Course Outcome 5

Experiment 35 **Date: 09.05.2024**

Drawing Different Shapes

Aim:

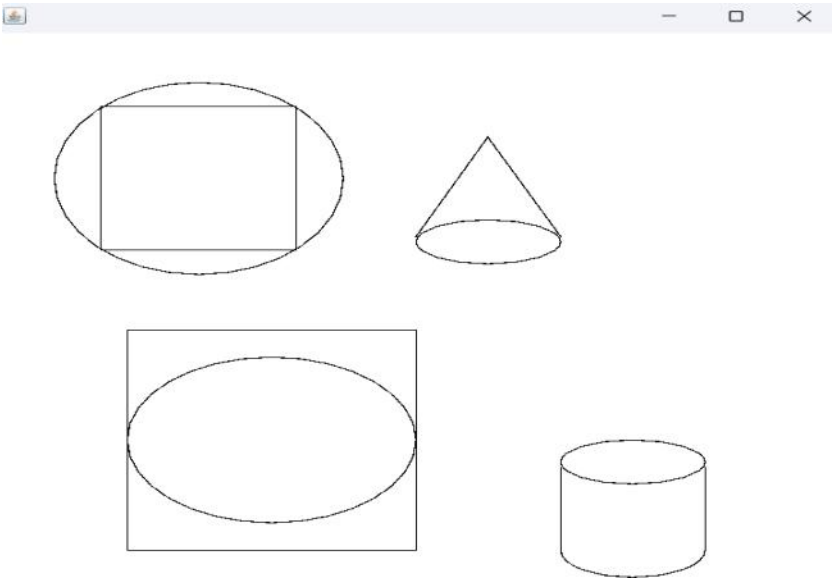
Create the Applet with the following shapes.

i)cone
ii)cylinder
iii)square inside the oval
Iv)circle inside rounded square

Program

```
import java.awt.*;

public class shape extends
Frame { Color c1;

public shape()
{ setVisible(true);
setSize(600, 600);}

public void paint(Graphics g)
{ g.drawOval(50,75,200,175);
g.drawRect(82,97,135,130);
g.drawOval(300,200,100,40);
g.drawLine(300,215,350,125);
g.drawLine(400,215,350,125);
g.drawRect(100,300,200,200);
g.drawOval(100,325,200,150);
g.drawOval(400,400,100,40);
g.drawLine(400,425,400,500);
g.drawLine(500,425,500,500);
g.drawArc(400,475,100,50,0,-
180);}

public static void main(String[] args)
{ new shape();

    }

  }
```

Output



Experiment 36 **Date: 09.05.2024**

Event Handling - 1

Aim:

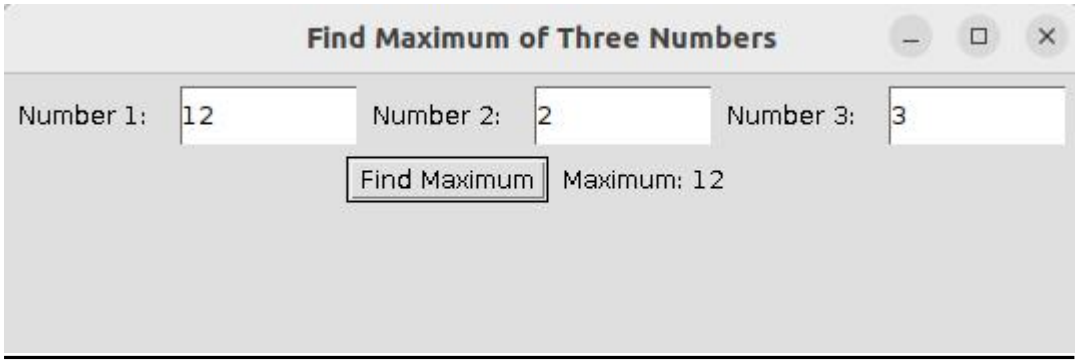
Program to find maximum of three numbers using AWT

