LAMPIRAN 1

SOURCE CODE UNTUK *FUNCTION BROWSE FILE*

```
private void browseActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        BufferedReader fileReader;
        String line;
        int returnVal = fc.showOpenDialog(SIPP.this);

        if (returnVal == JFileChooser.APPROVE_OPTION) {
            file = fc.getSelectedFile();
            nmfile = file.getPath();
            nmfile = cekPath(nmfile);
            tf.setText(nmfile);

            try {
                fileReader = new BufferedReader(new FileReader(file));
                while (true) {
                    line = fileReader.readLine();
                    if (line == null) {
                        break;
                    }
                }
                txtPath.setText(nmfile);

            } catch (IOException io) {
            }
        }
    }

    private void txtPathActionPerformed(java.awt.event.ActionEvent evt) {
        //PENCARIAN FILE:
}
```

LAMPIRAN 2

SOURCE CODE PERHITUNGAN KUNCI

```
private int keygenerate(String Kunci) {

        int key = 0;
        int key1 = 0;
        int key2 = 0;
        int key3 = 0;
        int key4 = 0;
        String temp = Kunci.substring(0, 1);
        String temp1 = Kunci.substring(1, 2);
        String temp2 = Kunci.substring(2, 3);
        String temp3 = Kunci.substring(3, 4);
        key = Integer.parseInt(temp);
        key1 = Integer.parseInt(temp1);
        key2 = Integer.parseInt(temp2);
        key3 = Integer.parseInt(temp3);
        key4 = key1 + key2 + key3;
        String temp4 = ""+key4;
        key4 = Integer.parseInt(""+temp4.charAt(temp4.length()-1));
        key = key + key4;
        if (key > 10||key==0) {
            key = key - key2;
            if (key <= 0) {
                key = key + key1;
            }
            if (key > 10) {
                key = key - key3;
            }
            if (key <= 0) {
                key = key + key1;
            }
            if (key > 10) {
                key = key - key2;
            }
            if (key <= 0) {
                key = Math.abs(key);
            }
        }
        return key;}
```

