LAMPIRAN

Lampiran 1 : Kode *Enkripsi-Dekripsi File* Pesan

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package code;
/**
 *
 * @author dell user
 */
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.nio.charset.StandardCharsets;
public class Coding {
    String path;
    int[] aKey = null;
    int[] aPlain = null;
    int[] HasilEnkrip = null;
    //Extended ASCII Convertr
    public static final char[] EXTENDED = {
0x007f,0x0080,0x0081,0x0082,

        0x0083,0x0084,0x0085,0x0086,0x0087,0x0088,0x0089,

        0x008a,0x008b,0x008c,0x008d,0x008e,0x008f,0x0090,


0x0091,0x0092,0x0093,0x0094,0x0095,0x0096,0x0097,0x0098,0x0099,

        0x009a,0x009b,0x009c,0x009d,0x009f,


0x00a1,0x00a2,0x00a3,0x00a4,0x00a5,0x00a6,0x00a7,0x00a8,0x00a9,

        0x00aa,0x00ab,0x00ac,0x00ad,0x00af,0x00b0,


0x00b1,0x00b2,0x00b3,0x00b4,0x00b5,0x00b6,0x00b7,0x00b8,0x00b9,

        0x00ba,0x00bb,0x00bc,0x00bd,0x00be,0x00bf,0x00c0,


0x00c1,0x00c2,0x00c3,0x00c4,0x00c5,0x00c6,0x00c7,0x00c8,0x00c9,

        0x00ca,0x00cb,0x00cc,0x00cd,0x00ce,0x00cf,0x00d0,


0x00d1,0x00d2,0x00d3,0x00d4,0x00d5,0x00d6,0x00d7,0x00d8,0x00d9,

        0x00da,0x00db,0x00dc,0x00dd,0x00de,0x00df,0x00e0,


0x00e1,0x00e2,0x00e3,0x00e4,0x00e5,0x00e6,0x00e7,0x00e8,0x00e9,

        0x00ea,0x00eb,0x00ec,0x00ed,0x00ee,0x00ef,0x00f0};


    public static final char getAscii(int code) {

        if (code >= 0x80 && code <=0xFF) {

            return EXTENDED[code - 128];

        }
    return (char) code;
}
public int CekExtended(char a){
    for (int i = 0; i < EXTENDED.length; i++) {
        if(a == EXTENDED[i]){
            return i+128;
        }
    }
    return 63;
}

public static final char printChar(int code) {
    return getAscii(code);
}
public void plainToAscii(String kata){
    byte[] aa = kata.getBytes(StandardCharsets.US_ASCII);
    aPlain = new int [aa.length];
    for (int i = 0; i < aa.length; i++) {
        if(aa[i] ==  63){
            aPlain[i]=CekExtended(kata.charAt(i));
        }else{
            aPlain[i] = aa[i];
        }
        //System.out.println("asci plain : " + aPlain[i]};
    }
}


public void keyToAscii(String kata){
    byte[] aa = kata.getBytes(StandardCharsets.US_ASCII);
    aKey = new int[aPlain.length];
    int count = 0;
       for (int i = 0; i < aPlain.length; i++) {
           if (count >= aa.length){
               count = 0;
           }
           aKey[i] = aa[count];
           // System.out.println("asci key : " + akey[i]);
           count++;
       }
}


public String Enkrip (String plain, String key){
    plainToAscii(plain);
    keyToAscii(key);
    String enkrip;
    char[] x = new char[aPlain.length];
    for (int i = 0; i < aPlain.length; i++) {
            //System.out.println("hasil : " + (i+1)+" " + (aKey[i] +
aPlain[i])%256);
            x[i] = printChar((aKey[i] + aPlain[i])%256);
        }
        enkrip = String.valueOf(x);
        System.out.println("Hasil Enkripsi \n"+enkrip);
//        System.out.prinln(Dekrip(enkrip, key));
        return enkrip;
    }

    public String Dekrip (String plain, String key) throws IOException{
        String dekrip;
        System.out.println(plain);
        plainToAscii(plain);
        keyToAscii(key);
        System.out.println("path ini "+ path);
        FileWriter fw = new FileWriter(path);
        BufferedWriter bw = new BufferedWriter(fw);
        char[] x = new char[aPlain.length];
        System.out.println((aPlain[16]-aKey[16] )%256);
        for (int i = 0; i < aPlain.length; i++) {
 //          System.our.println(aPlain[i]);
            x[i] = printChar((aPlain[i]-aKey[i] )%256);
            System.out.println((aPlain[9]-aKey[9] )%256);
            if( ((aPlain[i]-aKey[i] )%256)!= 10 ){
               bw.write(x[i]);
            }else{
               bw.newLine();
            }
        }
        bw.close();
        dekrip = String.valueOf(x);
//        System.out.println(enkrip);
        return dekrip;
    }

   public void setPath(String sPath){
       path = sPath;
   }
}
```

Lampiran 2 : Kode *Filter* Gambar

```
package code;


