follow M</font></p>	k to download the Microsoft .Net Framework. k is required to run the ISE JPEG Manipulator.		or <a #003300"<="" href="mailto:TeamISH&lt;br&gt;&lt;/FONT&gt;&lt;br&gt;&lt;/F&gt;&lt;br&gt;&lt;TABLE bgcolor=" td=""><td><pre>E@hotmail.com"&gt;TeamISE@hotmail.com</pre></td></a> . align="center" width="100%">	<pre>E@hotmail.com"&gt;TeamISE@hotmail.com</pre>	
<pre>/romework.</pre>	pha.zip"> alpha_test_code.jpg" Code" border="0">		<pre><p align="center">     <font color="#FFFF         &lt;B&gt;Team Image Sel         Lookabaugh&lt;/B&gt;&lt;book         &lt;B&gt;Department of         &lt;B&gt;Duriversity of         &lt;B&gt;Boulder, CO 80         &lt;B&gt;HTML 4.01 Tran         &lt;B&gt;Copyright &amp;cop         &lt;/FONT&gt;     &lt;/P&gt;&lt;/td&gt;&lt;td&gt;FF"> lective Encryption Sponsored by Tom r&gt; Computer Science Colorado at Boulder 0309-0430 nsitional py 2003-2004</font></p></pre>		
<p>    Click on this lin   of the software.   completed ISE cod   providing it for   </p>	k to download Team ISE's Alpha Test version This code is buggy and differs from the e available above. However, Team ISE is those who are interested.		<pre></pre>	FF"> >5/1/04	
 <d> <img alpha="" src="images/&lt;br&gt;alt= " test<br=""/></d>	ta.zip"> ISE_Beta_Release.jpg" Code" border="0">				

May 02, 04 2:34	Links.html	Page 1/3	May 02, 04 2:34	Links.html	Page 2/3
HTML PUBLIC "-/<br "http://www.w3. <html></html>	/W3C//DTD HTML 4.01 Transitional//EN" org/TR/html4/loose.dtd">	'	<font size="4"> Clicking on this but the Rijndael block c</font>	ton will open a web page containing ipher in a new browser window. This	information on s block cipher
<head> <meta conte<br="" http-equiv="Conte&lt;br&gt;&lt;META http-equiv="/><title>ISE Related Link</title></head>	<pre>ent-Type" content="text/html; charset= ent-Language" content="en-US"&gt; cs</pre>	=ISO-8859-1">	is used in the C++ c Compression files. I can be downloaded fr  	lass for selectively encrypting JPE( mplementations of the Rijndael AES b om this page.	3 Baseline block cipher
<pre><body #000000"="" <br="" alink="#F&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;.ext=" bgcolor="#FFFFFF" link="#0000FF" t="" vlink="#FF0000">F0000"&gt;</body></pre>			lorado odu/lookabaugh/" targot-" bl	onk "N	
<table <br="" bgcolor="#003300">&gt;</table>	align="center" width="100%">		<pre><img 0"="" src="images/But&lt;br&gt;border="/> </pre>	ton_Project_Sponsor.jpg" alt= "Spons	sor's Home"
<p> <font col<br="" size="+1"><b>University of Co</b></font></p>	.or="#FFFFFF"> blorado Computer Science Department <td>3&gt;</td> <td></td> <td></td> <td></td>	3>			
2003-2004  <font col<br="" size="+2"><b>ISE JPEG Selecti</b></font>	.or="#FFFFFF"> .ve Encryption Links Page		<font size="4"> Clicking on this but site in a new browse </font>	ton will open project sponsor Dr. To r window.	om Lookabaugh's

	><				Xsara Menu Maker. This These files were not wr >	is import the menu bar. The bar was of tool generated the javascript files titten by the team.	created using used.	[alt= "University of](http://www.co <img src=)	lorado.edu/" target="\_blank"> versity\_of\_Colorado.jpg" Colorado" border="0">	
			Clicking on this but in a new browser win	ton will open the University of Cold dow.	orado home page					
	[	](http://www.cs</td></td><td>.colorado.edu/)								
```  itton\_ijg.jpg" alt= "ijg.org" border=' ```	0">		Department\_of\_Computer\_Science.jpg" order="0">							
<										

Clicking on this bu site in a new brows image compression 1	Itton will open the Independent JPEG G er window. IJG writes and distribute ibrary.	∂roup's (IJG) ≥s a free JPEG	Clicking on this but Science Department's	ton will open the University of Colo home page in a new browser window.	orado Computer
><				lorado.edu/" target="\_blank">	
	sat.kuleuven.ac.be/~rijmen/rijndael/" itton\_rij\_link.jpg" alt= "Rijndael Alq	joithm"		vepartment\_or\_iiv.jpg" ait= "ITD" bo	praer="U">
			Clicking on this but Interdisciplinary Te	ton will open the University of Cold lecommunications Department's home	orado page in a new

Links.html	Page 3/3
o.edu" target="_blank">University on of the ado.edu" target="_blank">Compute	of r Science
a sever at the University of Colo rado.edu/people/tom_lookabaugh.ht ugh	rado at ml"
nail.com">TeamISE@hotmail.com	
re Encryption Sponsored by Tom ter Science ado at Boulder id30 inal 3-2004	
