Lab 2

Java Swing Basics and Event Handling

1. Write a Java Swing program to illustrate the concept of MouseListener, MouseWheelListener, WindowListener and KeyListener.

Question.java

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

public class Question1 {
   public static class MouseKeyListener extends JFrame implements MouseListener {
      public MouseKeyListener() {
         this.setTitle("Mouse Event Listener");
         this.setSize(500, 500);
         this.addMouseListener(this);
         this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
         this.setVisible(true);
      }
      @Override
      public void mouseClicked(MouseEvent e) {
         System.out.println("Mouse clicked");
      }
      @Override
      public void mousePressed(MouseEvent e) {
         System.out.println("Mouse pressed");
      }
      @Override
      public void mouseReleased(MouseEvent e) {
```

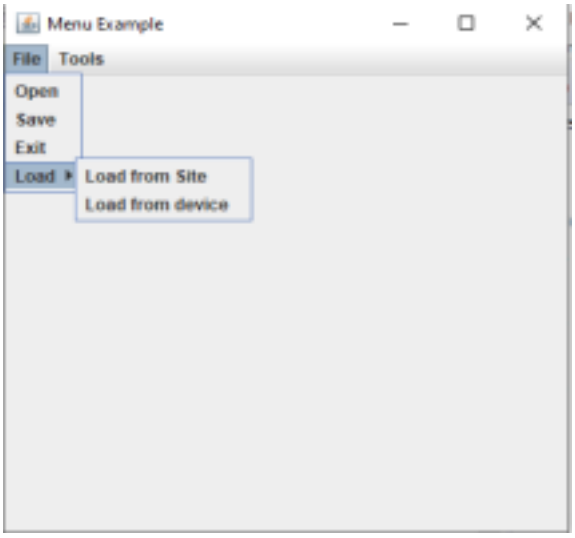
```
            System.out.println("Mouse released");
        }
        @Override
        public void mouseEntered(MouseEvent e) {
            System.out.println("Mouse entered");
        }
        @Override
        public void mouseExited(MouseEvent e) {
            System.out.println("Mouse exited");
        }
    }
    public static class WindowKeyListener extends JFrame implements WindowListener
{
        public WindowKeyListener() {
            this.setTitle("Window Event Listener");
            this.setSize(500, 500);
            this.addWindowListener(this);
            this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            this.setVisible(true);
        }
        @Override
        public void windowOpened(WindowEvent e) {
            System.out.println("Window opened");
        }
        @Override
        public void windowClosing(WindowEvent e) {
            System.out.println("Window is closing");
        }
```

```
    @Override
    public void windowClosed(WindowEvent e) {
        System.out.println("Window closed");
    }
    @Override
    public void windowIconified(WindowEvent e) {
        System.out.println("Window iconified");
    }
    @Override
    public void windowDeiconified(WindowEvent e) {
        System.out.println("Window deiconified");
    }

    @Override
    public void windowActivated(WindowEvent e) {
        System.out.println("Window activated");
    }
    @Override
    public void windowDeactivated(WindowEvent e) {
        System.out.println("Window deactivated");
    }
}
public static class KeyEventListener extends JFrame implements KeyListener {
    public KeyEventListener() {
        this.setTitle("Key Event Listener");
        this.setSize(500, 500);
        this.addKeyListener(this);
```

```
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setVisible(true);
    }
    @Override
    public void keyTyped(KeyEvent e) {
        System.out.println("Key typed");
    }
    @Override
    public void keyPressed(KeyEvent e) {
        System.out.println("Key pressed");
}
    @Override
    public void keyReleased(KeyEvent e) {
        System.out.println("Key released");
    }
  }
  public static void main(String[] args) {
    new MouseKeyListener();
    new WindowKeyListener();
    new KeyEventListener();
  }
}
```

2. Create a frame having menu as below.



Also give a message to user using JOptionPane of which menu-item user has clicked.

Question2.java

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

public class Question2 {
    public static void main(String[] args) {
        JFrame jf = new JFrame("Menu Example ");
        jf.setSize(500, 500);
        jf.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        jf.setLayout(null);
        JMenuBar m = new JMenuBar();
        m.setBounds(0, 0, 500, 20);
        JMenu m1 = new JMenu("file");
        JMenu m2 = new JMenu("tools");

        JMenuItem jm1 = new JMenuItem("open");
        JMenuItem jm2 = new JMenuItem("save");
        JMenuItem jm3 = new JMenuItem("exit");
        JMenu jm4 = new JMenu("load");
        JMenuItem jm11 = new JMenuItem("load from site");
        JMenuItem jm12 = new JMenuItem("load from device");
        m1.add(jm1);
        m1.add(jm2);
```

```
        m1.add(jm3);
        m1.add(jm4);
        jm4.add(jm11);
        jm4.add(jm12);
        m.add(m1);
        m.add(m2);
        jf.add(m);

        jf.setResizable(false);
        jf.setVisible(true);

        // next way of handiling simple actions using lamda expression
        jm1.addActionListener(e -> JOptionPane.showMessageDialog(jf, "You clicked
    'Open'"));
        jm2.addActionListener(e -> JOptionPane.showMessageDialog(jf, "You clicked
    'Save'"));
        jm3.addActionListener(e -> {
            JOptionPane.showMessageDialog(jf, "You clicked 'Exit'");
            System.exit(0);
        });
        jm11.addActionListener(e -> JOptionPane.showMessageDialog(jf, "You clicked
    'Load from Site'"));
        jm12.addActionListener(e -> JOptionPane.showMessageDialog(jf, "You clicked
    'Load from Device'"));

      }
    }
```

