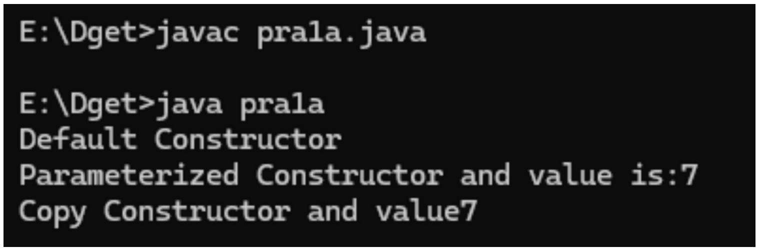
OOPS concept java-1

Aim: a. Write a program to create a class and implement a default, overloaded and copy Constructor.

Code:

```
class MyClass
{
   private int a;
   public MyClass()
   {
      System.out.println("Default Constructor");
   }
   public MyClass(int value)
   {
      a=value;
      System.out.println("Parameterized Constructor and value is:"+a);
   }
   public MyClass(MyClass other)
   {
      a = other.a;
      System.out.println("Copy Constructor and value"+a);
   }
}
public class pra1a
{
   public static void main(String[] args)
   {
      MyClass obj1 = new MyClass();
      MyClass obj2 = new MyClass(7);
      MyClass obj3 = new MyClass(obj2);

   }
}
```

Output:



```
E:\Dget>javac pra1a.java

E:\Dget>java pra1a
Default Constructor
Parameterized Constructor and value is:7
Copy Constructor and value7
```

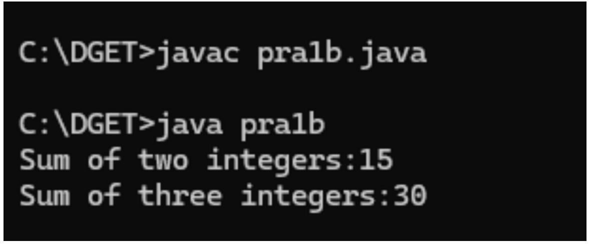
Aim: b. Write a program to create a class and implement the concepts of Method Overloading

Code:

```
class OperOver
{
   public int add(int a,int b)
```

```
    {
        return a+b;
    }
    public int add(int a,int b,int c)
    {
        return a+b+c;
    }
}
public class pra1b
{
    public static void main(String[] args)
    {
        OperOver obj= new OperOver();
        int sum1=obj.add(5,10);
        int sum2=obj.add(5,10,15);

        System.out.println("Sum of two integers:"+sum1);
        System.out.println("Sum of three integers:"+sum2);


    }
}
```

Output:



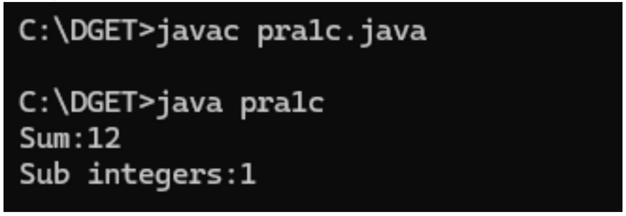
```
C:\DGET>javac pra1b.java

C:\DGET>java pra1b
Sum of two integers:15
Sum of three integers:30
```

Aim: c. Write a program to create a class and implement the concepts of Static methods

Code:

```
class DemoStaticMethods
{
    public static int add(int a,int b)
    {
        return a+b;
    }
    public static int sub(int a,int b)
    {
        return a-b;
    }
}
public class pra1c
{
    public static void main(String[] args)
    {
        int sum = DemoStaticMethods.add(8,4);
        int diff = DemoStaticMethods.sub(7,6);

        System.out.println("Sum:"+sum);
        System.out.println("Sub integers:"+diff);

    }
}
```

Output:



```
C:\DGET>javac pra1c.java

C:\DGET>java pra1c
Sum:12
Sub integers:1
```

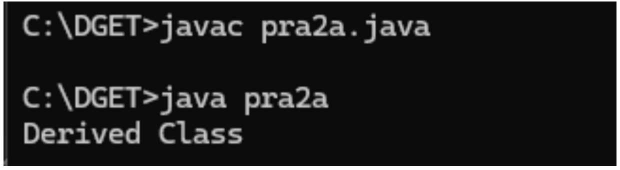
2)

Aim: a. Write a program to implement the concepts of Inheritance and Method overriding

