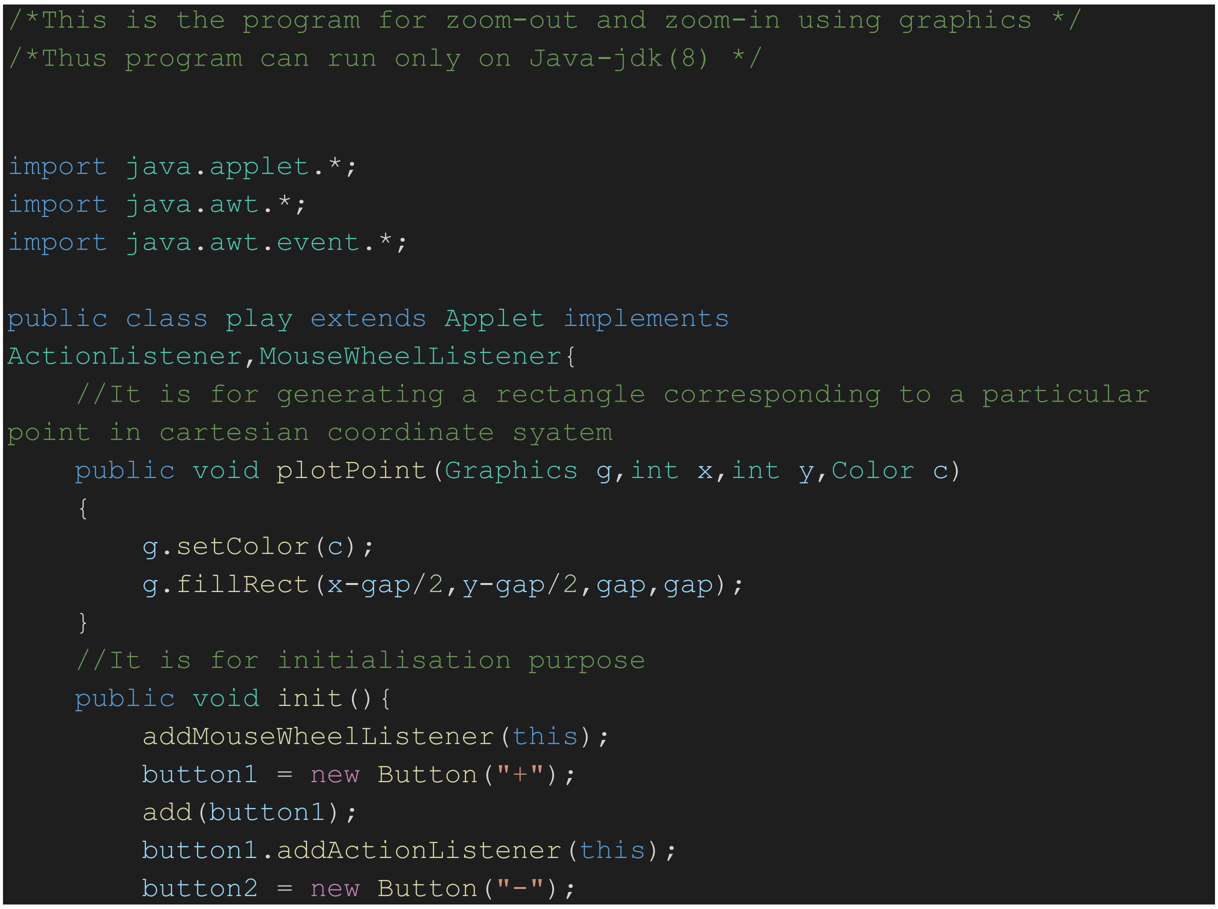