/**
 *
 * @author dell user
 */


import java.io.*;


public class FilterGambar extends javax.swing.filechooser.FileFilter{
    protected boolean isImageFile(String ext){
        return
(ext.equals("jpg")||ext.equals("png")||ext.equals("jpeg")||ext.equals("bmp"
));
    }


    public boolean accept(File f){
    if (f.isDirectory()){
        return true;
    }


    String extension = getExtension(f);
    if (extension.equals("jpg")||extension.equals("png")){
        return true;
    }
    return false;
}
    public String getDescription(){
        return "Supported Image Files";
    }

    public static String getExtension(File f){
        String s = f.getName();
        int i = s.lastIndexOf('.');
        if (i > 0 && i < s.length() - 1)
        return s.substring(i+1).toLowerCase();
        return "";
    }
}
```

Lampiran 3 : Kode Steganografi

```
package code;
/**
 *
 * @author dell user
 */

import java.io.File;
import java.awt.Graphics2D;
import java.awt.image.BufferedImage;
import java.awt.image.WritableRaster;
import java.awt.image.DataBufferByte;
import javax.imageio.ImageIO;
import javax.swing.JOptionPane;
```

```
public class Steganografi {


    public Steganografi(){
    }


    /* compress */
    public boolean encode(String path, String original, String ext1, String
stegan, String message){
        String file_name = image_path(path,original,ext1);
        BufferedImage image_orig = getImage(file_name);
        BufferedImage image = user_space(image_orig);
        image = add_text(image,message);
        return(setImage(image,new
File(image_path(path,stegan,"png")),"png"));
    }


    /*ekstrak*/
    public String decode(String path, String name){
        byte[] decode;
        try{
            BufferedImage image =
user_space(getImage(image_path(path,name,"png")));
            decode = decode_text (get_byte_data(image));
            return(new String(decode));
        }catch(Exception e){
          JOptionPane.showMessageDialog(null,
                "Tidak ada pesan rahasia yang tersembunyi di gambar
ini!","Error",
                JOptionPane.ERROR_MESSAGE);
          return "";
        }
    }
    private String image_path(String path, String name, String ext){
        return path + "/" + name + "." + ext;
    }

    private BufferedImage getImage(String f){
        BufferedImage image = null;
        File file = new File(f);
        try{
            image = ImageIO.read(file);
        }catch(Exception ex){
            JOptionPane.showMessageDialog(null,
              "Image could not be
read!","Error",JOptionPane.ERROR_MESSAGE);
        }
        return image;
    }
    private boolean setImage(BufferedImage image, File file, String ext){
        try{
            file.delete();
            ImageIO.write(image, ext, file);
            return true;
        }catch(Exception e){
            JOptionPane.showMessageDialog(null,
                    "File could not be
saved","Error",JOptionPane.ERROR_MESSAGE);
            return false;
        }
    }

    private BufferedImage add_text(BufferedImage image, String text){
        byte img[] = get_byte_data(image);
        byte msg[] = text.getBytes();
        byte len[] = bit_conversion(msg.length);
        try{
            encode_text(img, len, 0);
            encode_text(img, msg, 32);
        }catch(Exception e){
            JOptionPane.showMessageDialog(null,
                    "Target File cannot hold
message!","Error",JOptionPane.ERROR_MESSAGE);
        }
        return image;
    }


    private BufferedImage user_space(BufferedImage image){
        BufferedImage new_img = new BufferedImage(image.getWidth(),
image.getHeight(), BufferedImage.TYPE_3BYTE_BGR);
        Graphics2D graphics = new_img.createGraphics();
        graphics.drawRenderedImage(image, null);
        graphics.dispose();
        return  new_img;
    }


    private byte[] get_byte_data(BufferedImage image){
        WritableRaster raster = image.getRaster();
        DataBufferByte buffer = (DataBufferByte)raster.getDataBuffer();
        return buffer.getData();
    }


    private byte[] bit_conversion(int i){
        byte byte3 = (byte) ((i & 0xFF000000) >>> 24);
        byte byte2 = (byte) ((i & 0x00FF0000) >>> 16);
        byte byte1 = (byte) ((i & 0x0000FF00) >>> 8 );
        byte byte0 = (byte) ((i & 0x000000FF)       );
        return(new byte[]{byte3,byte2,byte1,byte0});
    }


    private byte[] encode_text(byte[] image, byte[] addition, int offset){
        if(addition.length + offset > image.length){
            throw new IllegalArgumentException("File not long enough!");
        }


        for(int i=0; i<addition.length; ++i){
            int add = addition[i];
            for(int bit=7; bit>=0; --bit, ++offset){
                int b = (add >>> bit) & 1;
                image[offset] = (byte)((image[offset] & 0xFE) | b);
            }
        }
        return image;
    }


    private byte[] decode_text(byte[] image){
        int length = 0;
        int offset = 32;
        for(int i=0; i<32; ++i){
            length = (length << 1) | (image[i] & 1);
        }
        byte[] result = new byte[length];


        for(int b=0; b<result.length; ++b ){
            for(int i=0; i<8; ++i, ++offset){
                result[b] = (byte)((result[b] << 1) | (image[offset] & 1));
            }
        }
    return result;
    }
```