Code:

```
class A
{
    void show()
    {
        System.out.println("Base Class");
    }
}
class B extends A
{
    void show()
    {
        System.out.println("Derived Class");
    }
}
class pra2a
{
    public static void main(String args[])
    {
        B s=new B();
        s.show();
    }
}
```

Output:



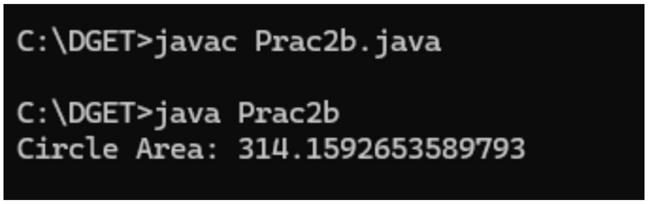
```
C:\DGET>javac pra2a.java

C:\DGET>java pra2a
Derived Class
```

Aim: b. Write a program to implement the concepts of Abstract classes and methods

Code:

```
abstract class Shape
{
   public abstract double area();
}
class Circle extends Shape
{
   private double radius;
   public Circle(double radius)
   {
      this.radius=radius;
   }
   @Override
   public double area()
   {
      return Math.PI*radius*radius;
   }
}
public class pra2b
{
   public static void main(String[] args)
   {
      Circle circle=new Circle(10.0);
      System.out.println("Circle Area:"+circle.area());
   }
}
```

Output:



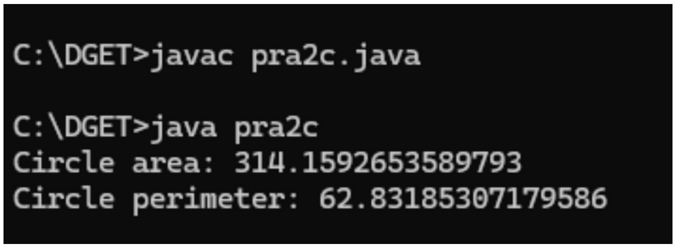
```
C:\DGET>javac Prac2b.java

C:\DGET>java Prac2b
Circle Area: 314.1592653589793
```

Aim: c. Write a program to implement the concept of interfaces

Code:

```
interface Shape
{
   double area();
   double perimeter();
}
class Circle implements Shape
{
   private double radius;
   public Circle(double radius)
   {
      this.radius=radius;
   }
   @Override
   public double area()
   {
      return Math.PI*radius*radius;
   }
   @Override
   public double perimeter()
   {
      return 2*Math.PI*radius;
   }
}
public class pra2c
{
   public static void main(String[] args)
   {
      Circle circle=new Circle(10.0);
      System.out.println("circle area:"+ circle.area());
      System.out.println("circle perimeter:"+ circle.perimeter());
   }
}
```

Output:



```
C:\DGET>javac pra2c.java

C:\DGET>java pra2c
Circle area: 314.1592653589793
Circle perimeter: 62.83185307179586
```

Exceptions-3

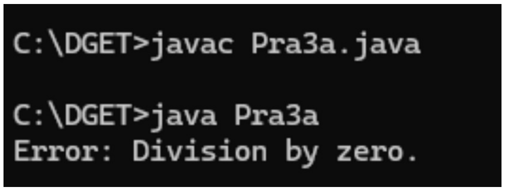
Aim: a. Write a program to raise built-in exceptions and raise them as per the requirements

Code:

```
public class pra3a
{
   public static void main(String[] args)
   {
      try
      {
         int result=divide(10,0);
         System.out.println("Result:"+result);
      }
      catch(ArithmeticException e)
      {
         System.err.println("Error:Division by zero.");
      }
   }
```

```
    public static int divide(int a, int b)
    {
        return a/b;
    }
}
```

Output:



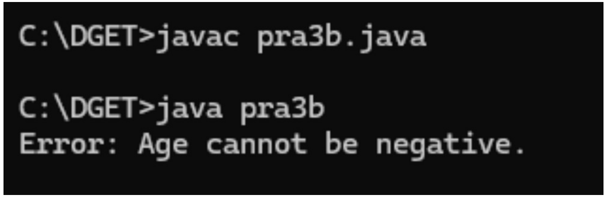
```
C:\DGET>javac Pra3a.java

C:\DGET>java Pra3a
Error: Division by zero.
```

Aim: b. Write a program to define user defined exceptions and raise them as per the requirements

