College of Engineering, Thiruvananthapuram

Object Oriented Programming Lab

Anirudh A. V.

S2 CSE R2, Roll No. 11

Department of Computer Science
& Engineering

March 10, 2022

1 **Drawing Shapes**

1.1 Aim

To Write a Java program that allows the user to draw lines, rectangles and ovals.

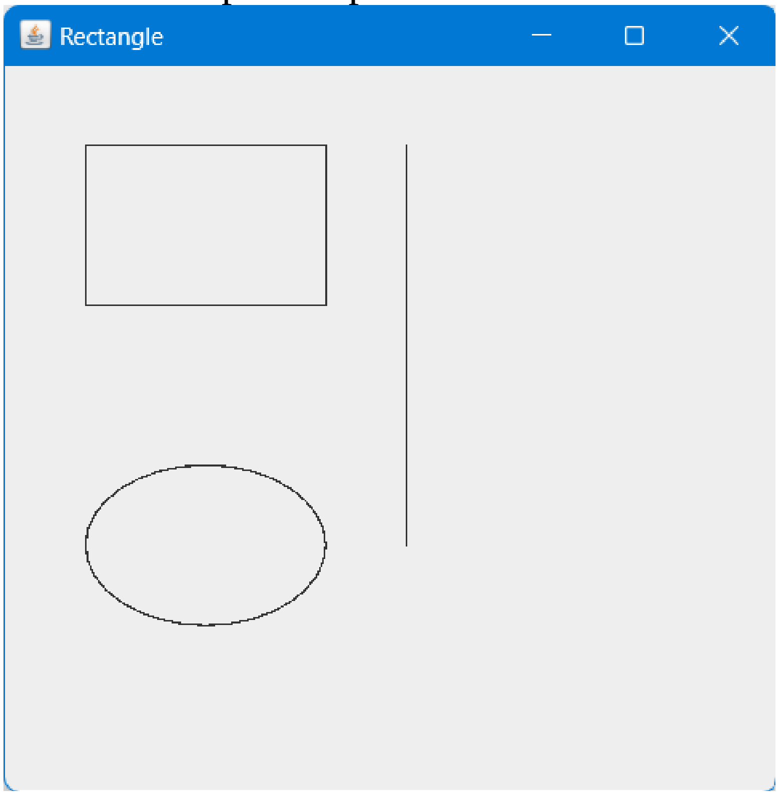
1.2 Algorithm

```
Step - 1   Start
Step - 2   Create a ‘mycanvas’ class extending JComponent, in which
 define a paint method with a Graphics as argument.
Step - 3   Inside the paint method, initialise Rectangles, Lines, and Ovals
 of required size.
Step - 4   Create a ‘Rectangle’ class as the main class that extends JFrame.
Step - 5   Create an instance of JFrame and set appropriate size and
 properties for it.
Step - 6   Add an anonymous object of ‘mycanvas’ class to the Frame.
Step - 7   Stop
```

1.3 Code

```java
import javax.swing.*;
import java.awt.*;

class mycanvas extends JComponent {
    public void paint(Graphics g){
        g.drawRect(40, 40, 120, 80);
        g.drawOval(40, 200, 120, 80);
        g.drawLine(200, 40, 200, 240);
    }

}

public class Rectangle extends JFrame {
    public static void main(String[] args) {
        JFrame F = new JFrame("Rectangle");
        F.setDefaultCloseOperation(EXIT_ON_CLOSE);
        F.setSize(400, 400);
        F.add(new mycanvas());
        F.setVisible(true);
    }
}
```

1.4 Sample Output



2 Mouse Events

2.1 Aim

To Write a Java program for handling mouse events.