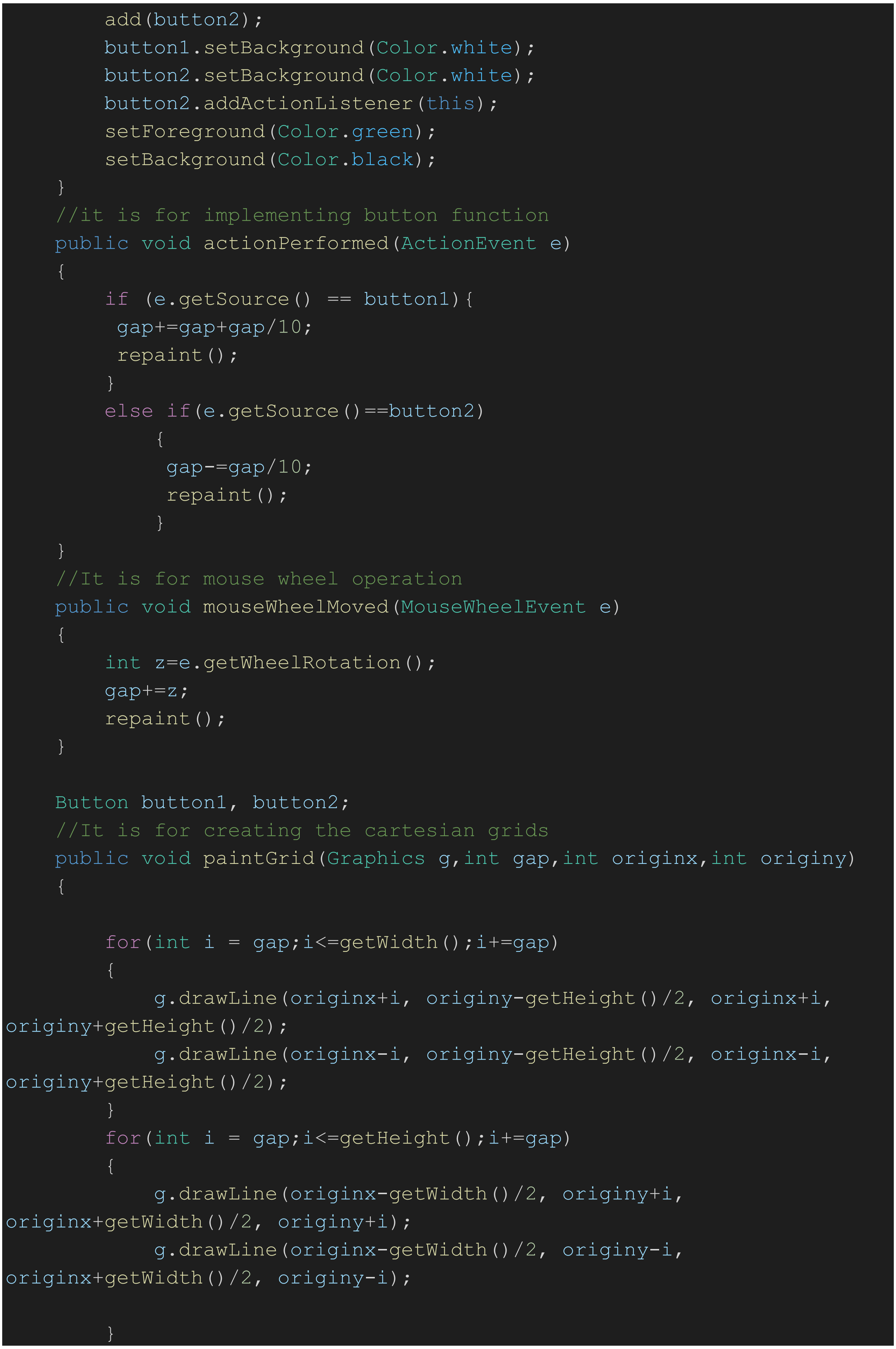
ASSIGNMENT-1