Code:

```
class CustomException extends Exception
{
    public CustomException(String message)
    {
        super(message);
    }
}
public class pra3b
{
    public static void main(String[] args)
    {
        try
        {
            int age=-20;
            if(age<0)
            {
                throw new CustomException("Age cannot be negative.");
            }
            System.out.println("Age:"+age);
        }
        catch(CustomException e)
        {
            System.err.println("Error:"+ e.getMessage());
        }
    }
}
```

Output:



```
C:\DGET>javac pra3b.java

C:\DGET>java pra3b
Error: Age cannot be negative.
```

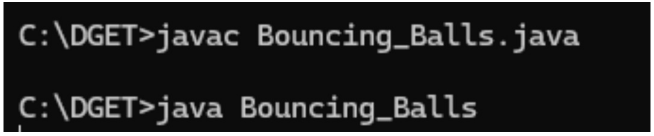
Practical no:-04 Multithreading

Aim: Write a java application to demonstrate 5 bouncing balls of different colors using threads.

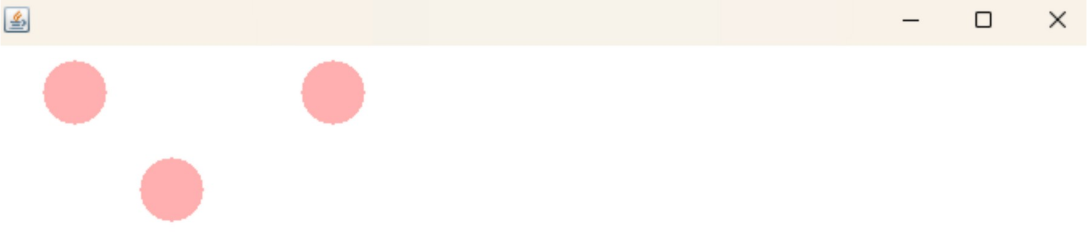
Code:

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

class Bouncing_Balls extends Frame implements MouseListener {
    int x = 40, y = 40, t1 = 1, t2 = 1;
    int x1 = 200, y1 = 40, t12 = 1, t22 = 1;
    int x2 = 100, y2 = 100, t13 = 1, t23 = 1;
    Thread th;

    Bouncing_Balls() {
        setSize(700, 800);
        setVisible(true);
        addMouseListener(this);

        // Creating a Runnable for thread
        Runnable ballMovement = new Runnable() {
            public void run() {
                while (true) {
                    x = x + t1;
                    y = y + t2;
                    x1 = x1 + t12;
                    y1 = y1 + t22;
                    x2 = x2 - t13;
                    y2 = y2 - t23;

                    // Check boundaries for each ball and reverse direction when hitting edge
                    if (x < 0 || x > 680) t1 = t1 * (-1);
                    if (y < 20 || y > 780) t2 = t2 * (-1);
                    if (x1 < 0 || x1 > 680) t12 = t12 * (-1);
                    if (y1 < 20 || y1 > 780) t22 = t22 * (-1);
                    if (x2 < 0 || x2 > 680) t13 = t13 * (-1);
                    if (y2 < 20 || y2 > 780) t23 = t23 * (-1);

                    try {
                        Thread.sleep(5);  // Slow down the thread for better visual effect
                    } catch (InterruptedException E) {
                        // Handle exception properly (although it's unlikely to be thrown here)
                    }
                    repaint();  // Repaint the frame to show the updated positions
                }
            }
        };
        th = new Thread(ballMovement);
    }
    public void mouseClicked(MouseEvent M) {
        th.start();  // Start the animation when the mouse is clicked
    }

    public void mousePressed(MouseEvent M) {}
    public void mouseReleased(MouseEvent M) {}
    public void mouseEntered(MouseEvent M) {}
    public void mouseExited(MouseEvent M) {}

    public void paint(Graphics g) {
        // Draw the three bouncing balls with different positions
        g.setColor(Color.pink);
        g.fillOval(x, y, 40, 40);  // First ball
        g.fillOval(x1, y1, 40, 40);  // Second ball
        g.fillOval(x2, y2, 40, 40);  // Third ball
    }

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

Output:



```
C:\DGET>javac Bouncing_Balls.java

C:\DGET>java Bouncing_Balls
```