Program

```
import java.awt.*;
import java.awt.event.*;
public class MaxOfThreeNumbersAWT extends Frame implements ActionListener {
TextField num1Field, num2Field, num3Field;
Label resultLabel;
public MaxOfThreeNumbersAWT() {
setLayout(new FlowLayout());
Label num1Label = new Label("Number 1: ");
num1Field = new TextField(10);
Label num2Label = new Label("Number 2: ");
num2Field = new TextField(10);
Label num3Label = new Label("Number 3: ");
num3Field = new TextField(10);
Button findMaxButton = new Button("Find Maximum");
findMaxButton.addActionListener(this);
resultLabel = new Label("Result will be displayed here");
add(num1Label);
add(num1Field);
add(num2Label);
add(num2Field);
add(num3Label);
add(num3Field);
add(findMaxButton);
add(resultLabel);
setTitle("Find Maximum of Three Numbers");
setSize(300, 200);
```

```
setVisible(true);
addWindowListener(new WindowAdapter() {
public void windowClosing(WindowEvent we) {
System.exit(0);
}
});
  }
public void actionPerformed(ActionEvent e) {
try {
int num1 = Integer.parseInt(num1Field.getText());
int num2 = Integer.parseInt(num2Field.getText());
int num3 = Integer.parseInt(num3Field.getText());
int max = Math.max(num1, Math.max(num2, num3));
resultLabel.setText("Maximum: " + max);
  } catch (NumberFormatException ex) {
resultLabel.setText("Invalid input. Please enter valid numbers.");
}
}
public static void main(String[] args) {
new MaxOfThreeNumbersAWT();
}
}
```

Output



Experiment 37 **Date: 09.05.2024**

Event Handling - 2

Aim:

Find the percentage of marks obtained by a student in 5 subjects. Display a happy face if he secures above 50% or a sad face if otherwise.