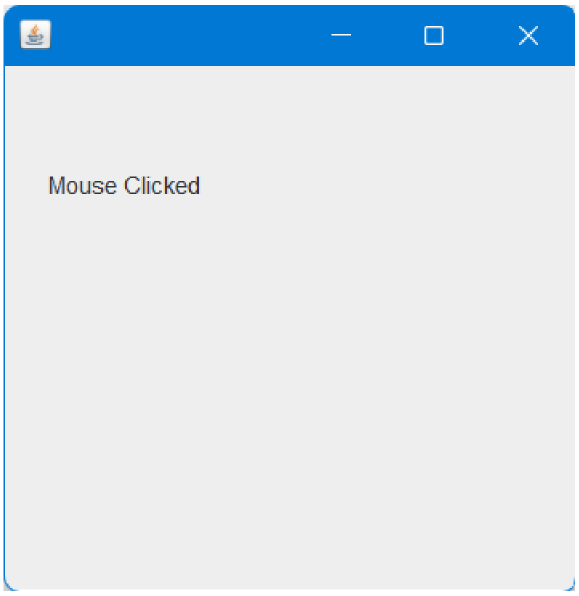
2.2 Algorithm

```
Step - 1  Start
Step - 2  Create a class ‘Mouseevent’ that inherits JFrame and
          implements MouseListener.
Step - 3  Create an instance of JFrame and set appropriate size and
          properties for it.
Step - 4  Add a JLabel ‘l’ to the frame. Add mouse listener to the
          frame.
Step - 5  If mouse enters the frame print ‘Mouse entered’ in the label.
Step - 6  If mouse button is pressed, print ‘Mouse pressed’ in the
          label.
Step - 7  If mouse button is released, print ‘Mouse released’ in the
          label
Step - 8  If mouse leaves the frame print ‘Mouse exited’ in the label
Step - 9  Stop
```

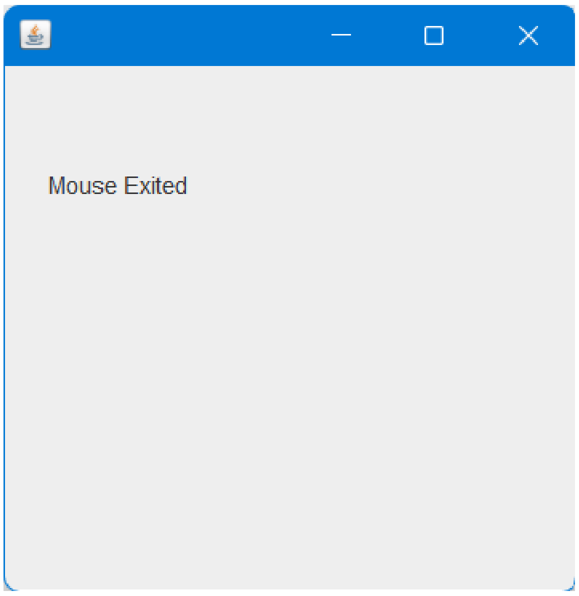
2.3 Code

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Mouseevent extends JFrame implements MouseListener {
    Label l;

    Mouseevent() {
        addMouseListener(this);

        l = new Label();
        l.setBounds(20, 50, 100, 20);
        add(l);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 300);
        setLayout(null);
        setVisible(true);
    }

    public void mouseClicked(MouseEvent e) {
        l.setText("Mouse Clicked");
    }

    public void mouseEntered(MouseEvent e) {
        l.setText("Mouse Entered");
    }

    public void mouseExited(MouseEvent e) {
        l.setText("Mouse Exited");
    }

    public void mousePressed(MouseEvent e) {
        l.setText("Mouse Pressed");
    }

    public void mouseReleased(MouseEvent e) {
        l.setText("Mouse Released");
    }

    public static void main(String[] args) {
        new Mouseevent();
    }
}
```

2.4 Sample Output