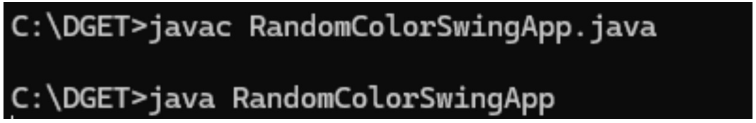
Practical no:-06 Swing

Aim: a. Create a swing application that randomly changes color on button click

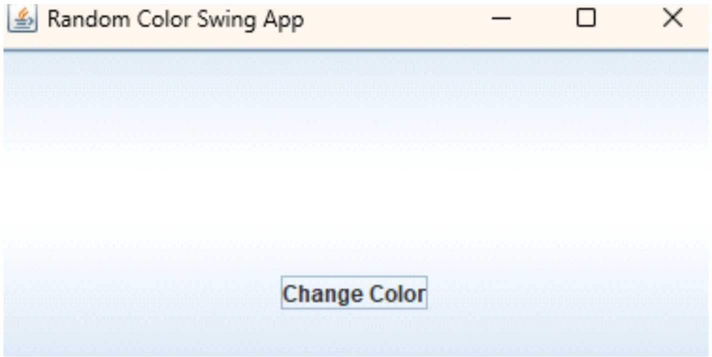
Code:

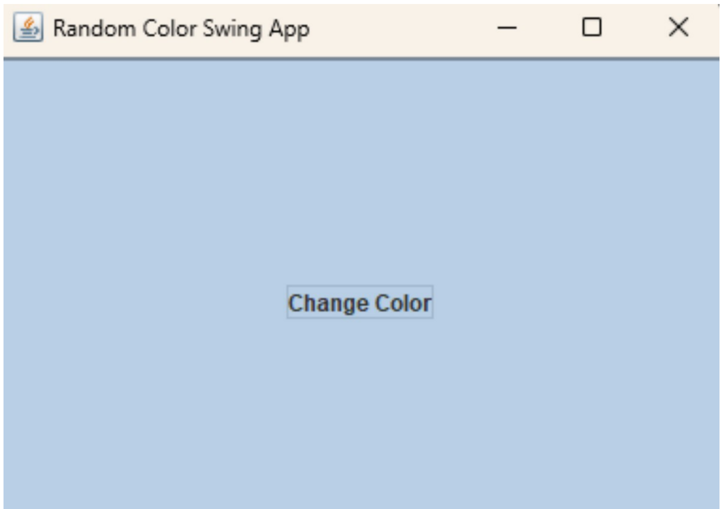
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.Random;

public class RandomColorSwingApp extends JFrame {

    private JButton changeColorButton;
    public RandomColorSwingApp() {
        setTitle("Random Color Swing App");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);

        // Create a button and add action listener to it
        changeColorButton = new JButton("Change Color");
        changeColorButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                changeBackgroundColor(); // Call the method to change background color
            }
        });

        // Set layout and add button to frame
        setLayout(new BorderLayout());
        add(changeColorButton, BorderLayout.CENTER);

        setVisible(true);
    }

    // Method to generate a random color and change the background
    private void changeBackgroundColor() {
        Random random = new Random();
        int red = random.nextInt(256); // Random value for Red (0-255)
        int green = random.nextInt(256); // Random value for Green (0-255)
        int blue = random.nextInt(256); // Random value for Blue (0-255)
        Color randomColor = new Color(red, green, blue); // Create a new color
        getContentPane().setBackground(randomColor); // Set the background color
    }

    public static void main(String[] args) {
        // Run the Swing application on the Event Dispatch Thread (EDT)
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new RandomColorSwingApp(); // Create and display the application
            }
        });
    }
}
```

Output:



