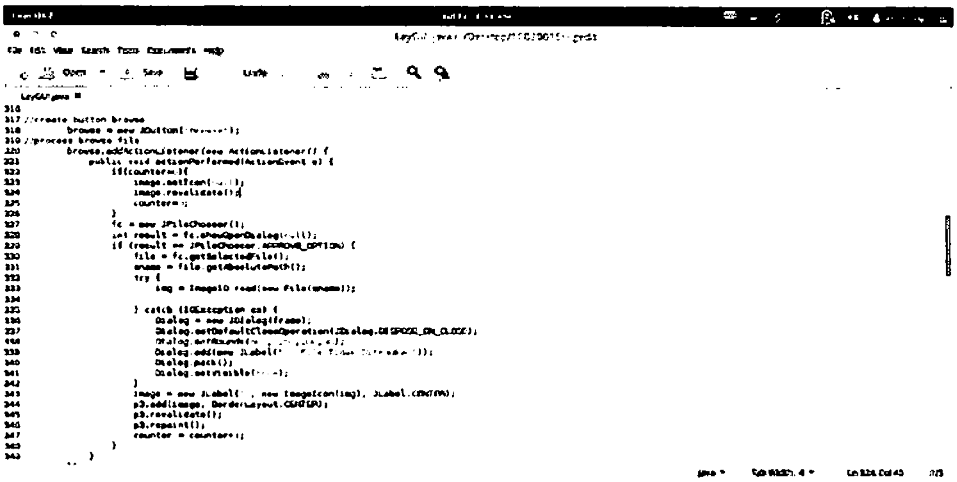
CHAPTER V

IMPLEMENTATION AND TESTING

5.1 Implementation

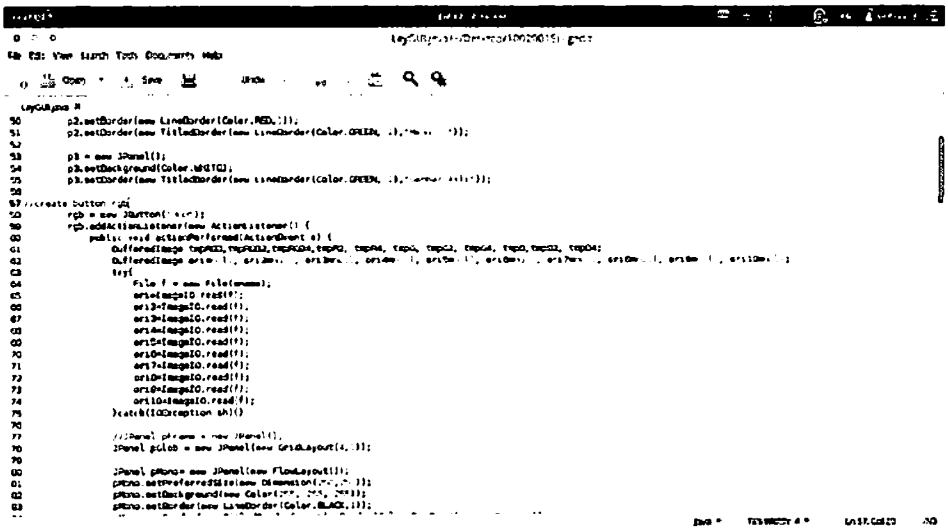
5.1.1 Button Menu Process



```
//create button browse
        browse = new JButton([illegible]);
//process browse file
        browse.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if(counter==[illegible]){
                    image.setIcon([illegible]);
                    image.revalidate();
                    counter=[illegible];
                }
                fc = new JFileChooser();
                int result = fc.showOpenDialog([illegible]);
                if (result == JFileChooser.APPROVE_OPTION) {
                    file = fc.getSelectedFile();
                    aname = file.getAbsolutePath();
                    try {
                        img = ImageIO.read(new File(aname));

                    } catch (IOException ea) {
                        Dialog = new JDialog(frame);
                        Dialog.setDefaultCloseOperation(JDialog.DISPOSE_ON_CLOSE);
                        Dialog.setBounds([illegible]);
                        Dialog.add(new JLabel([illegible]));
                        Dialog.pack();
                        Dialog.setVisible([illegible]);
                    }
                    image = new JLabel([illegible], new ImageIcon(img), JLabel.CENTER);
                    p3.add(image, BorderLayout.CENTER);
                    p3.revalidate();
                    p3.repaint();
                    counter = counter++;
                }
        }
```