3. Write a Java Program to create a window where user can draw anything by dragging mouse on it.

Question3.java

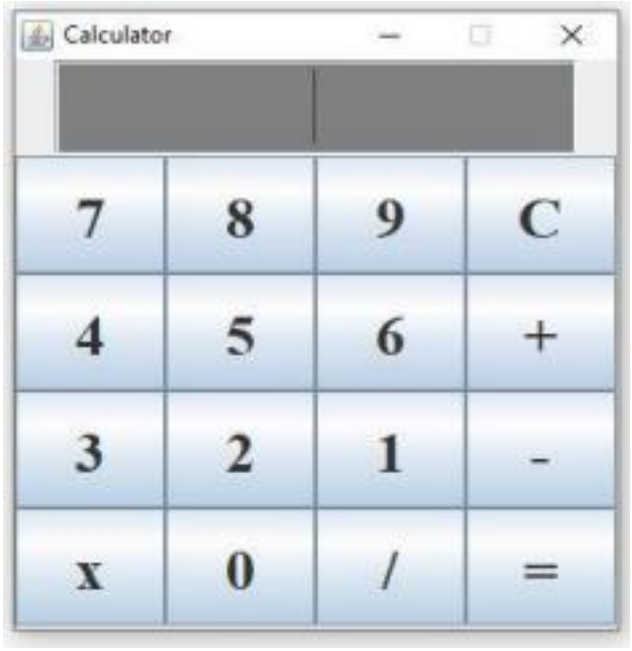
```
import java.awt.Color;
import java.awt.Graphics;
import java.awt.event.*;
import java.awt.event.MouseEvent;
import javax.swing.*;
public class Question3 extends MouseMotionAdapter{
    JFrame f;
   public Question3(){
    f= new JFrame("Draw Anything by Dragging Mouse");
    f.setSize(1000, 1000);
    f.setLayout(null);
```

```
 f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 f.addMouseMotionListener(this);
 f.setVisible(true);
}

@Override
public void mouseDragged(MouseEvent e){
   Graphics g=f.getGraphics();
   g.setColor(Color.red);
   g.fillOval(e.getX(),e.getY(),20,20);
}
 public static void main(String[] args) {
   new Question3();
 }

}
```

4. Write a Java program in Java to generate a frame as below:



Application above should perform appropriate actions as per the button click.

Question4.java

```
import java.awt.*;
import javax.swing.*;
public class Question4 {
   public static void main (String[]args){
      JButton[] buttons;
```

```
        JFrame j = new JFrame("Calculator");
        j.setLayout(null);
        JPanel p = new JPanel();
        p.setBounds(0, 0, 550, 80);
        JTextArea area = new JTextArea();
        area.setBounds(0, 0, 550, 100);
        j.add(area);
        JPanel jp1 = new JPanel();
        jp1.setBounds(0, 100, 550, 420);
        String[] button = {"7", "8", "9", "C", "4", "5", "6", "+", "3", "2", "1", "*", "0",
"%", "/","="};
         buttons = new JButton[button.length];
        for (int i = 0; i < button.length; i++) {
           buttons[i] = new JButton(button[i]);
           jp1.add(buttons[i]);
        }

        jp1.setLayout(new GridLayout(4, 4));
        j.add(jp1);

        j.setSize(550, 550);
        j.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        j.setVisible(true);

    }

}
```

Discussion and Conclusion:

In this lab, we learn about java swing and event handler. we create a simple application and handle them with different event listener like MouseListener, MouseWheelListener, WindowListener and KeyListener. Create a simple MenuBar , menubar contains menu and menuItem and simple calculator and handle each button.