```
Mouse Clicked
```



```
Mouse Exited
```

3 Adapter Classes

3.1 Aim

To Write a Java program for handling key events using Adapter classes.

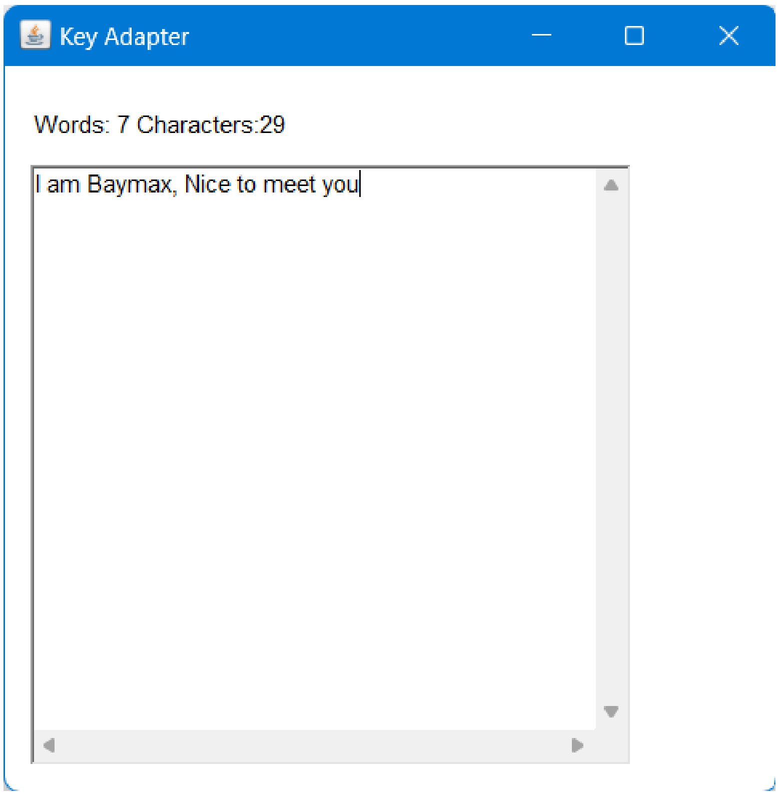
3.2 Algorithm

Step - 1 Start
Step - 2 Create a main class ‘Adapter’ that extends KeyAdapter
Step - 3 Create an instance of a Frame and set appropriate size and properties
Step - 4 Add a text are to the frame object
Step - 5 In the KeyReleased method, define the function for counting no. of words and no. of characters and set them to the Label.
Step - 6 Add Label to the Frame object.
Step - 7 Stop

3.3 Code

```
import java.awt.*;
import java.awt.event.*;

public class Adapter extends KeyAdapter {
    Label l;
    TextArea area;
    Frame f;

    Adapter() {
        f = new Frame("Key Adapter");
        l = new Label();
        l.setBounds(20, 50, 200, 20);
        area = new TextArea();
        area.setBounds(20, 80, 300, 300);
        area.addKeyListener(this);

        f.add(l);
        f.add(area);
        f.setSize(400, 400);

        f.addWindowListener (new WindowAdapter() {
            public void windowClosing (WindowEvent e) {
                f.dispose();
            }
        });

        f.setLayout(null);
        f.setVisible(true);
    }

    public void keyReleased(KeyEvent e) {
        String text = area.getText();
        String words[] = text.split(" ");
        l.setText("Words: " + words.length + " Characters:" + text.length());
    }

    public static void main(String[] args) {
        new Adapter();
    }
}
```

3.4 Sample Output