Figure 5.1 Button Browse RGB image

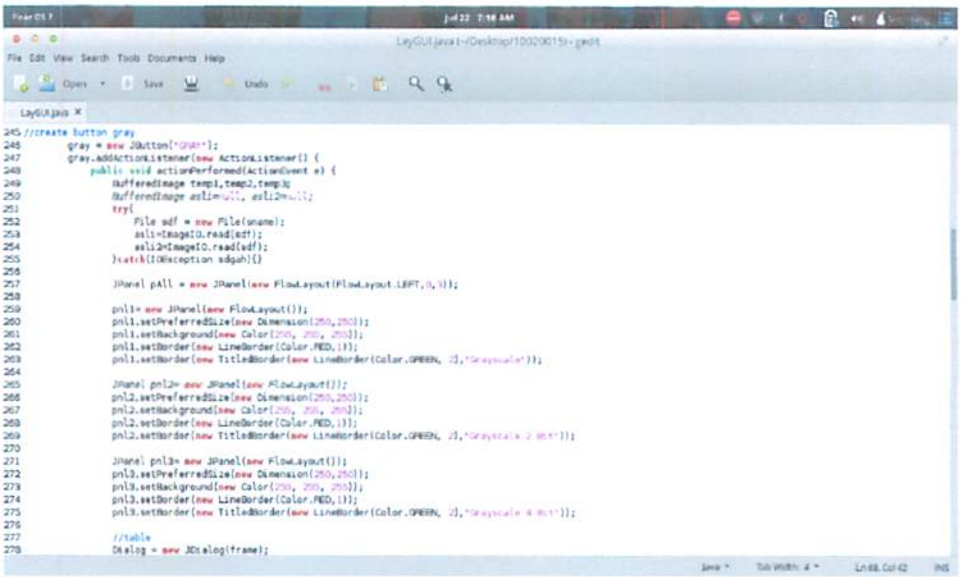
At the beginning of running the application is the user can choose the image to be processed with the browse button and appears on the screen of the new results.



```
        p2.setBorder(new LineBorder(Color.RED,[illegible]));
        p2.setBorder(new TitledBorder(new LineBorder(Color.GREEN, [illegible]),[illegible]));

        p3 = new JPanel();
        p3.setBackground(Color.WHITE);
        p3.setBorder(new TitledBorder(new LineBorder(Color.GREEN, [illegible]),[illegible]));

//create button rgb
        rgb = new JButton([illegible]);
        rgb.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                BufferedImage tmpRGB,tmpRGB2,tmpRGB4,tmpR2, tmpR4, tmpG, tmpG2, tmpG4, tmpB,tmpB2, tmpB4;
                BufferedImage ori=[illegible], ori2=[illegible], ori3=[illegible], ori4=[illegible], ori5=[illegible], ori6=[illegible], ori7=[illegible], ori8=[illegible], ori9=[illegible], ori10=[illegible];
                try{
                    File f = new File(aname);
                    ori=ImageIO.read(f);
                    ori2=ImageIO.read(f);
                    ori3=ImageIO.read(f);
                    ori4=ImageIO.read(f);
                    ori5=ImageIO.read(f);
                    ori6=ImageIO.read(f);
                    ori7=ImageIO.read(f);
                    ori8=ImageIO.read(f);
                    ori9=ImageIO.read(f);
                    ori10=ImageIO.read(f);
                }catch(IOException eh){}

                //JPanel pframe = new JPanel();
                JPanel pGlob = new JPanel(new GridLayout(4,[illegible]));

                JPanel pMono= new JPanel(new FlowLayout());
                pMono.setPreferredSize(new Dimension([illegible]));
                pMono.setBackground(new Color([illegible]));
                pMono.setBorder(new LineBorder(Color.BLACK,1));
```

Figure 5.2 Button RGB Process

The user can select the image that will be processed, after which the rgb button serves to display the results of the process of rgb.

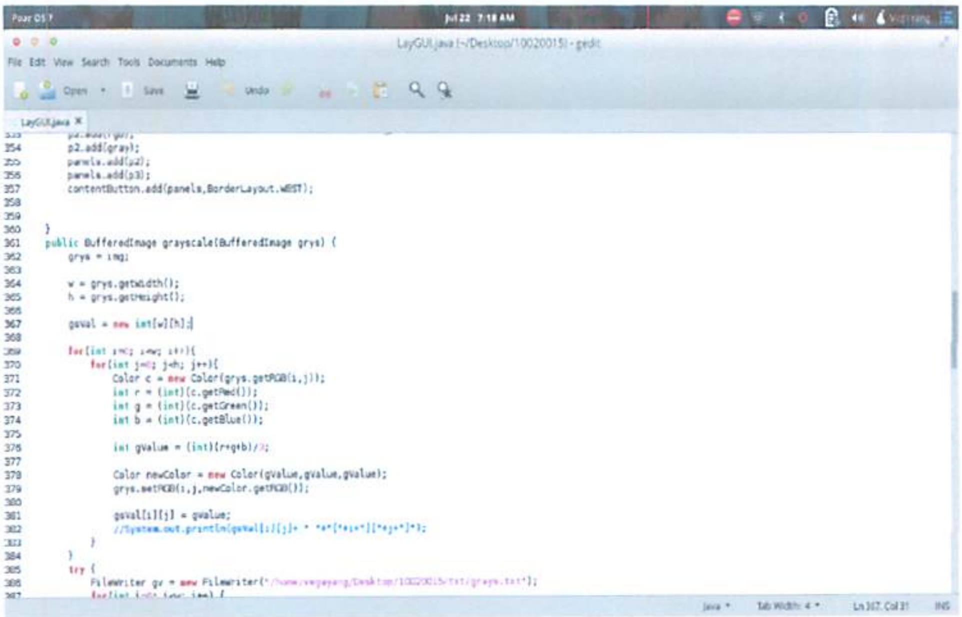

LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
LayGUI.java

```
//create button gray
        gray = new JButton("GRAY");
        gray.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                BufferedImage temp1,temp2,temp3;
                BufferedImage asli=null, asli2=null;
                try{
                    File sdf = new File(sname);
                    asli=ImageIO.read(sdf);
                    asli2=ImageIO.read(sdf);
                }catch(IOException sdgah){}

                JPanel pAll = new JPanel(new FlowLayout(FlowLayout.LEFT,0,3));

                pnl1= new JPanel(new FlowLayout());
                pnl1.setPreferredSize(new Dimension(250,250));
                pnl1.setBackground(new Color(255, 255, 255));
                pnl1.setBorder(new LineBorder(Color.RED,1));
                pnl1.setBorder(new TitledBorder(new LineBorder(Color.GREEN, 2),"Grayscale"));

                JPanel pnl2= new JPanel(new FlowLayout());
                pnl2.setPreferredSize(new Dimension(250,250));
                pnl2.setBackground(new Color(255, 255, 255));
                pnl2.setBorder(new LineBorder(Color.RED,1));
                pnl2.setBorder(new TitledBorder(new LineBorder(Color.GREEN, 2),"Grayscale 2 Bit"));

                JPanel pnl3= new JPanel(new FlowLayout());
                pnl3.setPreferredSize(new Dimension(250,250));
                pnl3.setBackground(new Color(255, 255, 255));
                pnl3.setBorder(new LineBorder(Color.RED,1));
                pnl3.setBorder(new TitledBorder(new LineBorder(Color.GREEN, 2),"Grayscale 4 Bit"));

                //table
                Dialog = new JDialog(frame);
```

Figure 5.3 Button Gray Process

The user can select the image that will be processed, then the grayscale function to display results from the process of Grayscale.