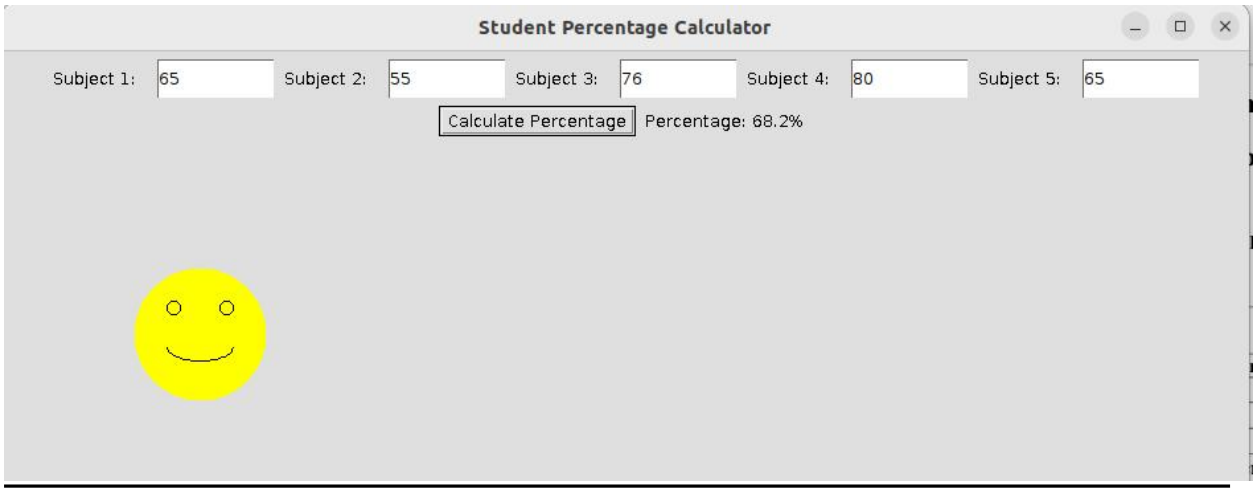
Program

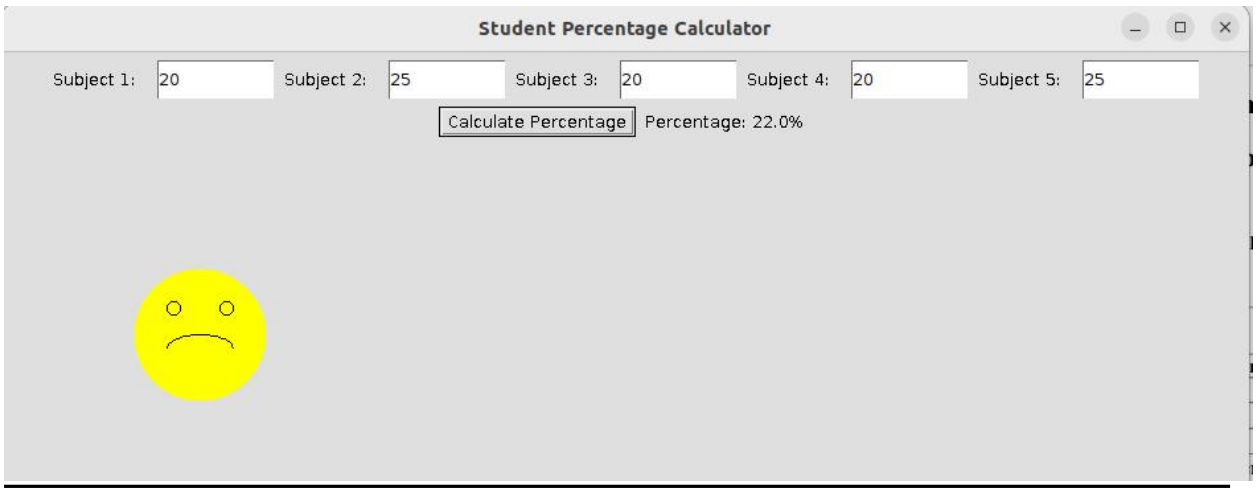
```
import java.awt.*;
import java.awt.event.*;
public class StudentPercentage extends Frame implements ActionListener {
TextField[] markFields = new TextField[5];
Label resultLabel;
public StudentPercentage() {
setLayout(new FlowLayout());
for (int i = 0; i < 5; i++) {
Label markLabel = new Label("Subject " + (i + 1) + ": ");
markFields[i] = new TextField(10);
add(markLabel);
add(markFields[i]);
}
Button calculateButton = new Button("Calculate Percentage");
calculateButton.addActionListener(this);
resultLabel = new Label("Result will be displayed here");
add(calculateButton);
add(resultLabel);
setTitle("Student Percentage Calculator");
setSize(300, 400);
setVisible(true);
 addWindowListener(new WindowAdapter() {
 public void windowClosing(WindowEvent we) {
 System.exit(0);
   }
```

```
  });
   }
public void actionPerformed(ActionEvent e) {
try {
int totalMarks = 0;
for (TextField markField : markFields) {
totalMarks += Integer.parseInt(markField.getText());
}
double percentage = (totalMarks / 500.0) * 100;
resultLabel.setText("Percentage: " + percentage + "%");
repaint();
} catch (NumberFormatException ex) {
resultLabel.setText("Invalid input. Please enter valid numbers.");
  }
}
public void paint(Graphics g) {
try {
int totalMarks = 0;
for (TextField markField : markFields) {
totalMarks += Integer.parseInt(markField.getText());
}
double percentage = (totalMarks / 500.0) * 100;
  if (percentage > 50) {
  drawHappyFace(g, 100, 200);
  } else {
 drawSadFace(g, 100, 200);
  }
  } catch (NumberFormatException ex) {
  }
```

```
}
  private void drawHappyFace(Graphics g, int x, int y) {
  g.setC olor(Color.YELLOW);
  g.fillOval(x, y, 100, 100);
  g.setColor(Color.BLACK);
  g.drawOval(x + 25, y + 25, 10, 10);
  g.drawOval(x + 65, y + 25, 10, 10);
  g.drawArc(x + 25, y + 50, 50, 20, 0, -180);
  }
  private void drawSadFace(Graphics g, int x, int y) {
  g.setColor(Color.YELLOW);
  g.fillOval(x, y, 100, 100);
  g.setColor(Color.BLACK);
  g.drawOval(x + 25, y + 25, 10, 10);
  g.drawOval(x + 65, y + 25, 10, 10);
  g.drawArc(x + 25, y + 50, 50, 20, 0, 180);
}
public static void main(String[] args) {
new StudentPercentage();
        }
}
```

