**Practical V**

**Roll No: 07 Date:16/11/22**

**Aim: Program to implement Steganography for hiding message inside the image file.**

**CODE:**

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.Scanner;

import javax.imageio.ImageIO;

public class LSB\_encode {

static final String MESSAGEFILE = "D:\\encimgs\\encmessage.txt";

static final String COVERIMAGEFILE = "D:\\encimgs\\logo.png";

static final String STEGIMAGEFILE = "D:\\encimgs\\steg.png";

// static final String STEGIMAGEFILE = "D:\\encimgs\\steg.png";

static final String DECODEDMESSAGEFILE = "D:\\encimgs\\message\_dec.txt";

public static String b\_msg="";

public static int len = 0;

public static void main(String[] args) throws Exception {

String contentOfMessageFile = (readMessageFile());

int[] bits=bit\_Msg(contentOfMessageFile);

System.out.println("Message in the file "+contentOfMessageFile);

for(int i=0;i<bits.length;i++)

System.out.print(bits[i]);

System.out.println();

BufferedImage theImage=readImageFile(COVERIMAGEFILE);

hideTheMessage(bits, theImage);

BufferedImage yImage=readImageFile(STEGIMAGEFILE);

DecodeTheMessage(yImage);

String msg="";

//System.out.println("len is "+len\*8);

for(int i=0;i<len\*8;i=i+8){

String sub=b\_msg.substring(i,i+8);

int m=Integer.parseInt(sub,2);

char ch=(char) m;

System.out.println("m "+m+" c "+ch);

msg+=ch;}

PrintWriter out = new PrintWriter(new FileWriter(DECODEDMESSAGEFILE, true), true);

out.write(msg);

out.close();}

public static BufferedImage readImageFile(String COVERIMAGEFILE){

BufferedImage theImage = null;

File p = new File (COVERIMAGEFILE);

try{

theImage = ImageIO.read(p);

}catch (IOException e){

e.printStackTrace();

System.exit(1);

}

return theImage;

}

public static String readMessageFile () throws FileNotFoundException{

String contentOfMessageFile = "";

File a = new File (MESSAGEFILE);

Scanner scan = new Scanner (a);

while (scan.hasNextLine()){

String next = scan.nextLine();

contentOfMessageFile += next;

if (scan.hasNextLine()){

contentOfMessageFile += "\n";}}

scan.close();

return contentOfMessageFile;}

public static int[] bit\_Msg(String msg){

int j=0;

int[] b\_msg=new int[msg.length()\*8];

for(int i=0;i<msg.length();i++){

int x=msg.charAt(i);

String x\_s=Integer.toBinaryString(x);

while(x\_s.length()!=8){

x\_s='0'+x\_s;}

System.out.println("dec value for "+x +" is "+x\_s);

for(int i1=0;i1<8;i1++) {

b\_msg[j] = Integer.parseInt(String.valueOf(x\_s.charAt(i1)));

j++;};}

return b\_msg;}

public static void DecodeTheMessage (BufferedImage yImage) throws Exception{

int j=0;

int currentBitEntry=0;

String bx\_msg="";

for (int x = 0; x < yImage.getWidth(); x++){

for ( int y = 0; y < yImage.getHeight(); y++){

if(x==0&&y<8){

//System.out.println("enc "+yImage.getRGB(x, y)+" dec "+yImage.getRGB(x, y)+" "+b\_msg);

int currentPixel = yImage.getRGB(x, y);

int red = currentPixel>>16;

red = red & 255;

int green = currentPixel>>8;

green = green & 255;

int blue = currentPixel;

blue = blue & 255;

String x\_s=Integer.toBinaryString(blue);

bx\_msg+=x\_s.charAt(x\_s.length()-1);

len=Integer.parseInt(bx\_msg,2);}

else if(currentBitEntry<len\*8){

//System.out.println("enc "+yImage.getRGB(x, y)+" dec "+yImage.getRGB(x, y)+" "+b\_msg);

int currentPixel = yImage.getRGB(x, y);

int red = currentPixel>>16;

red = red & 255;

int green = currentPixel>>8;

green = green & 255;

int blue = currentPixel;

blue = blue & 255;