5.1.2 RGB to Grayscale Process

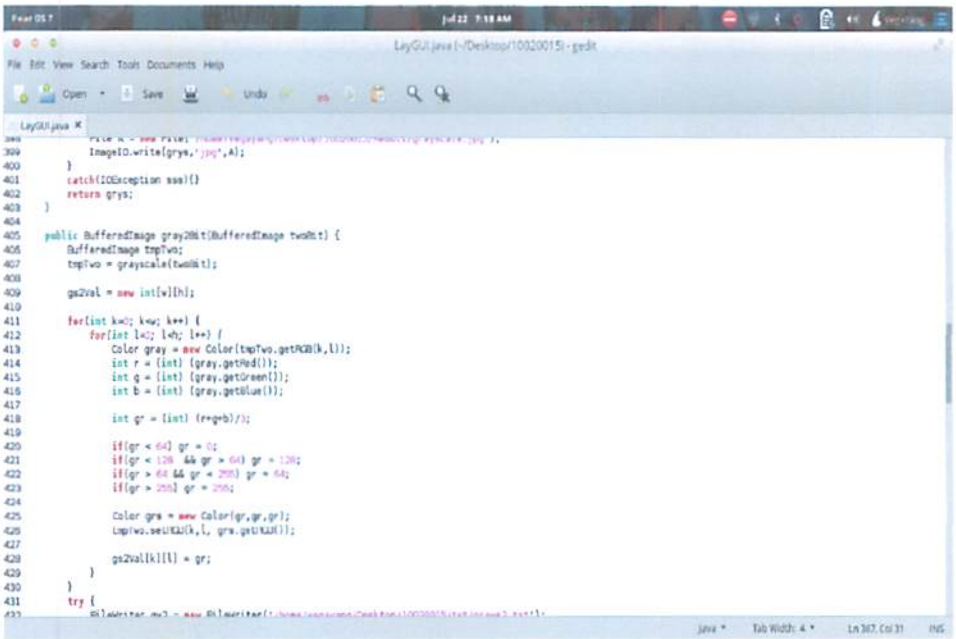

LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open Save Undo
LayGUI.java

```
        p2.add(gray);
        panels.add(p2);
        panels.add(p3);
        contentButton.add(panels,BorderLayout.WEST);


    }
    public BufferedImage grayscale(BufferedImage grys) {
        grys = img;

        w = grys.getWidth();
        h = grys.getHeight();

        gsval = new int[w][h];

        for(int i=0; i<w; i++){
            for(int j=0; j<h; j++){
                Color c = new Color(grys.getRGB(i,j));
                int r = (int)(c.getRed());
                int g = (int)(c.getGreen());
                int b = (int)(c.getBlue());

                int gValue = (int)(r+g+b)/3;

                Color newColor = new Color(gValue,gValue,gValue);
                grys.setRGB(i,j,newColor.getRGB());

                gsVal[i][j] = gvalue;
                //System.out.println(gsVal[i][j]+ " "+"["+i+"]["+j+"]");
            }
        }
        try {
            FileWriter gv = new FileWriter("/home/vegayang/Desktop/10020015/txt/grays.txt");
            for(int i=0; i<w; i++) {
```

Figure 5.4 RGB to Grayscale Method

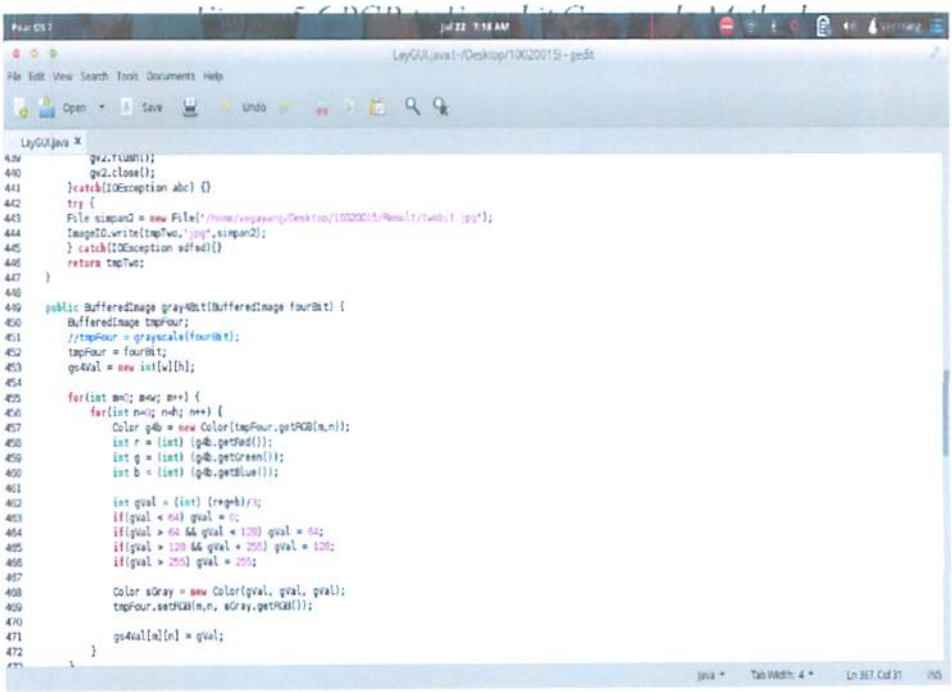
When the Button is clicked the image gray already inputed processed with the grayscale method.