Output


Student Percentage Calculator
Subject 1: 65
Subject 2: 55
Subject 3: 76
Subject 4: 80
Subject 5: 65
Calculate Percentage
Percentage: 68.2%


Student Percentage Calculator
Subject 1: 20
Subject 2: 25
Subject 3: 20
Subject 4: 20
Subject 5: 25
Calculate Percentage
Percentage: 22.0%

Experiment 38 **Date: 09.05.2024**

Event Handling - 3

Aim:

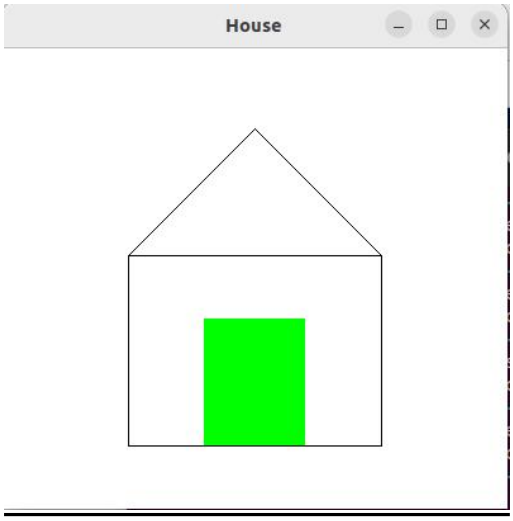
Construct a house on mouse click event, change the color of the door from blue to red.

Program

```
import java.awt.*;
import java.awt.event.*;
public class House extends Frame implements MouseListener {
private Color doorColor;
public House() {
setTitle("House");
setSize(400, 400);
setBackground(Color.WHITE);
doorColor = Color.BLUE;
addMouseListener(this);
setVisible(true);
addWindowListener(new WindowAdapter() {
public void windowClosing(WindowEvent e) {
dispose();
}});
}
public void paint(Graphics g) {
super.paint(g);
drawHouse(g);
}
public void drawHouse(Graphics g) {
g.setColor(Color.BLACK);
g.drawRect(100, 200, 200, 150);
g.drawLine(100, 200, 200, 100);
g.drawLine(200, 100, 300, 200);
g.setColor(doorColor);
```

```
g.fillRect(160, 250, 80, 100);
}
public void mouseClicked(MouseEvent e) {
int x = e.getX();
int y = e.getY();
if (x >= 160 && x <= 240 && y >= 250 && y <= 350) {
if (doorColor == Color.BLUE) {
doorColor = Color.GREEN;
} else {
doorColor = Color.BLUE;
}
repaint();
}}
public void mousePressed(MouseEvent e) {}
public void mouseReleased(MouseEvent e) {}
public void mouseEntered(MouseEvent e) {}
public void mouseExited(MouseEvent e) {}
public static void main(String[] args) {
new House();
}
}
```

Output



Experiment 39 **Date: 09.05.2024**

Event Handling - 4

Aim:

Implement a simple calculator using AWT components.