```
Words: 7 Characters:29
I am Baymax, Nice to meet you
```

4 Waveform

4.1 Aim

To Write a Java Swing program to print a wave form on the output screen

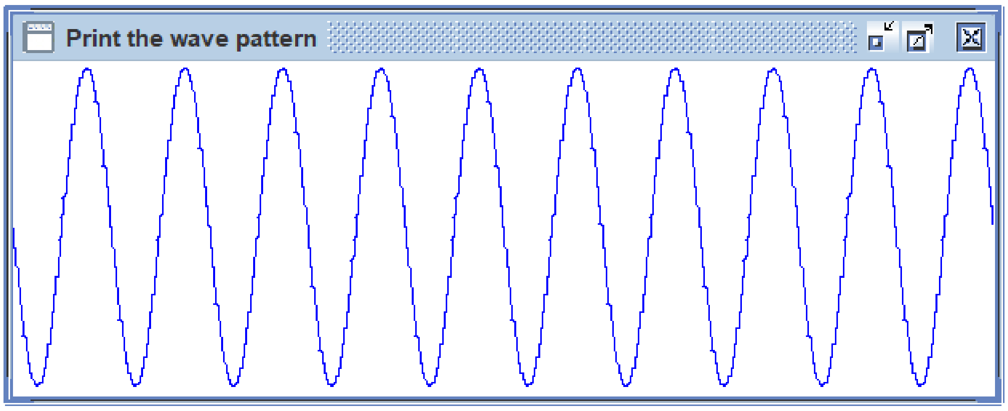
4.2 Algorithm

```
Step - 1  Start
Step - 2  Create a main class ‘WavePrint’ that extends JPanel
Step - 3  Declare necessary data members for waveform building.
Step - 4  Define a method ‘setCycles’ which assigns values for the
          data members of the class and sets the no. of cycles.
Step - 5  In the public method paintComponent, define the function
          for forming the waveform by passing a Graphics argument.
Step - 6  Create a JFrame instance and add the JPanel to it.
Step - 7  Set appropriate size and properties for the frame instance.
Step - 8  Stop
```

4.3 Code

```java
import java.awt.*;
import javax.swing.*;

public class WavePrint extends JPanel
{
    int SCALEFACTOR = 200;
    int cycles;
    int points;
    double[] sines;
    int[] pts;

    public  void setCycles(int cycles)
    {
        this.cycles = cycles;
        this.points = SCALEFACTOR * cycles * 2;
        this.sines = new double[points];
        for (int i = 0; i < points; i++)
        {
            double radians = (Math.PI / SCALEFACTOR) * i;
            this.sines[i] = Math.sin(radians);
        }
    }
    public void paintComponent(Graphics g)
    {
        int maxWidth = getWidth();
        double hstep = (double) maxWidth / (double) points;
        int maxHeight = getHeight();
        pts = new int[points];
        for (int i = 0; i < points; i++)
        {
            pts[i] = (int) (sines[i] * maxHeight / 2 * .95 + maxHeight / 2);
        }
        g.setColor(Color.BLUE);
        for (int i = 1; i < points; i++)
        {
            int x1 = (int) ((i - 1) * hstep);
            int x2 = (int) (i * hstep);
            int y1 = pts[i - 1];
            int y2 = pts[i];
            g.drawLine(x1, y1, x2, y2);
        }
    }

    public static void main(String[] args)
    {
        JFrame.setDefaultLookAndFeelDecorated(true);
        JFrame frame = new JFrame("Print the wave pattern");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setBackground(Color.white);
        frame.setSize(500, 200);