5.1.3 RGB Process


Pear OS 7
Jul 22 7:18 AM
LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
LayGUI.java
Java
Tab Width: 4
Ln 367, Col 31
INS

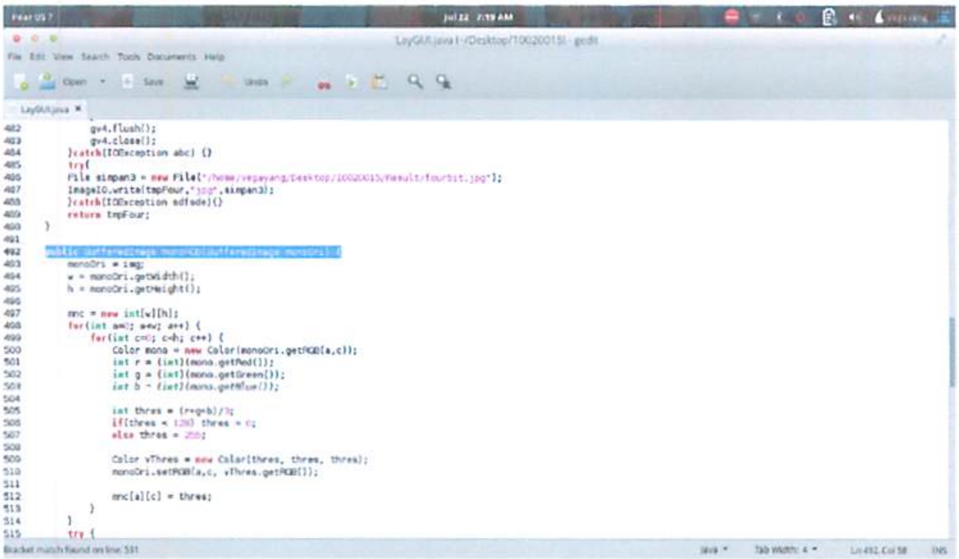
```
            ImageIO.write(grys,"jpg",A);
        }
        catch(IOException aaa){}
        return grys;
    }

    public BufferedImage gray2Bit(BufferedImage twoBit) {
        BufferedImage tmpTwo;
        tmpTwo = grayscale(twoBit);

        gs2Val = new int[w][h];

        for(int k=0; k<w; k++) {
            for(int l=0; l<h; l++) {
                Color gray = new Color(tmpTwo.getRGB(k,l));
                int r = (int) (gray.getRed());
                int g = (int) (gray.getGreen());
                int b = (int) (gray.getBlue());

                int gr = (int) (r+g+b)/3;

                if(gr < 64) gr = 0;
                if(gr < 128  && gr > 64) gr = 128;
                if(gr > 64 && gr < 255) gr = 64;
                if(gr > 255) gr = 255;

                Color grs = new Color(gr,gr,gr);
                tmpTwo.setRGB(k,l, grs.getRGB());

                gs2Val[k][l] = gr;
            }
        }
        try {
```

Figure 5.5 RGB to Two-bit Grayscale Method


Pear OS 7
Jul 22 7:18 AM
LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
LayGUI.java
Java
Tab Width: 4
Ln 367, Col 31
INS

```
            gv2.flush();
            gv2.close();
        }catch(IOException abc) {}
        try {
        File simpan2 = new File("/home/vegawang/Desktop/10020015/Result/twoBit.jpg");
        ImageIO.write(tmpTwo,"jpg",simpan2);
        } catch(IOException adfad){}
        return tmpTwo;
    }

    public BufferedImage gray4Bit(BufferedImage fourBit) {
        BufferedImage tmpFour;
        //tmpFour = grayscale(fourBit);
        tmpFour = fourBit;
        gs4Val = new int[w][h];

        for(int m=0; m<w; m++) {
            for(int n=0; n<h; n++) {
                Color g4b = new Color(tmpFour.getRGB(m,n));
                int r = (int) (g4b.getRed());
                int g = (int) (g4b.getGreen());
                int b = (int) (g4b.getBlue());

                int gVal = (int) (r+g+b)/3;
                if(gVal < 64) gVal = 0;
                if(gVal > 64 && gVal < 128) gVal = 64;
                if(gVal > 128 && gVal < 255) gVal = 128;
                if(gVal > 255) gVal = 255;

                Color sGray = new Color(gVal, gVal, gVal);
                tmpFour.setRGB(m,n, sGray.getRGB());

                gs4Val[m][n] = gVal;
            }
```

Then the inputed image processed with this 2 bit grayscale and 4 bit image.



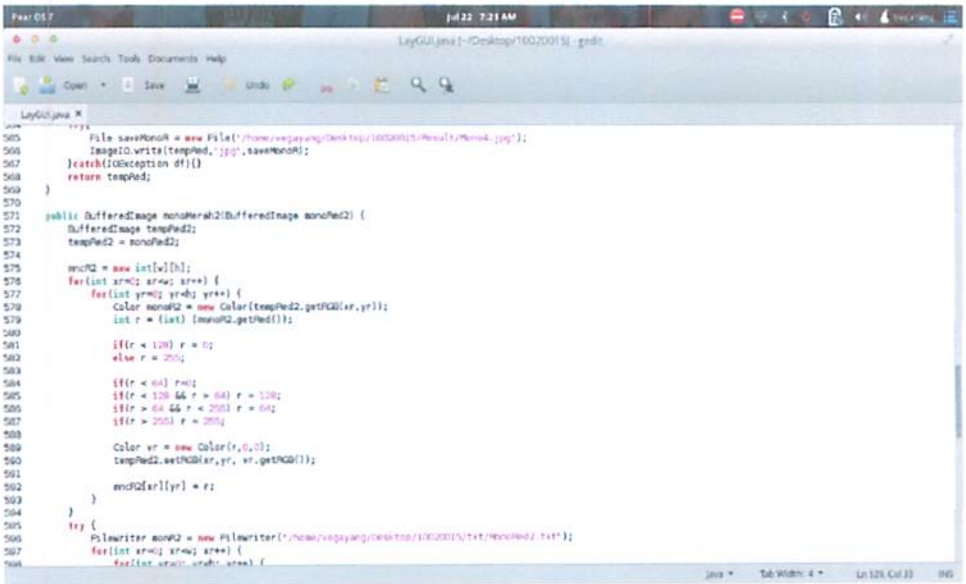
```
            gv4.flush();
            gv4.close();
        }catch(IOException abc) {}
        try{
        File simpan3 = new File("/home/vegayang/Desktop/10020015/Result/fourbit.jpg");
        ImageIO.write(tmpFour,"jpg",simpan3);
        }catch(IOException sdfsde){}
        return tmpFour;
    }

    public BufferedImage monoRGB(BufferedImage monoOri) {
        monoOri = img;
        w = monoOri.getWidth();
        h = monoOri.getHeight();

        mnc = new int[w][h];
        for(int a=0; a<w; a++) {
            for(int c=0; c<h; c++) {
                Color mono = new Color(monoOri.getRGB(a,c));
                int r = (int)(mono.getRed());
                int g = (int)(mono.getGreen());
                int b = (int)(mono.getBlue());

                int thres = (r+g+b)/3;
                if(thres < 128) thres = 0;
                else thres = 255;

                Color vThres = new Color(thres, thres, thres);
                monoOri.setRGB(a,c, vThres.getRGB());

                mnc[a][c] = thres;
            }
        }
        try {
```

Fear OS 7
Jul 22 7:19 AM
LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
LayGUI.java
Bracket match found on line: 531
Java
Tab Width: 4
Ln 492, Col 58
INS

Figure 5.7 RGB to Monochrome Image Method

In this RGB button the image will be processed into a monochrome image of the image is black and white.