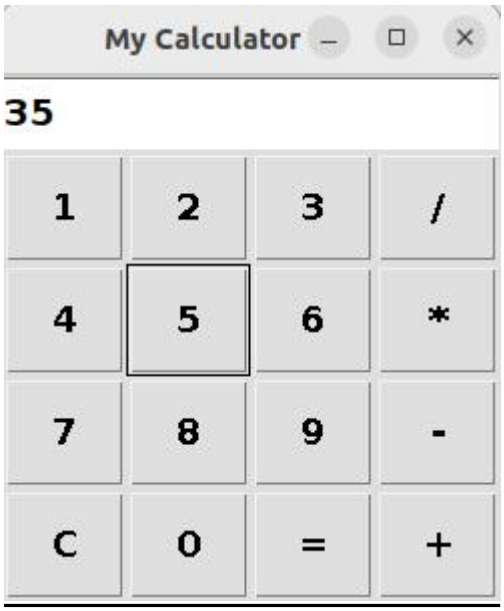
Program

```
import java.awt.*;
import java.awt.event.*;
class Calcu extends Frame implements ActionListener {
TextField tInput;
Panel panel;
String btnString[] = {"1", "2", "3", "/",
"4", "5", "6", "*",
"7", "8", "9", "-",
"C", "0", "=", "+"};
Button btn[] = new Button[16];
int num1 = 0, num2 = 0, result = 0;
char op;
public Calcu() {
Font f = new Font("Cambria", Font.BOLD, 18);
tInput = new TextField(10);
tInput.setFont(f);
panel = new Panel();
add(tInput, "North");
add(panel, "Center");
panel.setLayout(new GridLayout(4,4));
for(int i=0; i < 16; i++) {
btn[i] = new Button(btnString[i]);
btn[i].setFont(f);
btn[i].addActionListener(this);
panel.add(btn[i]);
}
```

```
addWindowListener(new WindowAdapter(){
public void windowClosing(WindowEvent we) {
System.exit(0);
}
});
}
public void actionPerformed(ActionEvent ae) {
String str = ae.getActionCommand();
if(str.equals("+")) {
op = '+';
num1 = Integer.parseInt(tInput.getText());
tInput.setText("");
}
else if(str.equals("-")) {
op = '-';
num1 = Integer.parseInt(tInput.getText());
tInput.setText("");
}
else if(str.equals("*")) {
op = '*';
num1 = Integer.parseInt(tInput.getText());
tInput.setText("");
}
else if(str.equals("/")) {
op = '/';
num1 = Integer.parseInt(tInput.getText());
tInput.setText("");
}
else if(str.equals("=")) {
num2 = Integer.parseInt(tInput.getText());
switch(op) {
```

```
case '+' : result = num1 + num2;
break;
case '-' : result = num1 - num2;
break;
case '*' : result = num1 * num2;
break;
case '/' : result = num1 / num2;
break;
}
tInput.setText(result + "");
result = 0;
}
else if(str.equals("C")) {
tInput.setText("");
num1 = num2 = result = 0;
}
else {
tInput.setText(tInput.getText() + str);
}
}
public static void main(String args[]) {
Calcu m = new Calcu();
m.setTitle("My Calculator");
m.setSize(250,300);
m.setVisible(true);
}
}
```

Output



Experiment 40 **Date: 09.05.2024**

Event Handling - 5

Aim:

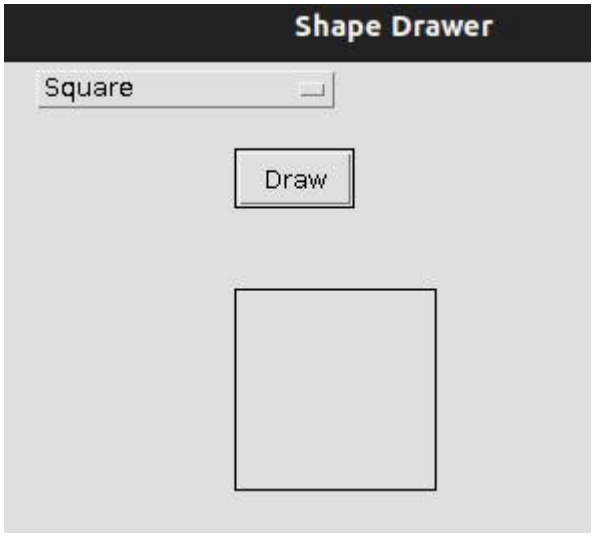
Develop a program that has a Choice component which contains the names of shapes such as rectangle, triangle, square and circle. Draw the corresponding shapes for given parameters as per user's choice.