        WavePrint sw = new WavePrint();
        sw.setCycles(10);
        frame.add(sw);
        frame.setVisible(true);
    }
}
```

4.4 Sample Output



5 Calculator

5.1 Aim

To Write a Java program that works as a simple calculator. Arrange Buttons for digits and the + - * % operations properly. Add a text field to display the result. Handle any possible exceptions like divide by zero. Use Java Swing

5.2 Algorithm

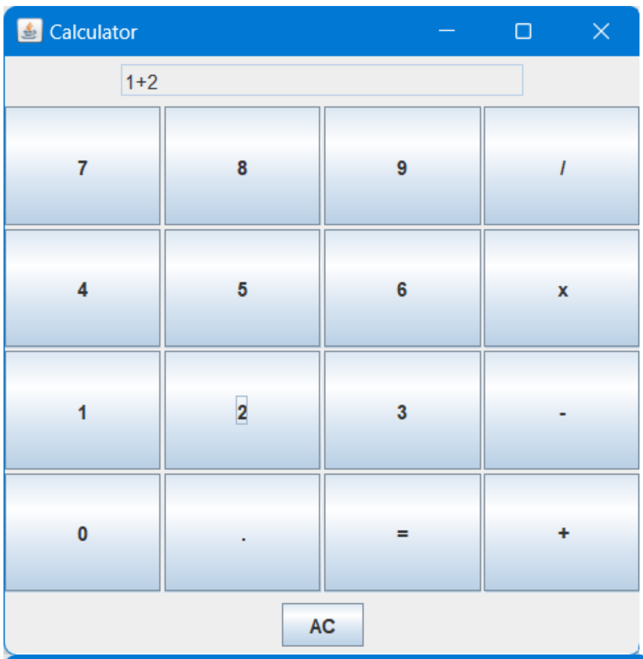
Step - 1 START
Step - 2 Create a java frame inside the constructor by extending the frame class and create button b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, bp, bs, bm, bd, be, bc as buttons in the calculator and a textfield txtf for display
Step - 3 Let str1 and str2 be two empty strings and let f be a Boolean value initialized to true
Step - 4 Register the listeners of each button with the event source
Step - 5 In the event handling method, if buttons corresponding to any of the numbers is presses and if f=true add the number corresponding to the button to str1 else add it to str2
Step - 6 If button corresponding to any operator is pressed, set f=false and let the operator be stored in op as a character
Step - 7 If button be is pressed, convert str1 and str2 to integers n1 and n2 and do the suitable operation using the operator stored in op and print it to the textfield and set str1 as the current answer, set f=true and set str2 to empty string
Step - 8 Enclose the operation in a try block and catch the exceptions and handle it suitably
Step - 9 If button bc is pressed, empty the textfield and set str1 and str2 to empty string and f to true
Step - 10 Call the constructor of the class in the main method
Step - 11 STOP

5.3 Code

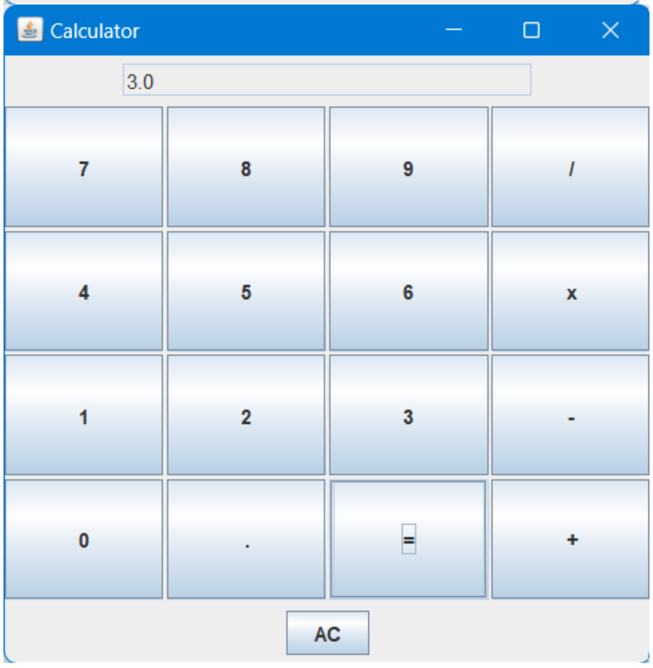
```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
```

```java
public class Calculator extends JFrame implements ActionListener {
    static JFrame f;
    static JButton b1, b2, b3, b4, b5, b6, b7, b8, b9, b0, dot;
    static JButton plus, minus, mult, divide, clear, equals;
    static JTextField t1;
    static String s0, s1, s2;

    Calculator() {
        s1 = s2 = s0 = "";
    }

    public static void main(String[] args) {
        Calculator c = new Calculator();