String x\_s=Integer.toBinaryString(blue);

b\_msg+=x\_s.charAt(x\_s.length()-1);

currentBitEntry++;

//System.out.println("curre "+currentBitEntry);

}}}

System.out.println("bin value of message hidden inside image is "+b\_msg)}

public static void hideTheMessage (int[] bits, BufferedImage theImage) throws Exception{

File f = new File (STEGIMAGEFILE);

BufferedImage sten\_img=null;

int bit\_l=bits.length/8;

int[] bl\_msg=new int[8];

System.out.println("bit lent "+bit\_l);

String bl\_s=Integer.toBinaryString(bit\_l);

while(bl\_s.length()!=8){

bl\_s='0'+bl\_s;

}

for(int i1=0;i1<8;i1++) {

bl\_msg[i1] = Integer.parseInt(String.valueOf(bl\_s.charAt(i1)));

};

int j=0;

int b=0;

int currentBitEntry=8;

for (int x = 0; x < theImage.getWidth(); x++){

for ( int y = 0; y < theImage.getHeight(); y++){

if(x==0&&y<8){

int currentPixel = theImage.getRGB(x, y);

int ori=currentPixel;

int red = currentPixel>>16;

red = red & 255;

int green = currentPixel>>8;

green = green & 255;

int blue = currentPixel;

blue = blue & 255;

String x\_s=Integer.toBinaryString(blue);

String sten\_s=x\_s.substring(0, x\_s.length()-1);

sten\_s=sten\_s+Integer.toString(bl\_msg[b]);

//j++;

int temp=Integer.parseInt(sten\_s,2);

int s\_pixel=Integer.parseInt(sten\_s, 2);

int a=255;

int rgb = (a<<24) | (red<<16) | (green<<8) | s\_pixel;

theImage.setRGB(x, y, rgb);

//System.out.println("original "+ori+" after "+theImage.getRGB(x, y));

ImageIO.write(theImage, "png", f);

b++;}

else if (currentBitEntry < bits.length+8 ){

int currentPixel = theImage.getRGB(x, y);

int ori=currentPixel;

int red = currentPixel>>16;

red = red & 255;

int green = currentPixel>>8;

green = green & 255;

int blue = currentPixel;

blue = blue & 255;

String x\_s=Integer.toBinaryString(blue);

String sten\_s=x\_s.substring(0, x\_s.length()-1);

sten\_s=sten\_s+Integer.toString(bits[j]);

j++;

int temp=Integer.parseInt(sten\_s,2);

int s\_pixel=Integer.parseInt(sten\_s, 2);

int a=255;

int rgb = (a<<24) | (red<<16) | (green<<8) | s\_pixel;

theImage.setRGB(x, y, rgb);

//System.out.println("original "+ori+" after "+theImage.getRGB(x, y));