5.1.4 RGB to Red Monochrome Process

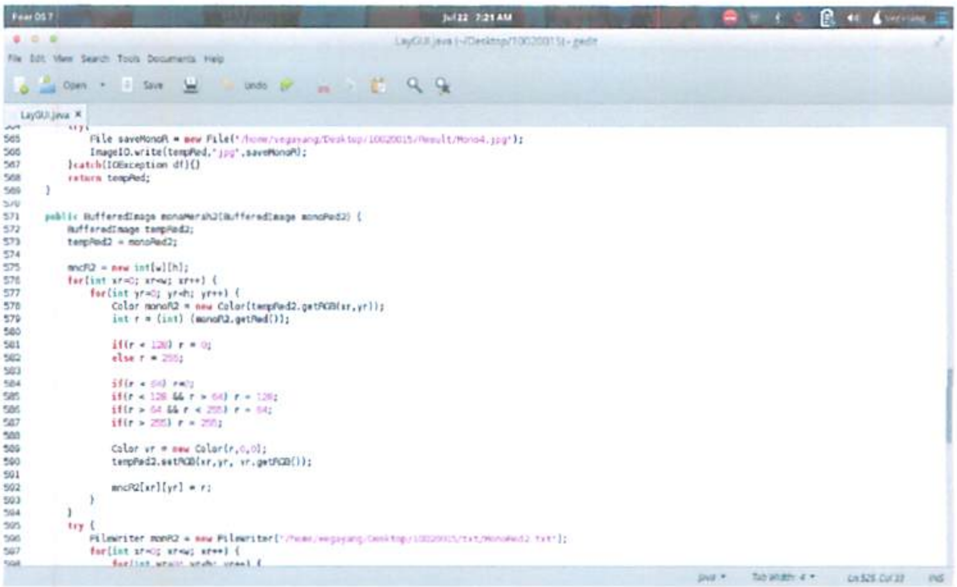


```
            File saveMonoR = new File("/home/vegayang/Desktop/10020015/Result/Mono4.jpg");
            ImageIO.write(tempRed,"jpg",saveMonoR);
        }catch(IOException df){}
        return tempRed;
    }

    public BufferedImage monoMerah2(BufferedImage monoRed2) {
        BufferedImage tempRed2;
        tempRed2 = monoRed2;

        mncR2 = new int[w][h];
        for(int xr=0; xr<w; xr++) {
            for(int yr=0; yr<h; yr++) {
                Color monoR2 = new Color(tempRed2.getRGB(xr,yr));
                int r = (int) (monoR2.getRed());

                if(r < 128) r = 0;
                else r = 255;

                if(r < 64) r=0;
                if(r < 128 && r > 64) r = 128;
                if(r > 64 && r < 255) r = 64;
                if(r > 255) r = 255;

                Color vr = new Color(r,0,0);
                tempRed2.setRGB(xr,yr, vr.getRGB());

                mncR2[xr][yr] = r;
            }
        }
        try {
            FileWriter monR2 = new FileWriter("/home/vegayang/Desktop/10020015/txt/MonoRed2.txt");
            for(int xr=0; xr<w; xr++) {
```

Fear OS 7
Jul 22 7:21 AM
LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
LayGUI.java
Java
Tab Width: 4
Ln 125, Col 33
INS

Figure 5.8 RGB to Red Monochrome Image

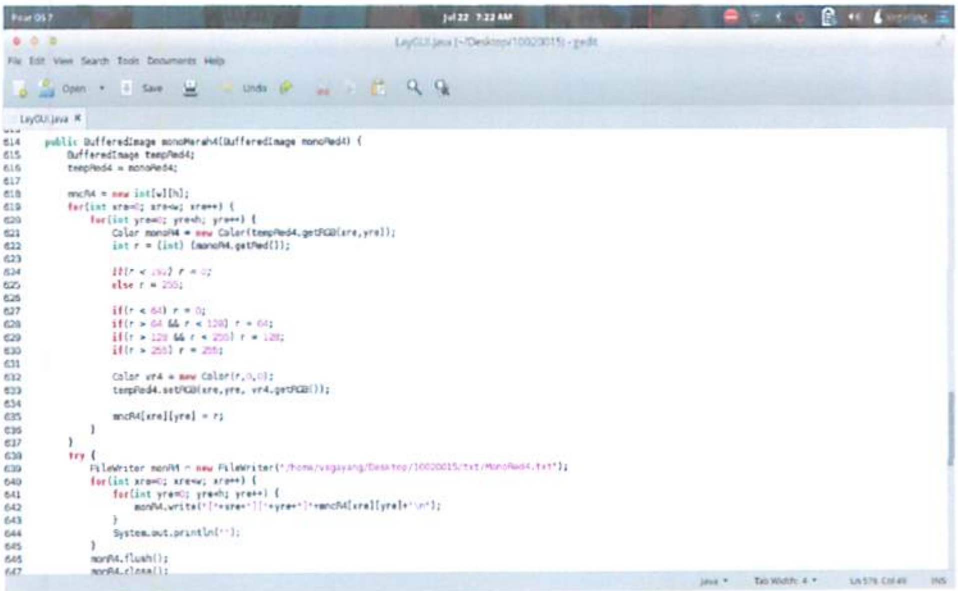
This monochrome image process value red in taken to be processed into a monochrome image, the result of that process is the image of black and red.