```
C:\DGET>javac RandomColorSwingApp.java

C:\DGET>java RandomColorSwingApp
```





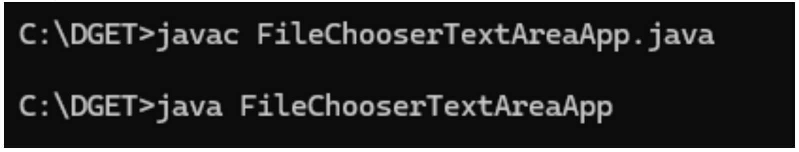
Aim: b)create a swing application to demostrate use of textarea using scrollpane to show contest of text file in textarea selected using file chooser

Code:

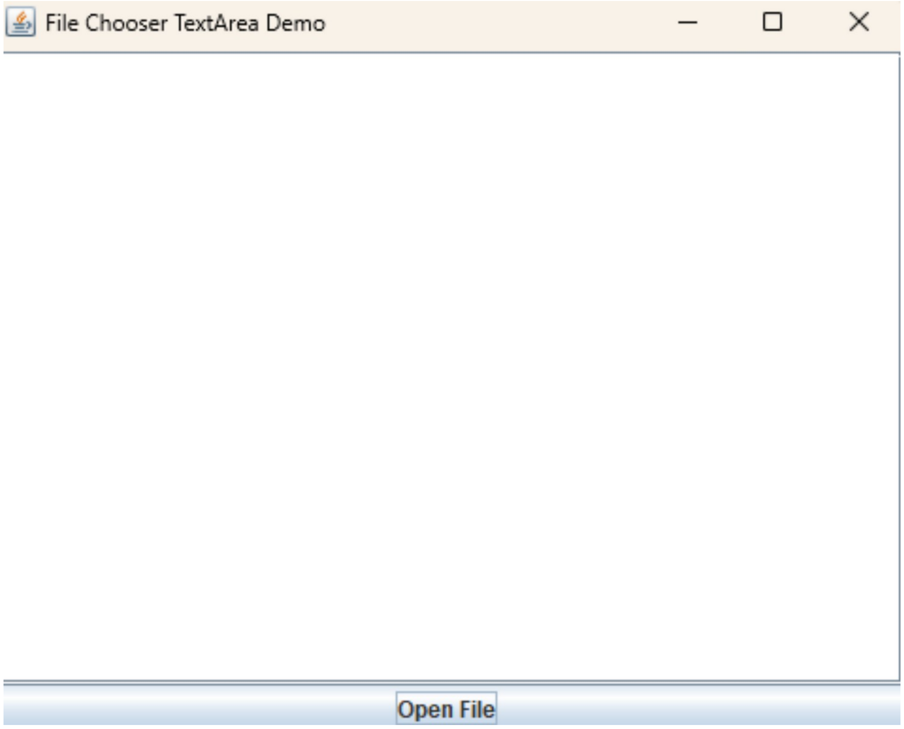
```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.*;

public class FileChooserTextAreaApp extends JFrame {
    private JTextArea textArea;
    private JScrollPane scrollPane;
    private JButton openFileButton;

    public FileChooserTextAreaApp() {
        // Set up the frame
        setTitle("File Chooser TextArea Demo");
        setSize(500, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null); // Center the window

        // Create the JTextArea and set it to be non-editable
        textArea = new JTextArea();
        textArea.setEditable(false);
        // Wrap the JTextArea in a JScrollPane
        scrollPane = new JScrollPane(textArea);

        // Create a button to open the file chooser
        openFileButton = new JButton("Open File");
        openFileButton.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                openFileChooser();
            }
        });

        // Set up the layout and add components
        setLayout(new BorderLayout());
        add(scrollPane, BorderLayout.CENTER);
        add(openFileButton, BorderLayout.SOUTH);

        // Display the frame
        setVisible(true);
    }

    // Method to open file chooser and display file content in JTextArea
    private void openFileChooser() {
        JFileChooser fileChooser = new JFileChooser();
        int returnValue = fileChooser.showOpenDialog(this);

        if (returnValue == JFileChooser.APPROVE_OPTION) {
            File selectedFile = fileChooser.getSelectedFile();
            try (BufferedReader reader = new BufferedReader(new FileReader(selectedFile))) {
                StringBuilder fileContent = new StringBuilder();
                String line;
                while ((line = reader.readLine()) != null) {
                    fileContent.append(line).append("\n");
                }
                textArea.setText(fileContent.toString()); // Set content of file to JTextArea
            } catch (IOException ex) {
                JOptionPane.showMessageDialog(this, "Error reading file: " + ex.getMessage(),
                        "File Read Error", JOptionPane.ERROR_MESSAGE);
            }
        }
    }