        JPanel p1 = new JPanel();
        JPanel p2 = new JPanel();
        JPanel p3 = new JPanel();

        f = new JFrame("Calculator");

        b1 = new JButton("1");
        b2 = new JButton("2");
        b3 = new JButton("3");
        b4 = new JButton("4");
        b5 = new JButton("5");
        b6 = new JButton("6");
        b7 = new JButton("7");
        b8 = new JButton("8");
        b9 = new JButton("9");
        b0 = new JButton("0");
        dot = new JButton(".");
        plus = new JButton("+");
        minus = new JButton("-");
        mult = new JButton("x");
        divide = new JButton("/");
        equals = new JButton("=");
        clear = new JButton("AC");

        t1 = new JTextField(24);
        t1.setEditable(false);

        b1.addActionListener(c);
        b2.addActionListener(c);
        b3.addActionListener(c);
        b4.addActionListener(c);
        b5.addActionListener(c);
        b6.addActionListener(c);
        b7.addActionListener(c);
        b8.addActionListener(c);
        b9.addActionListener(c);
        b0.addActionListener(c);
        dot.addActionListener(c);
        plus.addActionListener(c);
        minus.addActionListener(c);
        mult.addActionListener(c);
        divide.addActionListener(c);
        equals.addActionListener(c);
        clear.addActionListener(c);

        p1.add(t1);

        p2.add(b7);
        p2.add(b8);
        p2.add(b9);
        p2.add(divide);

        p2.add(b4);
        p2.add(b5);
        p2.add(b6);
        p2.add(mult);

        p2.add(b1);
        p2.add(b2);
        p2.add(b3);
        p2.add(minus);

        p2.add(b0);
        p2.add(dot);
        p2.add(equals);
        p2.add(plus);

        p3.add(clear);

        p2.setLayout(new GridLayout(4, 4, 2, 2));
        // p1.setBounds(20, 10, 350, 30);
        f.add(p1, BorderLayout.NORTH);
        f.add(p2);
        f.add(p3, BorderLayout.SOUTH);

        f.setDefaultCloseOperation(EXIT_ON_CLOSE);
        f.setSize(400, 400);
        f.setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        String s = e.getActionCommand();
        if ((s.charAt(0) >= '0' && s.charAt(0) <= '9') || s.charAt(0) == '.')
    {

            if (!s1.equals(""))
                s2 = s2 + s;
            else
                s0 = s0 + s;

            t1.setText(s0 + s1 + s2);

        } else if (s.equals("AC")) {

            s0 = s1 = s2 = "";
            t1.setText(s0 + s1 + s2);

        } else if (s.charAt(0) == '=') {

            double result;

            if (s1.equals("+"))
                result = Double.parseDouble(s0) + Double.parseDouble(s2);
            else if (s1.equals("-"))
                result = Double.parseDouble(s0) - Double.parseDouble(s2);
            else if (s1.equals("x"))
                result = Double.parseDouble(s0) * Double.parseDouble(s2);
            else
                result = Double.parseDouble(s0) / Double.parseDouble(s2);

            s0 = Double.toString(result);
            t1.setText(s0);
            s1 = s2 = "";
        }
        else {
            if (s1.equals("") || s2.equals("")) {
                s1 = s;
            } else {
                double result;