	<pre>.edu" target="_blank"&gt;University on of the ado.edu" target="_blank"&gt;Compute sever at the University of Colo ado.edu/people/tom_lookabaugh.ht gh ail.com"&gt;TeamISE@hotmail.com ="center" width="100%"&gt; e Encryption Sponsored by Tom ter Science ado at Boulder 430 3-2004</pre>

May 02, 04 2:37	Contact.html	Page 1/2	May 02, 04 2:37	Contact.html	Page 2/2
<pre>May 02, 04 2:37 </pre> <pre> </pre>	Contact.html DIC//DTD HTML 4.01 Transitional//EN" rg/TR/html4/loose.dtd"> C-Type" content="text/html; charset=IS Language" content="en-US"> n At="#000000" link="#0000FF" DO00"> .ign="center" width="100%"> C="#FFFFFF"> Drado Computer Science Department C="#FFFFFF"> a Encryption Contact Page import the menu bar. The bar was created the javascript files us then by the team. (menu.js" type="text/javascript">'isemenu.js" type="text/javascript">'isemenu.js'isemenu.js" type="text/javascript">'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.js'isemenu.	Page 1/2 0-8859-1"> br> tated using ied. SCRIPT> 111 be  B>.	<pre>May 02, 04 2:37 </pre> <pre> <a a="" department<="" href="http://www.color Colorado&lt;/a&gt; &lt;pre&gt; students under the super &lt;a href=" http:="" www.cs.cd="">.          This website is located a Boulder. Questions: Conta <a _blank"="" href="http://www.cs.cd target=">Tom Looka or <a _blank"="" href="http://www.cs.cd target=">Tom Looka or <a #003300"="" <tr="" a.="" href="mailto:TeamISE@l &lt;/FONT&gt; &lt;/P&gt; &lt;/TABLE bgcolor="> &lt;   </a></a></a></a></pre> <td><pre>Contact.html rado.edu" target="_blank"&gt;University of vision of the olorado.edu" target="_blank"&gt;Computer S on a sever at the University of Colorad act olorado.edu/people/tom_lookabaugh.html" abaugh hotmail.com"&gt;TeamISE@hotmail.com. lign="center" width="100%"&gt; "&gt; ctive Encryption Sponsored by Tom omputer Science olorado at Boulder 09-04302/B&gt; ///04 "&gt; ///04 </pre></td> <td>Page 2/2</td>	<pre>Contact.html rado.edu" target="_blank"&gt;University of vision of the olorado.edu" target="_blank"&gt;Computer S on a sever at the University of Colorad act olorado.edu/people/tom_lookabaugh.html" abaugh hotmail.com"&gt;TeamISE@hotmail.com. lign="center" width="100%"&gt; "&gt; ctive Encryption Sponsored by Tom omputer Science olorado at Boulder 09-04302/B&gt; ///04 "&gt; ///04 </pre>	Page 2/2
<pre>dowever, you can also cor <a href="mailto:TeamISE@r&lt;br&gt;&lt;hr&gt;&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;notmail.com"><b>TeamISE@hotmail.com<td>&gt;&gt;</td></b></a>.</pre>	>>				
<pre></pre>	esponse will most likely be made if yo than Team ISE. However, Team ISE wo ments or concerns you may have.	u contact uld be			
<p> <font size="2"></font></p>					
This project was done by					

## Sponsor.html

May 02, 04 2:36	Sponsor.html	Page 1/2	May 02, 04 2:36	Sponsor.html	Page 2/2
<pre><!--up of the public "-//</td--><td><pre>/W3C//DTD HTML 4.01 Transitional//EN" org/TR/html4/loose.dtd"&gt; ht-Type" content="text/html; charset=IS ht-Language" content="en-US"&gt; Charyption Sponsor</pre></td><td>0-8859-1"&gt;</td><td><pre><rue></rue></pre> </td></pre> <p> <font size="2"> <td>ton_Project_Sponsor.jpg" alt= "Sponsor's ddle"&gt; by lorado.edu" target="_blank"&gt;University c</td><td>Home"</td></font></p>	<pre>/W3C//DTD HTML 4.01 Transitional//EN" org/TR/html4/loose.dtd"&gt; ht-Type" content="text/html; charset=IS ht-Language" content="en-US"&gt; Charyption Sponsor</pre>	0-8859-1">	<pre><rue></rue></pre>	ton_Project_Sponsor.jpg" alt= "Sponsor's ddle"> by lorado.edu" target="_blank">University c	Home"
<pre><body #003300"="" a<="" alink="#FF &lt;TABLE bgcolor=" bgcolor="#FFFFFF" pre="" te="" vlink="#FF0000"></body></pre>	ext="#000000" link="#0000FF" "0000"> align="center" width="100%">		<pre>students under the supe <a href="http://www.cs&lt;br&gt;Department&lt;/a&gt;.&lt;br&gt;&lt;/FONT&gt;&lt;/pre&gt;&lt;/td&gt;&lt;td&gt;ervision of the&lt;br&gt;.colorado.edu" target="_blank">Computer</a></pre>	Science	
<tp> <pre> <font <b="" cold="" size="+1">University of Col 2003-2004 ISE JPEG Selectiv </font>  </pre></tp>	pr="#FFFFFF"> .orado Computer Science Department< pr="#FFFFFF"> ye Encryption Sponsor	br>	<p> <font size="2"> This website is located Boulder. Questions: Con <a _blank"="" href="http://www.cs. target=">Tom Loc or <a href="mailto:TeamISH &lt;/FONT&gt; &lt;/P&gt;&lt;/td&gt;&lt;td&gt;d on a sever at the University of Colora&lt;br&gt;ntact&lt;br&gt;.colorado.edu/people/tom_lookabaugh.html&lt;br&gt;okabaugh&lt;/a&gt;&lt;br&gt;E@hotmail.com">TeamISE@hotmail.com</a>.</a></font></p>	ıdo at ."	