NAME-ANUBHAV ANAND
ENROLLMENT NUMBER-2020CSB102
SUBJECT-COMPUTER GRAPHICS(Assignment-1)
G-SUITE ID-2020csb102.anubhav@students.iiests.ac.in

Test.java-



```
/*This is the program for zoom-out and zoom-in using graphics */
/*Thus program can run only on Java-jdk(8) */


import java.applet.*;
import java.awt.*;
import java.awt.event.*;

public class play extends Applet implements
ActionListener,MouseWheelListener{
    //It is for generating a rectangle corresponding to a particular
point in cartesian coordinate syatem
    public void plotPoint(Graphics g,int x,int y,Color c)
    {
        g.setColor(c);
        g.fillRect(x-gap/2,y-gap/2,gap,gap);
    }
    //It is for initialisation purpose
    public void init(){
        addMouseWheelListener(this);
        button1 = new Button("+");
        add(button1);
        button1.addActionListener(this);
        button2 = new Button("-");
```



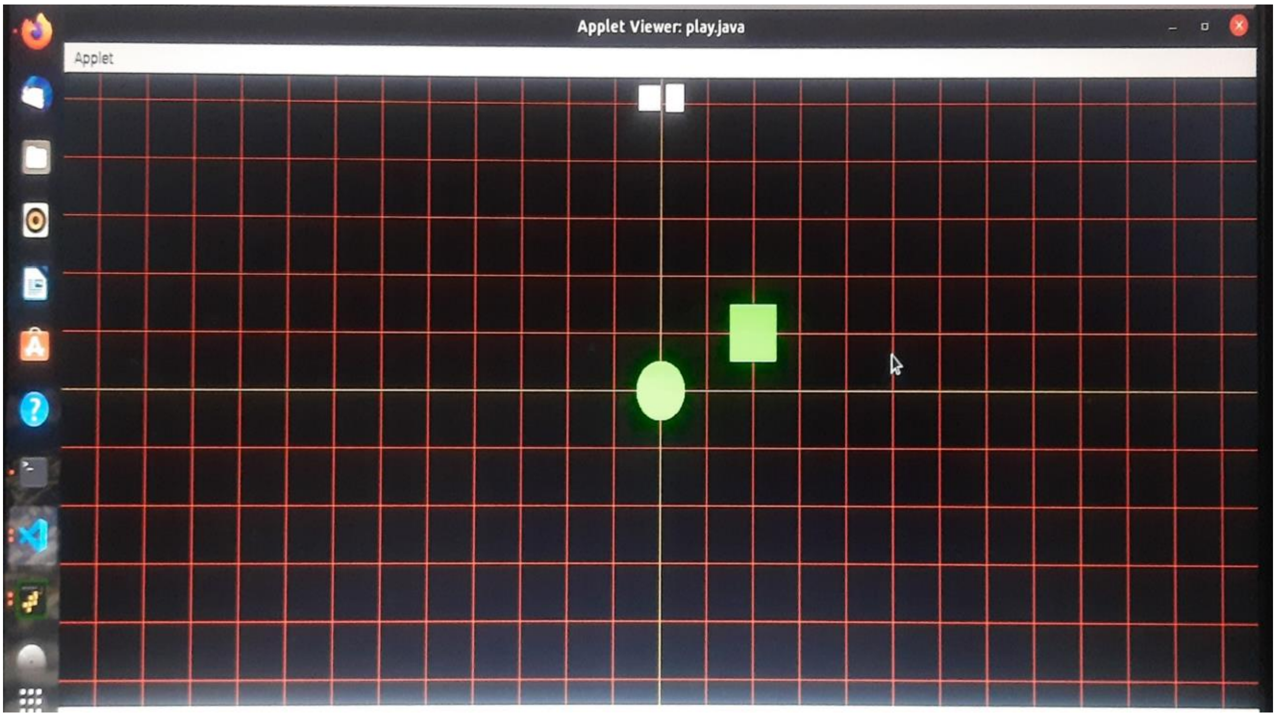
```
        add(button2);
        button1.setBackground(Color.white);
        button2.setBackground(Color.white);
        button2.addActionListener(this);
        setForeground(Color.green);
        setBackground(Color.black);
    }
    //it is for implementing button function
    public void actionPerformed(ActionEvent e)
    {
        if (e.getSource() == button1){
         gap+=gap+gap/10;
         repaint();
        }
        else if(e.getSource()==button2)
            {
             gap-=gap/10;
             repaint();
            }
    }
    //It is for mouse wheel operation
    public void mouseWheelMoved(MouseWheelEvent e)
    {
        int z=e.getWheelRotation();
        gap+=z;
        repaint();
    }

    Button button1, button2;
    //It is for creating the cartesian grids
    public void paintGrid(Graphics g,int gap,int originx,int originy)
    {

        for(int i = gap;i<=getWidth();i+=gap)
        {
            g.drawLine(originx+i, originy-getHeight()/2, originx+i,
originy+getHeight()/2);
            g.drawLine(originx-i, originy-getHeight()/2, originx-i,
originy+getHeight()/2);
        }
        for(int i = gap;i<=getHeight();i+=gap)
        {
            g.drawLine(originx-getWidth()/2, originy+i,
originx+getWidth()/2, originy+i);
            g.drawLine(originx-getWidth()/2, originy-i,
originx+getWidth()/2, originy-i);

        }
    }

    int gap = 50;
    //It ia a normal paint function to call other functions to generate
the graphics in applet
    public void paint(Graphics g){

            g.setColor(Color.orange);
            int originx=getX()+getWidth()/2;
            int originy=getY()+getHeight()/2;
            g.drawLine(originx-getWidth()/2, originy,
originx+getWidth()/2, originy);
            g.drawLine(originx, originy-getHeight()/2, originx,
originy+getHeight()/2);
            paintGrid(g,gap,originx,originy);
            Color c=new Color(100,100,100);
            plotPoint(g,originx+(gap*2),originy-(gap*1),c);
            g.fillOval(originx-25,originy-25,50,50);
            // plotPoint(g,originx,originy,Color.red);
            // g.clearRect(0,0,getHeight(),getWidth());

            // try{Thread.sleep(3000);}
            // catch (InterruptedException ie){ie.printStackTrace();}
    }
}
```