LAMPIRAN 3

SOURCE CODE ENKRIPSI

```
public static void proses(int key,int key2) {
        int ss = 0;
        try {
            File inputImage = new File(nmfile);
            gambar = ImageIO.read(inputImage);
            imageWidth = gambar.getWidth();
            imageHeight = gambar.getHeight();
            besarMatrik = imageWidth * imageHeight;
            ss = gambar.getType();
        } catch (IOException io) {
            System.out.println("Error:" + io.getMessage());
        }
        JFrame frame = new JFrame("Hasil Enkripsi");
        JFrame frame1 = new JFrame("Hasil Permutasi Piksel");
        enkripsi e = new enkripsi();
        inisialisasi();


        ///padding piksel
        int tempNode[] = new int[matriks_image.length];
        int nodeAwal = 0, nodeTujuan = 0;
        int Kr = key;
        p = padding_piksel(imageWidth, imageHeight);
        int mLama = 0;
        mLama = imageHeight;
        if (imageWidth % imageHeight != 0) {
            if (imageHeight < imageWidth) {

                imageWidth = imageWidth + p;
                a = (imageWidth) / imageHeight;
                b = imageHeight;
                pad = false;
            } else if (imageHeight > imageWidth) {
                imageHeight = imageHeight + p;
                a = (imageHeight) / imageWidth;
                b = imageWidth;
                pad = true;
            }
        }
        ///
        gg = new BufferedImage(imageWidth, imageHeight, ss);//
        ubah();//

        int matriks_hasil[][] = new int[imageWidth][imageHeight];
        for (int q = 0; q < imageWidth; q++) {
            for (int r = 0; r < mLama; r++) {
                if (q >= matriks_image.length) {
                    matriks_hasil[q][r] = -1;
                } else {
                    matriks_hasil[q][r] = matriks_image[q][r];
                }

            }
        }
        matriks_image = new int[imageWidth][imageHeight];
        System.out.println(imageHeight + "--" + imageWidth);
```

```
        for (int q = 0; q < imageWidth; q++) {
            System.arraycopy(matriks_hasil[q], 0, matriks_image[q], 0,
imageHeight);
        }

        System.out.println("a =" + a);
        System.out.println("b :" + b);

//        rubah();
        // iterasi sebanyak kr
        for (int iter = 0; iter < Kr; iter++) {
//            /fungsi permutasi piksel
            for (int k = 0; k < a; k++) {
                for (int i = 0; i < b; i++) {
                    for (int y = 0; y < tempNode.length; y++) {
                        tempNode[y] = -1;
                    }
                    int isitempNode = 0;
                    nodeAwal = b / 2 + i + k;
                    nodeAwal = cek(nodeAwal, b);
                    for (int j = 0; j < b / 2; j++) {
                        nodeAwal = nodeAwal + j;
                        nodeAwal = cek(nodeAwal, b);
                        nodeAwal = cekNode(nodeAwal, tempNode, b);
                        nodeTujuan = nodeAwal + 2 + j + i + k;
                        nodeTujuan = cek(nodeTujuan, b);
                        nodeTujuan = cekNode(nodeTujuan, tempNode, b);
                        tempNode[isitempNode] = nodeAwal;
                        isitempNode++;
                        tempNode[isitempNode] = nodeTujuan;
                        isitempNode++;
                        swap(nodeAwal, nodeTujuan, b, i, j, k);
                    }
                }
            }
            rubah();
            RGBtoGrayscale();////rgb grayscale on / off
//mematikan proses xor
            if (pad) {
                for (int k = 0; k < imageWidth; k++) {
                    for (int l = 0; l < imageHeight - 1; l++) {
                        matriks_image[k][l + 1] =
xor(matriks_image[k][l], matriks_image[k][l + 1]);
                    }
                }
            } else if (!pad) {
                for (int k = 0; k < imageWidth; k++) {
                    for (int l = 0; l < imageHeight - 1; l++) {
                        matriks_image[k][l + 1] =
xor(matriks_image[k][l], matriks_image[k][l + 1]);
                    }
                }
            }//mematikan proses xor
            int temp[];
            for (int l = 0; l < matriks_image.length; l++) {

                int shift_no = 0;
                shift_no =( l % 25)+ key2;
                temp = rotate_left(matriks_image, shift_no, l,
imageHeight);
                System.arraycopy(temp, 0, matriks_image[l], 0,
imageHeight);
            }
            ///
```

```
            rubah();
        }
        rubah();
        simpan4();
        tampilGambar(frame);
        fileWrite();
    }

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

        try {
            File inputImage = new File(nmfile);
            gambar = ImageIO.read(inputImage);
            imageWidth = gambar.getWidth();
            imageHeight = gambar.getHeight();
            //System.out.println(imageWidth + " : " + imageHeight);
        } catch (IOException io) {
            System.out.println("Error:" + io.getMessage());
        }
    }
}
```