<pre><!-- INDEP<br--><!-- INDEP<br--><!-- INDEP<br--></pre> <pre></pre>	s import the menu bar. The bar was cre tool generated the javascript files us tten by the team. a/menu.js" type="text/javascript">a/isemenu.js" type="text/javascript"> <br on_white.jpg" alt="Team ISE"> ace="Verdana" size="6"> 'B>	ated using ed. IPT> SCRIPT>	<table <br="" bgcolor="#003300">&lt;  </table>	<pre>align="center" width="100%"&gt; FF"&gt; lective Encryption Sponsored by Tom r&gt; Computer Science Colorado at Boulder 0309-0430 nsitional py 2003-2004 FF"&gt; &gt;5/1/04</pre>	
<pre><r></r></pre> <pre><r></r></pre> <pre><r></r></pre> <pre></pre> <pre>&lt;</pre>	crption Project is being sponsored by A augh. In addition to sponsoring se the JPEG Baseline Compression format, P e selective encryption reserach on the g. Professor Lookabaugh is the Faculty brado.edu" target="_blank">Interdiscipl cam corado at Boulder. To visit Professor L con below.	<pre>ssociate lective rofessor MPEG-2 Director inary ookabaugh's</pre>	 		

May 02, 04 2:36	Team_ISE.html	Page 1/2	May 02, 04 2:36	Team_ISE.html	Page 2/2
HTML PUBLIC "-//</td <td>W3C//DTD HTML 4.01 Transitional//EN"</td> <td></td> <td><table <="" bgcolor="#003300" td=""><td>align="center" width="100%"&gt;</td><td></td></table></td>	W3C//DTD HTML 4.01 Transitional//EN"		<table <="" bgcolor="#003300" td=""><td>align="center" width="100%"&gt;</td><td></td></table>	align="center" width="100%">	
<pre>"nttp://www.w3.o <html></html></pre>	rg/IR/ntm14/100se.ata">		<		
<head> <meta conten<br="" http-equiv="Conten&lt;br&gt;&lt;META http-equiv="/><title>Team Image Select </title></head>	t-Type" content="text/html; charset=IS t-Language" content="en-US"> ive Encryption	0-8859-1">	<pre><p align="center">     <font color="#FFFFF     &lt;B&gt;Team Image Sel     Lookabaugh&lt;/B&gt;&lt;br/&gt;Cor     &lt;B&gt;Department of     &lt;B&gt;University of     &lt;/pre&gt;&lt;/td&gt;&lt;td&gt;F"> ective Encryption Sponsored by Tom &gt; Computer Science Colorado at Boulder</font></p></pre>		
<body bgcolor="#FFFFFF" te<br="">vlink="#FF0000" alink="#FF</body>	xt="#000000" link="#0000FF" 0000">		<pre><b>Boulder, CO 80 <b>HTML 4.01 Tran <b>Copyright &amp;cop</b></b></b></pre>	309-0430 sitional y 2003-2004	
<table a<br="" bgcolor="#003300"><p> <font colo<br="" size="+1"><b>University of Col 2003-2004 <font colo<br="" size="+2"><b>ISE JPEG Selectiv </b></font> </b></font></p></table>	lign="center" width="100%"> r="#FFFFFF"> orado Computer Science Department< r="#FFFFFF"> e Encryption Team Page	br>	<p align="right"> <font color="#FFFFF Last Updated: &lt;B&gt; &lt;/FONT&gt; &lt;/P&gt;  &lt;/rable&gt; &lt;/BODY&gt; &lt;/HTML&gt;&lt;/td&gt;&lt;td&gt;F"> 5/1/04</font></p>		

  |  |  |  |  || Xsara Menu Maker. This These files were not wri > | import the menu bar. The bar was cre tool generated the javascript files us tten by the team. | ated using ed. |  |  |  |
|  |