    // Main method to start the application
    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                new FileChooserTextAreaApp();
            }
        });
    }
```

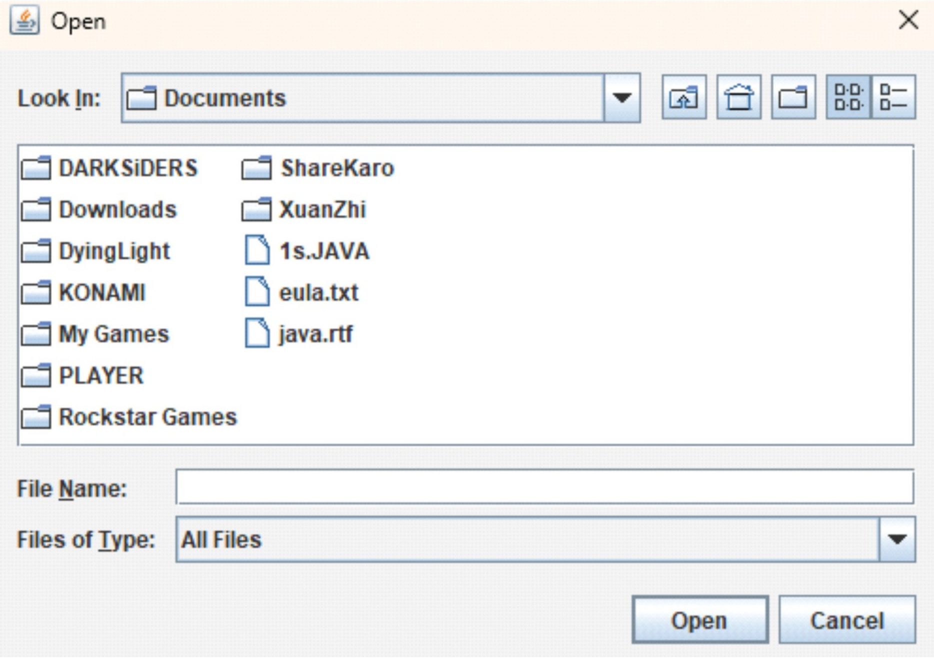
}

Output:



```
C:\DGET>javac FileChooserTextAreaApp.java

C:\DGET>java FileChooserTextAreaApp
```





Look In: Documents

```
DARKSiDERS      ShareKaro
Downloads       XuanZhi
DyingLight      1s.JAVA
KONAMI          eula.txt
My Games        java.rtf
PLAYER
Rockstar Games
```

File Name:

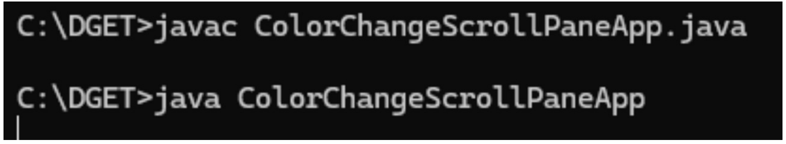
Files of Type: All Files

Aim: c. Create a Swing application to demonstrate use of scroll pane to change its color selected using colour chooser.

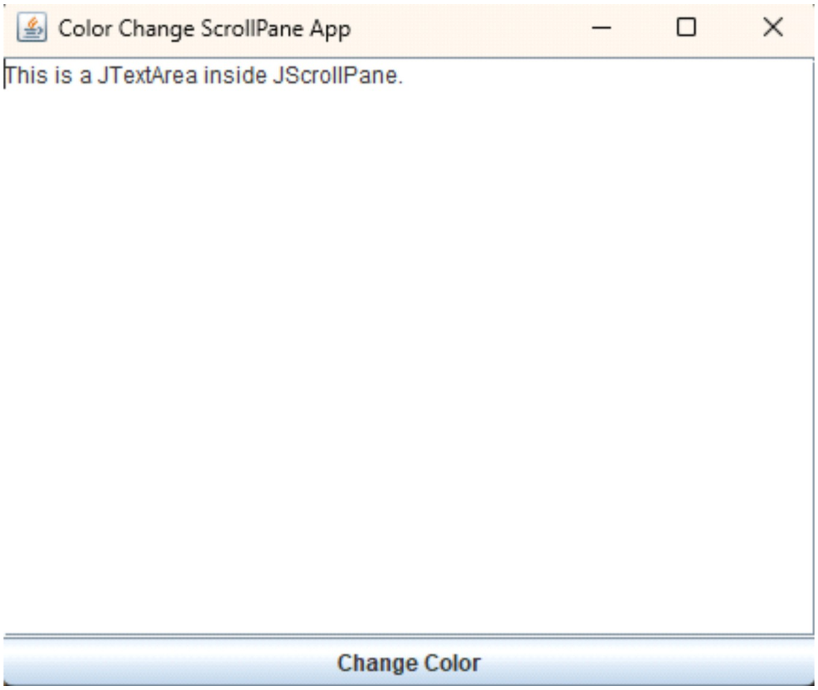
Code:

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

public class ColorChangeScrollPaneApp extends JFrame {
   private JScrollPane scrollPane;
   private JButton colorChooserButton;

   public ColorChangeScrollPaneApp() {
      // Set up the frame
      setTitle("Color Change ScrollPane App");
      setSize(400, 300);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setLocationRelativeTo(null); // Center the window

      // Create a JTextArea inside JScrollPane
      JTextArea textArea = new JTextArea("This is a JTextArea inside JScrollPane.");
      textArea.setLineWrap(true);
      textArea.setWrapStyleWord(true);

      // Create a JScrollPane containing the JTextArea
      scrollPane = new JScrollPane(textArea);

      // Create a button to change color
      colorChooserButton = new JButton("Change Color");
      colorChooserButton.addActionListener(new ActionListener() {
         public void actionPerformed(ActionEvent e) {
            changeBackgroundColor();
         }
      });

      // Set up the layout and add components
      setLayout(new BorderLayout());
      add(scrollPane, BorderLayout.CENTER);
      add(colorChooserButton, BorderLayout.SOUTH);

      // Display the frame
      setVisible(true);
   }

   // Method to change the background color using JColorChooser
   private void changeBackgroundColor() {
      // Show color chooser dialog
      Color selectedColor = JColorChooser.showDialog(this, "Choose Background Color",
scrollPane.getBackground());
    if (selectedColor != null) {
      // Set the selected color as the background color of the viewport
      scrollPane.getViewport().setBackground(selectedColor);
    }
  }

  // Main method to start the application
  public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
      public void run() {
        new ColorChangeScrollPaneApp();
      }
    });
  }
}
```