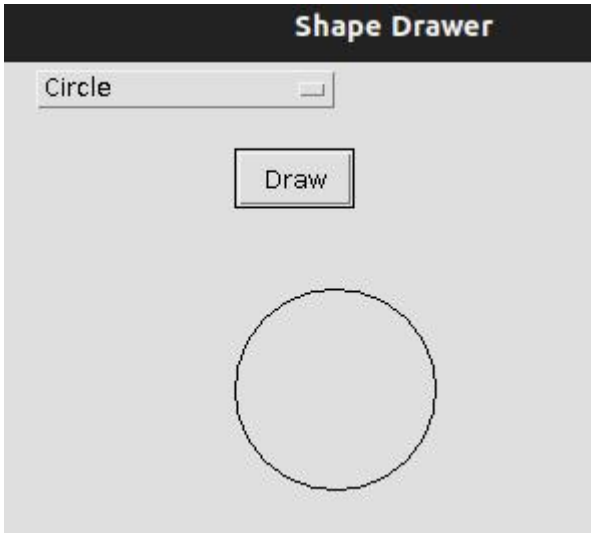
Program

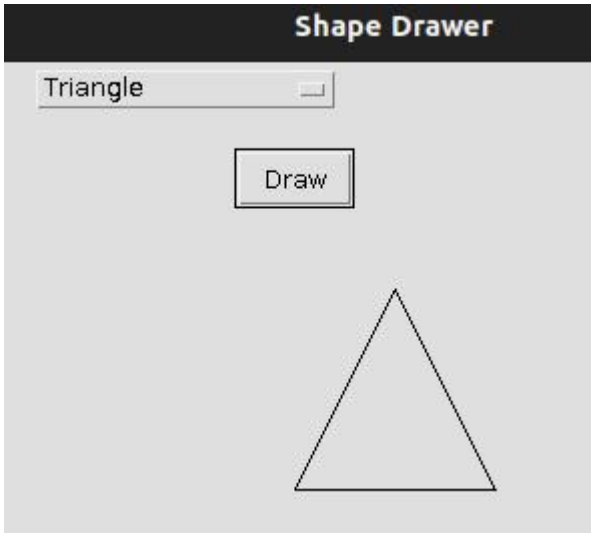
```
import java.awt.*;
import
java.awt.event.*;

public class ShapeDrawer extends Frame implements
ActionListener { Button drawButton;

String selectedShape; public
ShapeDrawer() { setSize(400,
400); setTitle("Shape
Drawer"); setResizable(false);
setLayout(null);

Choice c = new Choice();
c.add("Selec the Choice");
c.add("Square");

c.add("Circle");

c.add("Rectangle");

c.add("Triangle");
c.setBounds(20, 40, 150, 20);
add(c);

c.addItemListener(new ItemListener() { public
void itemStateChanged(ItemEvent e) {

selectedShape = c.getSelectedItem(); }

});

drawButton = new Button("Draw");
drawButton.setBounds(120, 80, 60, 30);
add(drawButton);
drawButton.addActionListener(this);
setVisible(true); }
```