Fear OS 7
Jul 22 7:21 AM
LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
LayGUI.java

```
        try{
            File saveMonoR = new File("/home/vegayang/Desktop/10020015/Result/Mono4.jpg");
            ImageIO.write(tempRed,"jpg",saveMonoR);
        }catch(IOException df){}
        return tempRed;
    }

    public BufferedImage monoMerah2(BufferedImage monoRed2) {
        BufferedImage tempRed2;
        tempRed2 = monoRed2;

        mncR2 = new int[w][h];
        for(int xr=0; xr<w; xr++) {
            for(int yr=0; yr<h; yr++) {
                Color monoR2 = new Color(tempRed2.getRGB(xr,yr));
                int r = (int) (monoR2.getRed());

                if(r < 128) r = 0;
                else r = 255;

                if(r < 64) r=0;
                if(r < 128 && r > 64) r = 128;
                if(r > 64 && r < 255) r = 64;
                if(r > 255) r = 255;

                Color vr = new Color(r,0,0);
                tempRed2.setRGB(xr,yr, vr.getRGB());

                mncR2[xr][yr] = r;
            }
        }
        try {
            FileWriter monR2 = new FileWriter("/home/vegayang/Desktop/10020015/txt/MonoRed2.txt");
            for(int xr=0; xr<w; xr++) {
                for(int yr=0; yr<h; yr++) {
```

Figure 5.9 RGB to Red Monochrome Two-bits

Monochrome red image uses restriction to determinan the outcome, with 4 channel color; black, dark gray, light gray, and white value of grayscale is limited by producing the 4 colors.

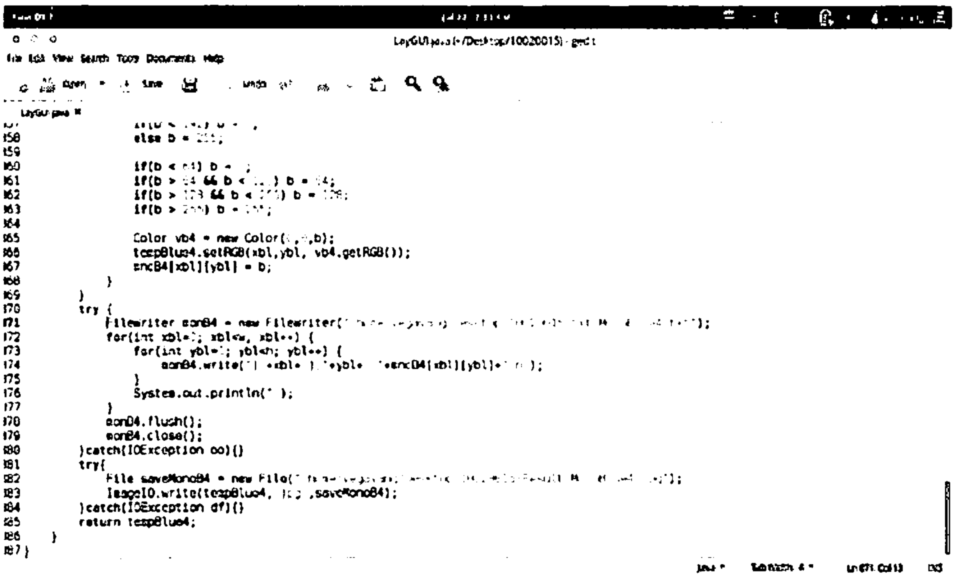

Fear OS 7
Jul 22 7:22 AM
LayGUI.java (~/Desktop/10020015) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
LayGUI.java

```
    public BufferedImage monoMerah4(BufferedImage monoRed4) {
        BufferedImage tempRed4;
        tempRed4 = monoRed4;

        mncR4 = new int[w][h];
        for(int xre=0; xre<w; xre++) {
            for(int yre=0; yre<h; yre++) {
                Color monoR4 = new Color(tempRed4.getRGB(xre,yre));
                int r = (int) (monoR4.getRed());

                if(r < 192) r = 0;
                else r = 255;

                if(r < 64) r = 0;
                if(r > 64 && r < 128) r = 64;
                if(r > 128 && r < 255) r = 128;
                if(r > 255) r = 255;

                Color vr4 = new Color(r,0,0);
                tempRed4.setRGB(xre,yre, vr4.getRGB());

                mncR4[xre][yre] = r;
            }
        }
        try {
            FileWriter monR4 = new FileWriter("/home/vegayang/Desktop/10020015/txt/MonoRed4.txt");
            for(int xre=0; xre<w; xre++) {
                for(int yre=0; yre<h; yre++) {
                    monR4.write("["+xre+"]["+yre+"]"+mncR4[xre][yre]+"\n");
                }
                System.out.println("");
            }
            monR4.flush();
            monR4.close();
```

Figure 5.10 RGB to Red Monochrome Four-bits

Monochrome red image 4 bit grayscale proceduce image sharper on the detection of its line. On the green and blue channels is also the same step her but only replace the color that will be processed only.

5.1.5 Create File TXT Process



```
                else b = 255;

                if(b < 64) b = 0;
                if(b > 64 && b < 128) b = 64;
                if(b > 128 && b < 192) b = 128;
                if(b > 192) b = 255;

                Color vb4 = new Color(0,0,b);
                tempBlue4.setRGB(xbl,ybl, vb4.getRGB());
                encB4[xbl][ybl] = b;
            }
        }
        try {
            FileWriter monB4 = new FileWriter("[illegible]");
            for(int xbl=0; xbl<w, xbl++) {
                for(int ybl=0; ybl<h; ybl++) {
                    monB4.write("[" +xbl+ "][" +ybl+ "]" +encB4[xbl][ybl]+ "\n");
                }
                System.out.println(" ");
            }
            monB4.flush();
            monB4.close();
        }catch(IOException oo){}
        try{
            File saveMonoB4 = new File("[illegible]");
            ImageIO.write(tempBlue4, "jpg" ,saveMonoB4);
        }catch(IOException df){}
        return tempBlue4;
    }
}
```

Figure 5.11 Process Create File TXT

Then in each process grayscale 2 bits and 4 bit, thresholding rgb 2-bits and 4 bit, value on the image are stored in the txt file.