            if (s1.equals("+"))
                result = Double.parseDouble(s0) + Double.parseDouble(s2);
            else if (s1.equals("-"))
                result = Double.parseDouble(s0) - Double.parseDouble(s2);
            else if (s1.equals("x"))
                result = Double.parseDouble(s0) * Double.parseDouble(s2);
            else
                result = Double.parseDouble(s0) / Double.parseDouble(s2);
            s0 = Double.toString(result);
            s1 = s;
            s2 = "";
            }
            t1.setText(s0+s1+s2);
        }
    }
}
```

5.4 Sample Output



1+2



3.0

6 Traffic Lights – Radio Buttons

6.1 Aim

To Write a Java program that simulates a traffic light. The program lets the user select one of three lights: red, yellow, or green. When a radio button is selected, the light is turned on, and only one light can be on at a time. No light is on when the program starts.

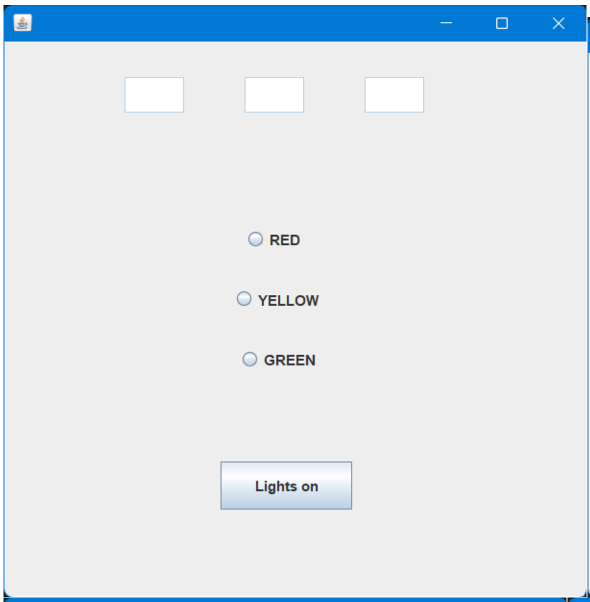
6.2 Algorithm

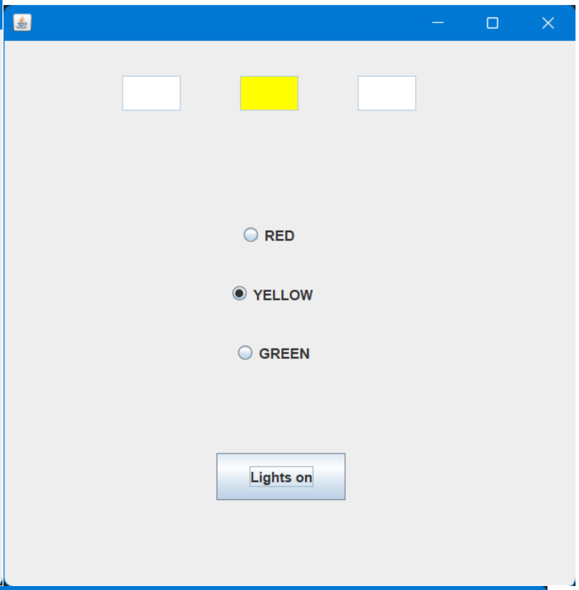
```
1) START
2) Create a java frame inside the constructor and create three radio
buttons jr1, jr2, jr3 with labels Red, Yellow, Green and let
x=0, y=0, z=0
3) Register each listener with the source and add the components to
the frame
4) In the event handling method of itemstatechange, if the jr1 is
pressed let x=1 and call the paint method. If the jr2 is pressed let y=1
and call the paint method. If the jr3 is pressed let z=1 and call the
paint method.
5) Inside the paint method, draw three circles of radius 60
6) If x=1, then set the colour of first circle to red and the other to
white and set x=0
7) If y=1, then set the colour of second circle to yellow and the other
to white and set y=0
8) If z=1, then set the colour of third circle to green and the other to
white and set z=0
9) STOP
```

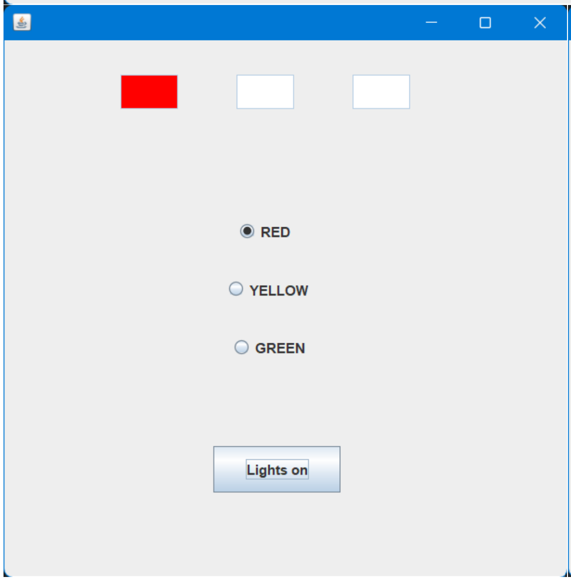
6.3 Code

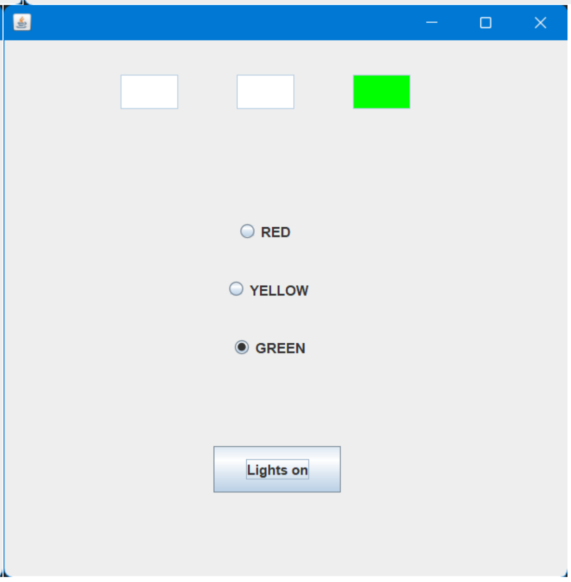
```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Traffic extends JFrame implements ActionListener {
    static JRadioButton red, yellow, green;
    static JTextField R, Y, G;