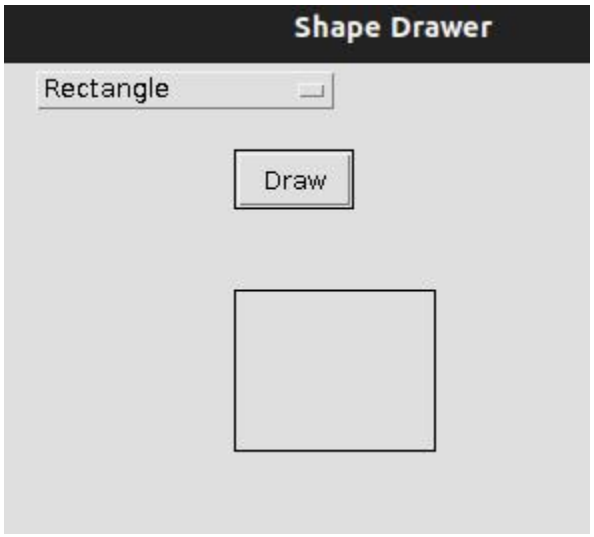
```
public void actionPerformed(ActionEvent e)
{ repaint(); }

public void paint(Graphics g)
{ super.paint(g);
g.setColor(Color.BLACK); switch
(selectedShape) {

case "Square":

g.drawRect(120, 150, 100, 100);

break; case
"Circle":

g.drawOval(120, 150, 100,100);

break;

case "Rectangle":

g.drawRect(120, 150, 100, 80);

break; case
"Triangle":

int[] xPoints = {200, 150, 250};

int[] yPoints = {150, 250, 250};

g.drawPolygon(xPoints, yPoints, 3); break; }

}

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

      }
```

Output


Shape Drawer
Square
Draw


Shape Drawer
Circle
Draw


Shape Drawer
Triangle
Draw


Shape Drawer
Rectangle
Draw

Experiment 41 **Date 09.05.2024**

Handling Mouse Events

Aim:

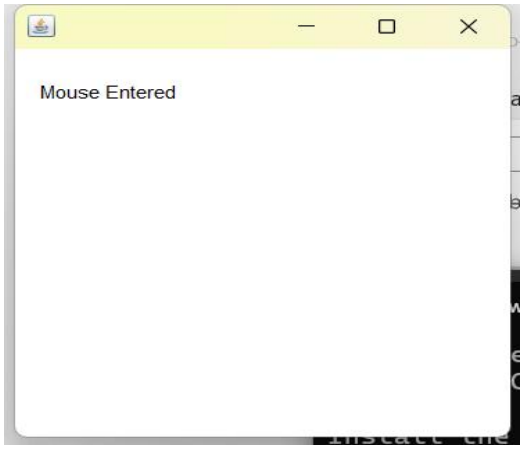
Develop a program to handle all mouse events.