Test.html-

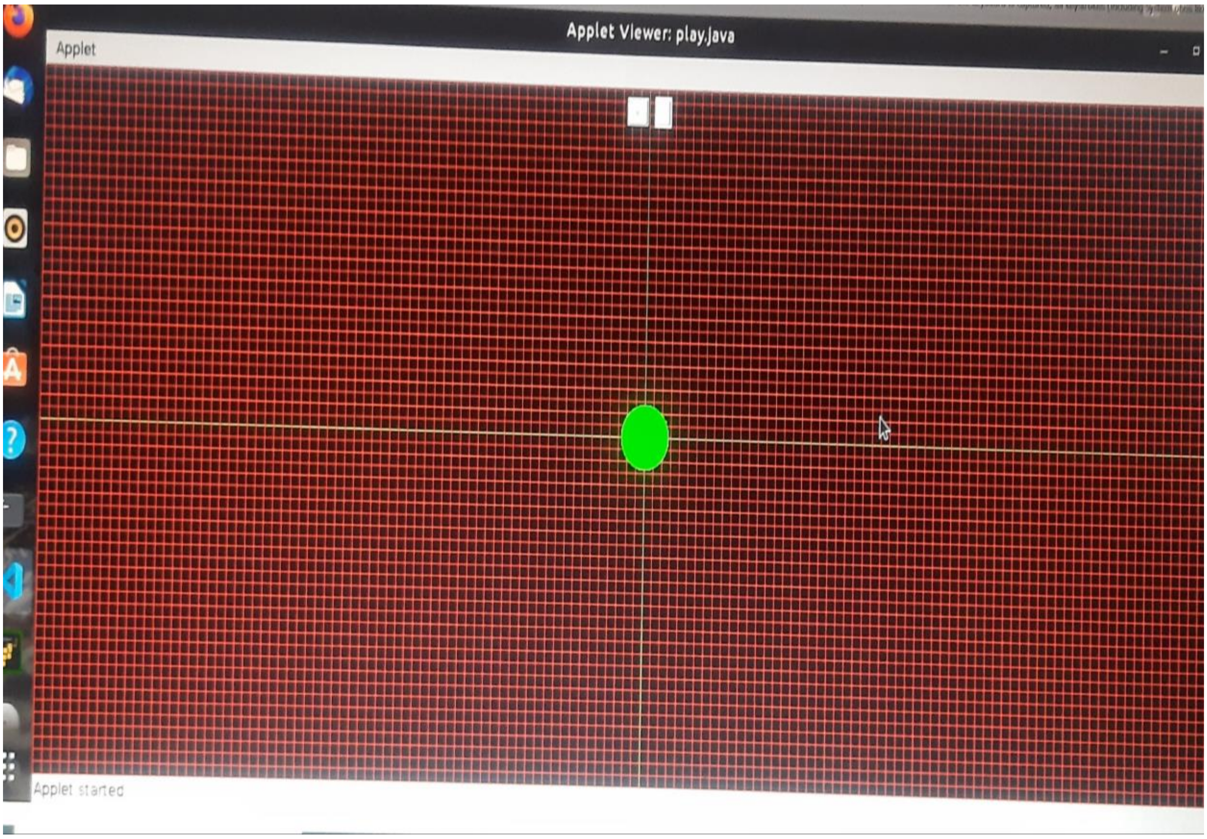
```
<html>
    <head></head>
    <body>
        <applet code ="play.java"  height="500" width="500"></applet>
    </body>
</html>
```