5.2 Testing

5.2.1 Show Button Process

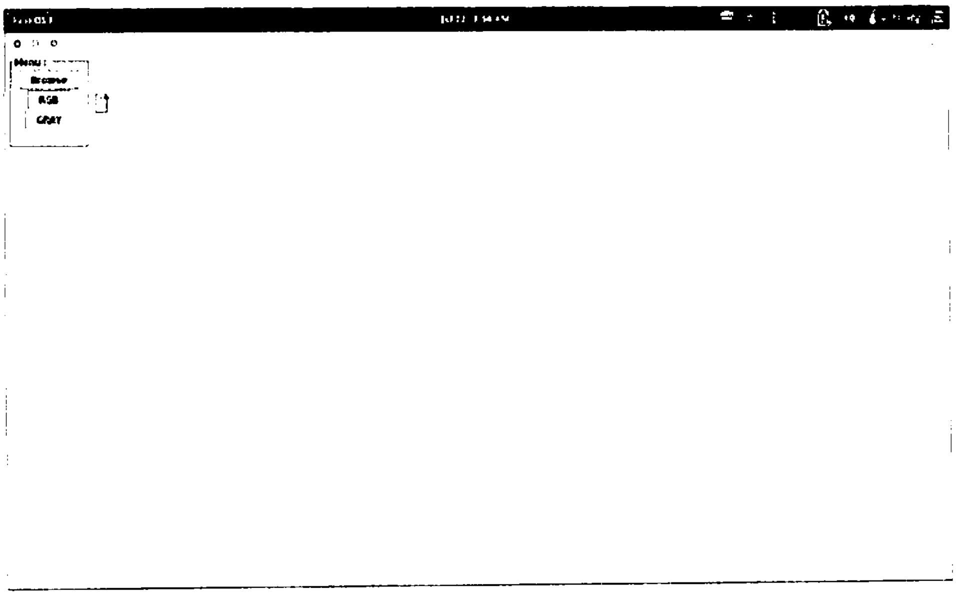


Figure 5.12 Show Button Process Image

Initial display on the application of this is there are three buttons that can be selected, browse, rgb and grayscale.

5.2.2 Show File Directory

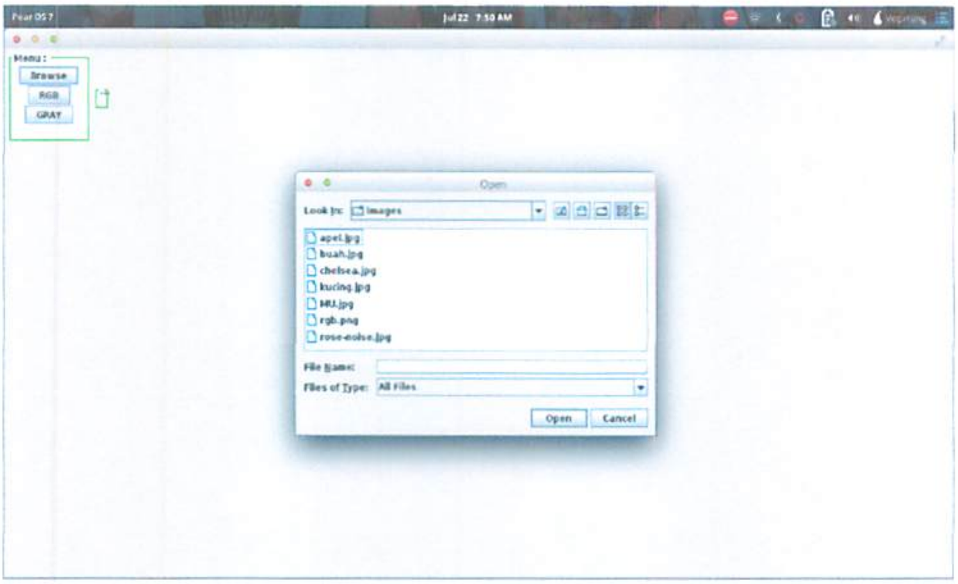


Figure 5.13 Show File Directory

Then if it is clicked the browse button it will pop up a screen file directory to select the image to be processed.

5.2.3 Show Result Browse Image

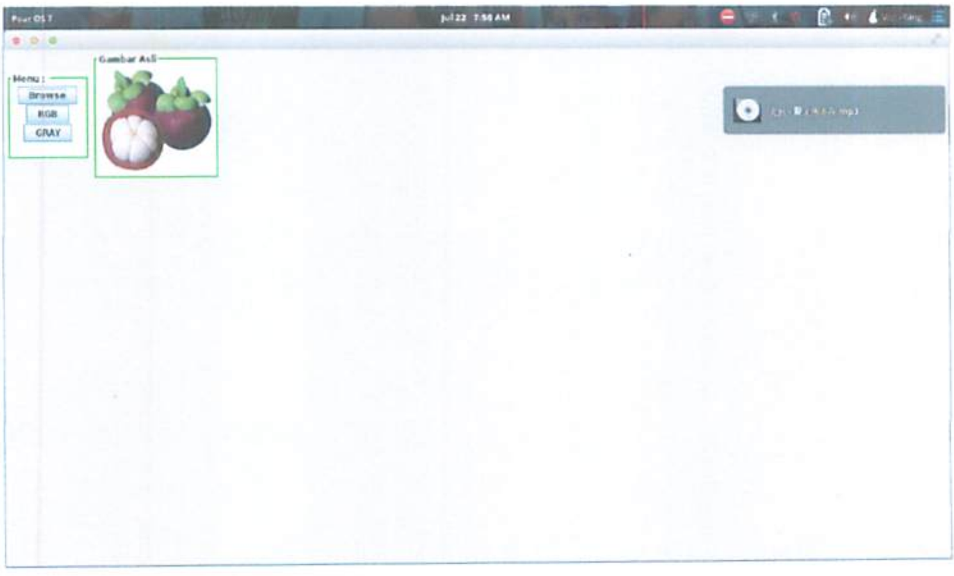


Figure 5.14 Show Result Browse Image

After the image has been selected then the image will appear on the screen button above, the image is in border titled.