LAMPIRAN 4
SOURCE CODE DEKRIPSI

```
public void proses(int key,int key2) {
        try {
            File inputImage = new File(nmfile);
            gambar = ImageIO.read(inputImage);
            imageWidth = gambar.getWidth();
            imageHeight = gambar.getHeight();
            int ss = 0;
            ss = gambar.getType();
            //System.out.println(imageWidth + " : " + imageHeight);
        } catch (IOException io) {
            System.out.println("Error:" + io.getMessage());
        }

        JFrame frame = new JFrame("Hasil Dekripsi");
        gg = new BufferedImage(imageWidth, imageHeight, 1);
        dekripsi e = new dekripsi();
        inisialisasi();

        int tempNode[] = new int[matriks_hasil.length];
        int nodeAwal = 0, nodeTujuan = 0;
        int Kr = key;
        //ubah();
        rubah();
// iterasi sebanyak kr
        for (int iter = 0; iter < Kr; iter++) {

            //fungsi ketiga pergeseran piksel
            int temp[];
            for (int l = 0; l < matriks_hasil.length; l++) {
                int shift_no = 0;
                shift_no = (l % 25)+ key2;
                //for(int k = 0; k<m;k++){
                temp = rotate_right(matriks_hasil, shift_no, l,
imageHeight);
                System.arraycopy(temp, 0, matriks_hasil[l], 0,
imageHeight);
            }
            if (imageHeight < imageWidth) {
```

```
//                imageWidth = imageWidth + p;
                a = (imageWidth) / imageHeight;
                b = imageHeight;
                del_pad = false;
            } else if (imageHeight > imageWidth) {
//                imageHeight = imageHeight + p;
                a = (imageHeight) / imageWidth;
                b = imageWidth;
                del_pad = true;
            }
            rubah();
//          //       fungsi kedua XOR on off
            if (del_pad) {
                for (int k = imageWidth - 1; k > -1; k--) {
                    for (int l = imageHeight - 2; l > - 1; l--) {
                        matriks_hasil[k][l + 1] =
xor(matriks_hasil[k][l], matriks_hasil[k][l + 1]);
                    }
                }
            } else if (!del_pad) {
                for (int k = imageWidth - 1; k > -1; k--) {
                    for (int l = imageHeight - 2; l > - 1; l--) {
                        matriks_hasil[k][l + 1] =
xor(matriks_hasil[k][l], matriks_hasil[k][l + 1]);
                    }
                }
            }//mematikan xor
//            rubah();
//
            //fungsi permutasi piksel

            for (int k = 0; k < a; k++) {
                for (int i = 0; i < b; i++) {
                    for (int y = 0; y < tempNode.length; y++) {
                        tempNode[y] = -1;
                    }
                    int isitempNode = 0;
                    nodeAwal = b / 2 + i + k;
                    nodeAwal = cek(nodeAwal, b);
                    for (int j = 0; j < b / 2; j++) {
```

```
                        nodeAwal = nodeAwal + j;
                        nodeAwal = cek(nodeAwal, b);
                        nodeAwal = cekNode(nodeAwal, tempNode, b);
                        nodeTujuan = nodeAwal + 2 + j + i + k;
                        nodeTujuan = cek(nodeTujuan, b);
                        nodeTujuan = cekNode(nodeTujuan, tempNode, b);
                        tempNode[isitempNode] = nodeAwal;
                        isitempNode++;
                        tempNode[isitempNode] = nodeTujuan;
                        isitempNode++;
                        swap(nodeAwal, nodeTujuan, b, i, j, k);
                    }
                }
            }
            rubah();
        }
        if (imageHeight < imageWidth) {

            imageWidth = imageWidth - p;

        } else if (imageHeight > imageWidth) {
            imageHeight = imageHeight - p;

        }
        gg = new BufferedImage(imageWidth, imageHeight, 1);

        rubah();
//        rubah();
        RGBtoGrayscale();/// rgb grayscale on / off
//
        //rubah();
        simpan();
```

LAMPIRAN 5
SOURCE CODE STEP

```
private void butStepEnkripActionPerformed(java.awt.event.ActionEvent evt)
{
        // TODO add your handling code here:
        enkripsi proses = new enkripsi(txtPath.getText());
        System.out.println("step:" + step);
        if (step == 0) {//proses padding piksel
            proses.step(0);
            //proses.simpan();
        } else if (step == 1) {//proses permutasi piksel
            proses.step(1);
            //proses.simpan1();
        } else if (step == 2) {//proses RGBtoGrayscale on // off rubah
step dibawah
            proses.RGBtoGrayscale();//proses on off
            //proses.simpan2();//gak perlu
        } else if (step == 3) {//proses XOR // jika RGBtoGrayscale step
jadi 3  dan 2 rgb
            proses.step2();
            //proses.simpan3();
        } else if (step == 4) {//proses shifting piksel // jika
RGBtoGrayscale step jadi 4  dan 3 rgb
            proses.step3();
            //proses.simpan4();
            step = -1;
        }
        step++;

    }
```

LAMPIRAN 6

SOURCE CODE EXIT

```
private void exitActionPerformed(java.awt.event.ActionEvent evt) {
                    // TODO add your handling code here:
                              System.exit(1);
                                   }
```