    Traffic() {
    }

    public void actionPerformed(ActionEvent e) {
        // String s = e.getActionCommand();
        if (red.isSelected()) {
            R.setBackground(Color.RED);
            Y.setBackground(Color.WHITE);
            G.setBackground(Color.WHITE);
        }
        if (yellow.isSelected()) {
            R.setBackground(Color.WHITE);
            Y.setBackground(Color.YELLOW);
            G.setBackground(Color.WHITE);
        }
        if (green.isSelected()) {
            R.setBackground(Color.WHITE);
            Y.setBackground(Color.WHITE);
            G.setBackground(Color.GREEN);
        }
    }



    public static void main(String[] args) {
        Traffic trf = new Traffic();

        JFrame f = new JFrame();
        f.setDefaultCloseOperation(EXIT_ON_CLOSE);
        f.setLayout(null);
        f.setSize(500, 500);
        f.setBackground(Color.WHITE);
        f.setVisible(true);

        R = new JTextField();
        Y = new JTextField();
        G = new JTextField();

        R.setEditable(false);
        Y.setEditable(false);
        G.setEditable(false);

        R.setBounds(100, 30, 50, 30);
        Y.setBounds(200, 30, 50, 30);
        G.setBounds(300, 30, 50, 30);

        R.setBackground(Color.WHITE);
        Y.setBackground(Color.WHITE);
        G.setBackground(Color.WHITE);

        red = new JRadioButton("RED");
        yellow = new JRadioButton("YELLOW");
        green = new JRadioButton("GREEN");

        red.setBounds(200, 150, 80, 30);
        yellow.setBounds(190, 200, 80, 30);
        green.setBounds(195, 250, 80, 30);

        ButtonGroup bg = new ButtonGroup();

        bg.add(red);
        bg.add(yellow);
        bg.add(green);

        red.addActionListener(trf);
        yellow.addActionListener(trf);
        green.addActionListener(trf);

        f.add(R);
        f.add(Y);
        f.add(G);
        f.add(red);
        f.add(yellow);
        f.add(green);

    }
}
```

6.4 Sample Output


RED
YELLOW
GREEN
Lights on


RED
[✓ YELLOW]
GREEN
Lights on


[✓ RED]
YELLOW
GREEN
Lights on


RED
YELLOW
[✓ GREEN]
Lights on