5.2.4 Show Grayscale Image

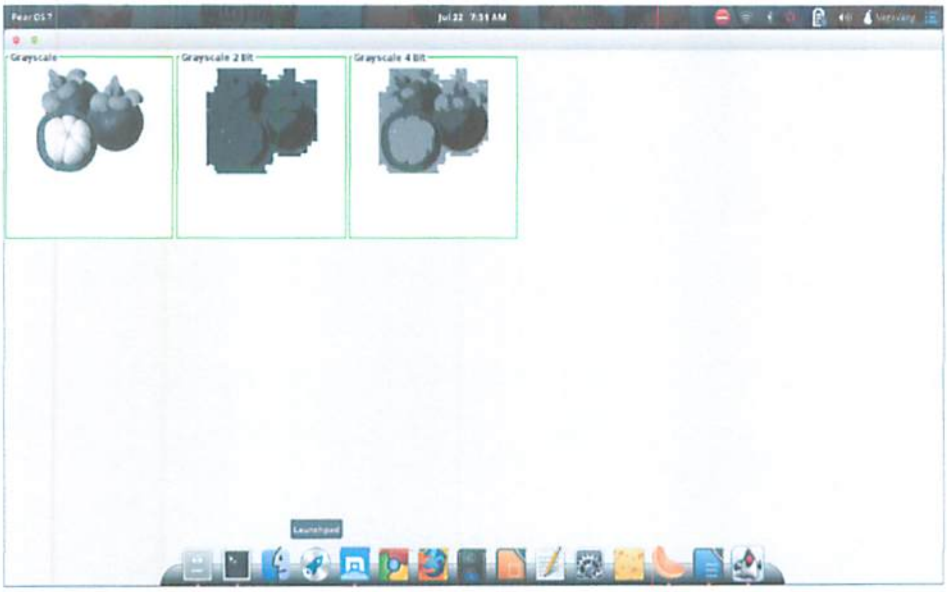


Figure 5.15 Show Result Grayscale Image

When the image you selected appears on the screen button, and the button grayscale is selected it will pop up a new frame that displays all results process grayscale 2 bit and 4 bits.

5.2.5 Show RGB Image

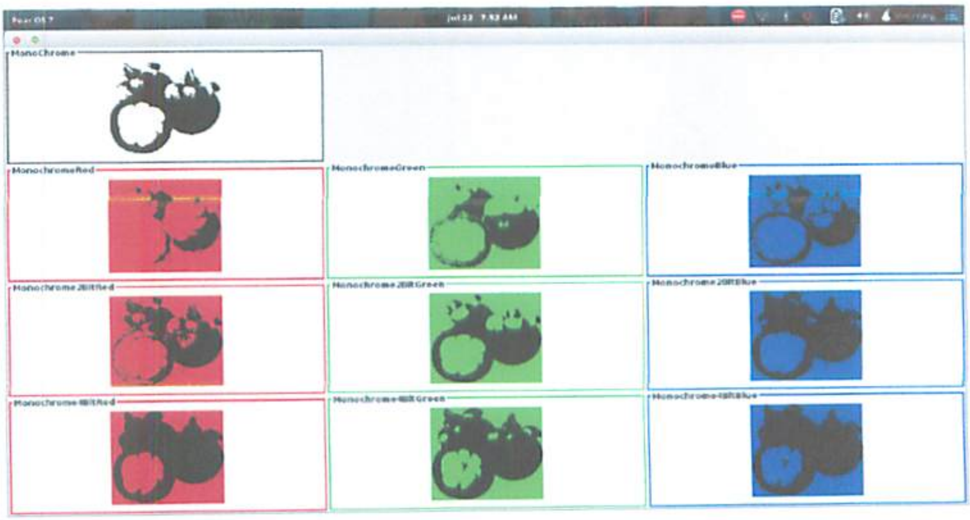


Figure 5.16 Show Result RGB Image

The image is already selected, and then the rgb button is clicked it will pop up a new frame to display the image processed rgb thresholding.

Tabel 5.1 Table Value of Grayscale

X,Y	[7][32]	[7][33]	[7][34]	[7][35]	[7][36]
Grayscale	253	253	253	253	253
Two-bits	64	64	64	64	64
Four-Bit	128	128	128	128	128

In the table above is the result value of grayscale, 2bit and 4 bits, sample taken on the x,y start [7][32] of five samples. So you can see the result of differences in the image three. Differences in value are different at certain positions, but it is certain positions it can also have the same value. In fact many once stored on the txt file that is as big as the picture, but it is not possible to input on this chart so taken as many as five samples only.

Table 5.2 Table Value of Red Monochrome

X,Y	[3][83]	[3][84]	[3][85]	[3][86]	[3][87]
Monochrome Red	255	255	255	255	255
Monochrome Red 2-bit	255	255	255	0	0
Monochrome Red 4-bit	255	255	255	255	255

In the table above, the comparasion of the value of the monochrome red, 2bits and 4 bits, it can be seen that in certain positions have the same value and different values above 255 is the value of her red so the image arising out of black and red, the colors of black on the image while the dominant red color on image.

Table 5.3 Table Value of Green Monochrome

X,Y	[3][85]	[3][86]	[3][87]	[3][88]	[3][89]
Monochrome Green	255	255	255	0	0
Monochrome Green 2-bit	0	0	0	0	0
Monochrome Green 4-bit	0	0	0	0	0

In the table above, the comparison of the value of the monochrome green, 2bits and 4 bit, it can be seen that in certain positions have the same value and different, values above 255 is the value of her green so the image arising out of black and green. The colors of black on the image while the green color is dominat on the same image with a monochrome green.

Table 5.4 Table Value of Blue Monochrome

X,Y	[8][75]	[8][76]	[8][77]	[8][78]	[8][79]
Monochrome Blue	0	0	0	255	255
Monochrome Blue 2-bit	0	0	0	0	0
Monochrome Blue 4-bit	0	0	0	0	0

In the table above, the value that is in monochrome red, green, blue, all almost the same just that the differences lie in certain positions. The value that is used for 0 and 255 (channel value). If one of the colors from the rgb is used then the color will be the color of the dominant.