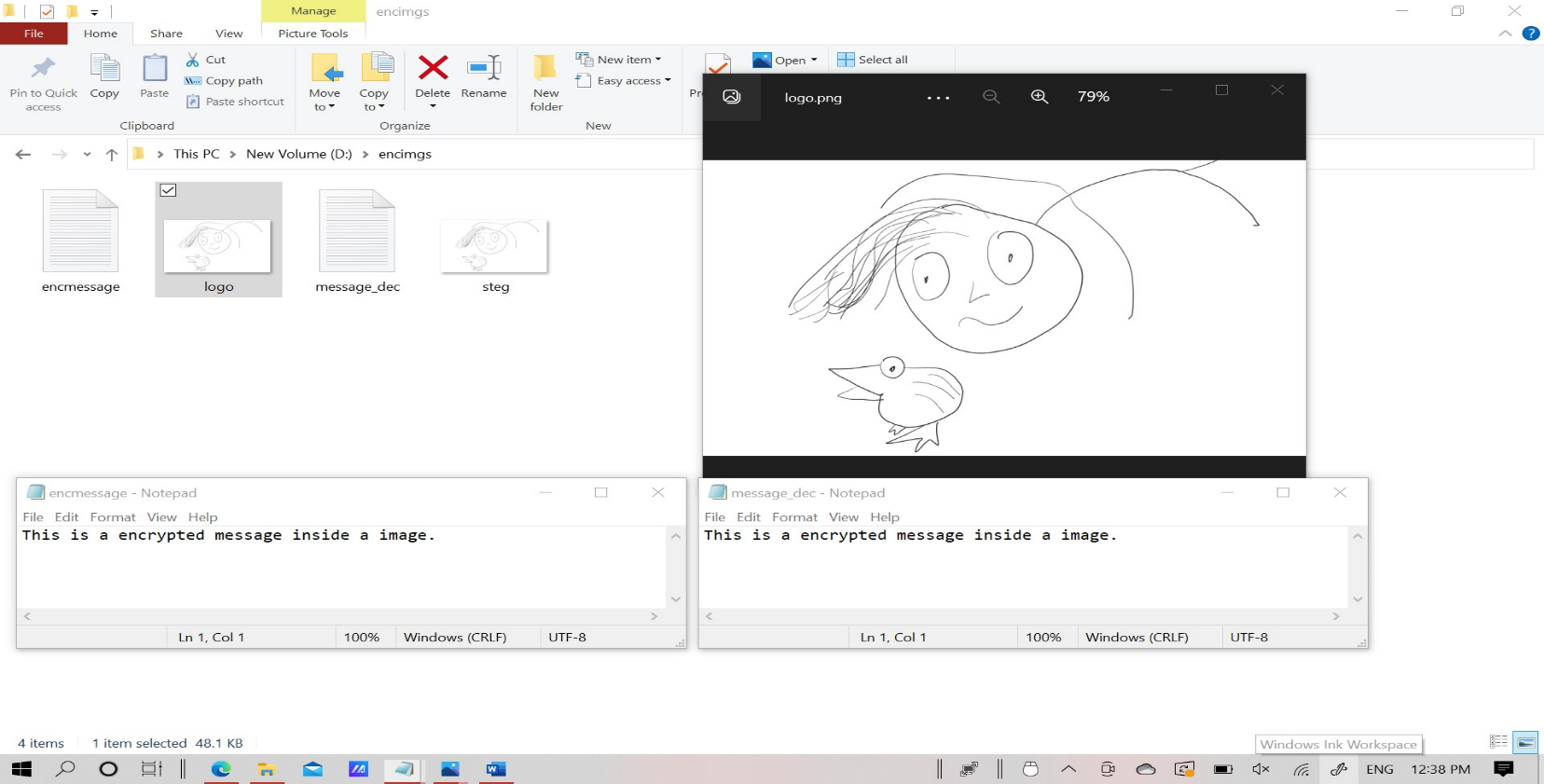
ImageIO.write(theImage, "png", f);

currentBitEntry++;

//System.out.println("curre "+currentBitEntry);

}}}}}

**OUTPUT:**

****

Message in the file This is a encrypted message inside a image.

01010100011010000110100101110011001000000110100101110011001000000110000100100000011001010110111001100011011100100111100101110000011101000110010101100100001000000110110101100101011100110111001101100001011001110110010100100000011010010110111001110011011010010110010001100101001000000110000100100000011010010110110101100001011001110110010100101110

bit lent 43

bin value of message hidden inside image is 01010100011010000110100101110011001000000110100101110011001000000110000100100000011001010110111001100011011100100111100101110000011101000110010101100100001000000110110101100101011100110111001101100001011001110110010100100000011010010110111001110011011010010110010001100101001000000110000100100000011010010110110101100001011001110110010100101110

BUILD SUCCESSFUL (total time: 18 seconds)