APPLET VIEW-

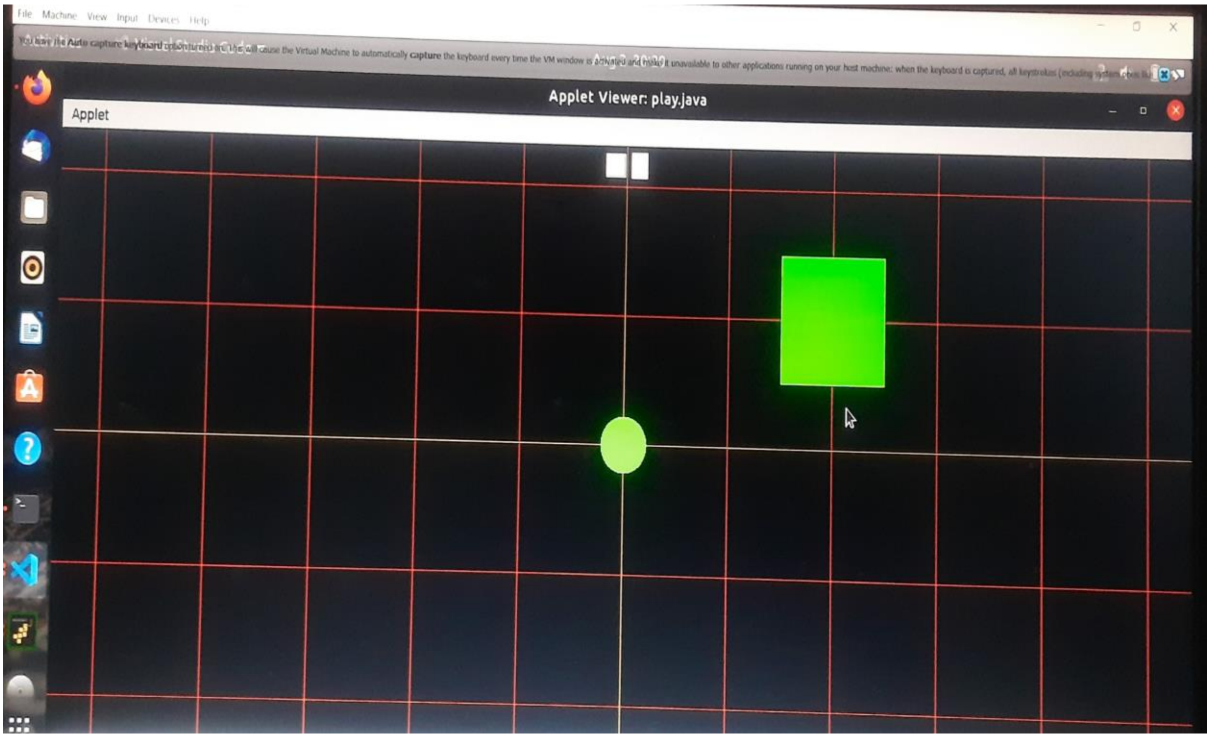
NORMAL VIEW-



ZOOM-OUT->



ZOOM-IN->



You have the Auto capture keyboard option turned on. This will cause the Virtual Machine to automatically capture the keyboard every time the VM window is activated and make it unavailable to other applications running on your host machine: when the keyboard is captured, all keystrokes (including system ones li