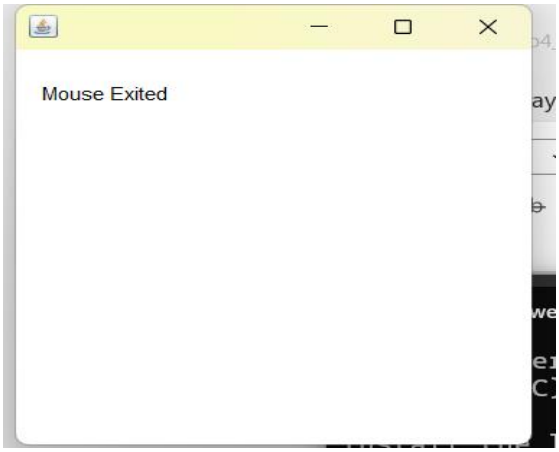
Program

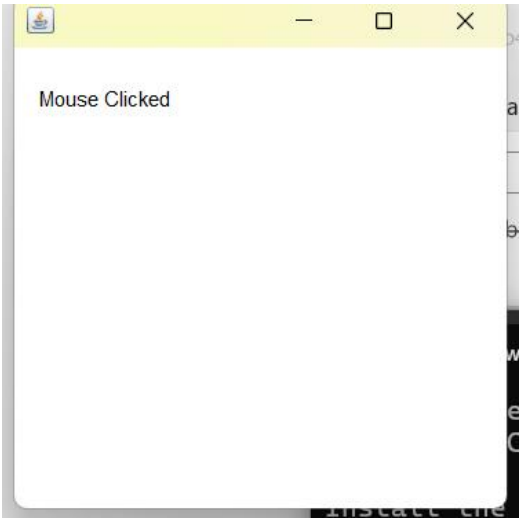
```
import    java.awt.*;
import
java.awt.event.*;

public class MouseListenerExample extends Frame implements
MouseListener{ Label l;

MouseListenerExample()
{ addMouseListener(this)
;       l=new       Label();
l.setBounds(20,50,100,2
0); add(l);

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(MouseEvente)

{l.setText("Mouse Exited");

}

public void mouse Pressed(MouseEvente)

{ l.setText("Mouse Pressed");

}

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

}

public static void main(String[] args) {

new MouseListenerExample();
```

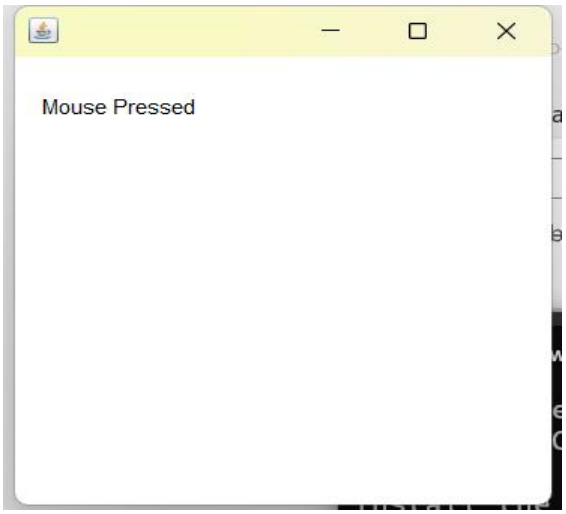
```
}
}
```

Output









Experiment 42 **Date 09.05.2024**

Handling Window Events

Aim:

Develop a program to handle all window events

Program

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

import java.awt.event.WindowListener;

public class Window implements
WindowListener { public Window() {

Frame f = new Frame("WindowListener
Example"); Label l = new Label("Handling
window events"); l.setBounds(100, 90, 240,
120); l.setForeground(Color.GREEN);

Font f1=new Font("Serif", Font.BOLD, 22);
l.setFont(f1);

f.add(l);
f.addWindowListener(thi
s); f.setSize(400, 300);
f.setLayout(null);
f.setVisible(true); }

public void windowOpened(WindowEvent e)
{ System.out.println("Window is opened!"); }
public void windowClosing(WindowEvent e)
{ System.out.println("Window is closing...");
System.exit(0); }

public void windowClosed(WindowEvent e)
{ System.out.println("Window is closed!"); }
public void windowIconified(WindowEvent
e) { System.out.println("Window is
iconified!"); }

public void windowDeiconified(WindowEvent e)
{ System.out.println("Window is deiconified!"); }
public void windowActivated(WindowEvent e)
{ System.out.println("Window is activated!"); }
public void windowDeactivated(WindowEvent e)
{ System.out.println("Window is
deactivated!"); }
```

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

Output

Window is activated!

Window is opened!

Window is deactivated!

Experiment 43 **Date 09.05.2024**

Handling Key Events

Aim:

Develop a program to handle Key events.

Program

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

public class KeyListenerExample extends Frame implements KeyListener
{ Label l;

TextArea area;
KeyListenerExample
() { l = new Label();

l.setBounds (20, 50, 100, 20);
area = new TextArea();
area.setBounds (20, 80, 300,
300);
area.addKeyListener(this);

add(l);
add(area);

setSize (400, 400);
setLayout (null);
setVisible (true); }

public void keyPressed (KeyEvent e)
{ l.setText ("Key Pressed"); }

public void keyReleased (KeyEvent
e) { l.setText ("Key Released");

}public void keyTyped (KeyEvent e)
{ l.setText ("Key Typed"); }

public static void main(String[] args)
{ new KeyListenerExample();

    }

    }
```

Output