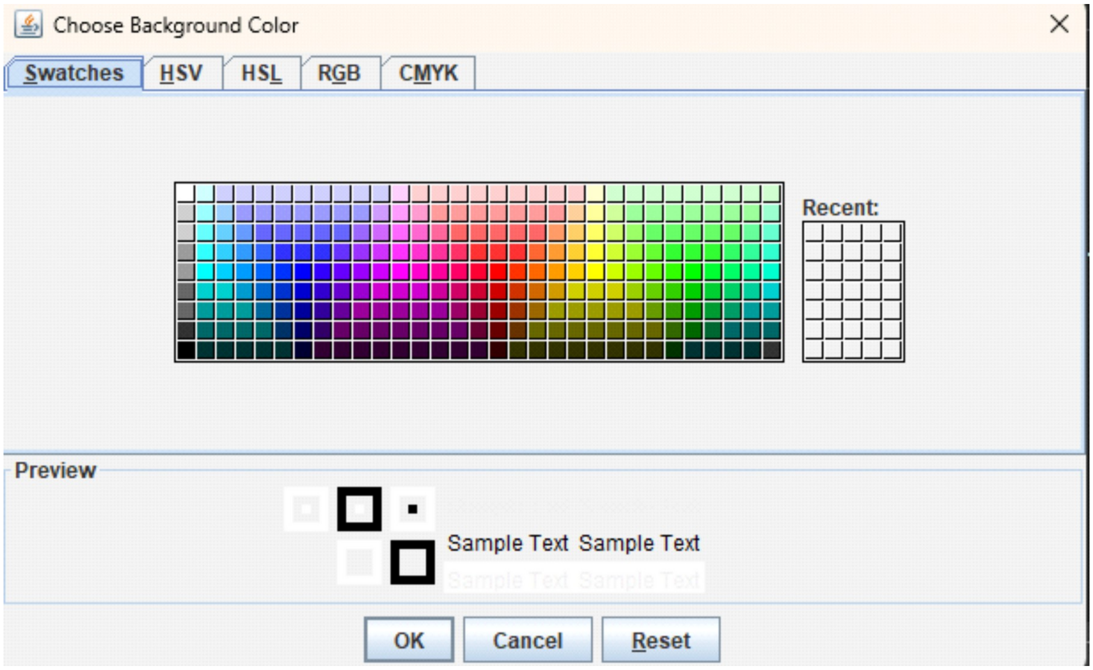
Output:



```
C:\DGET>javac ColorChangeScrollPaneApp.java

C:\DGET>java ColorChangeScrollPaneApp
```



This is a JTextArea inside JScrollPane.



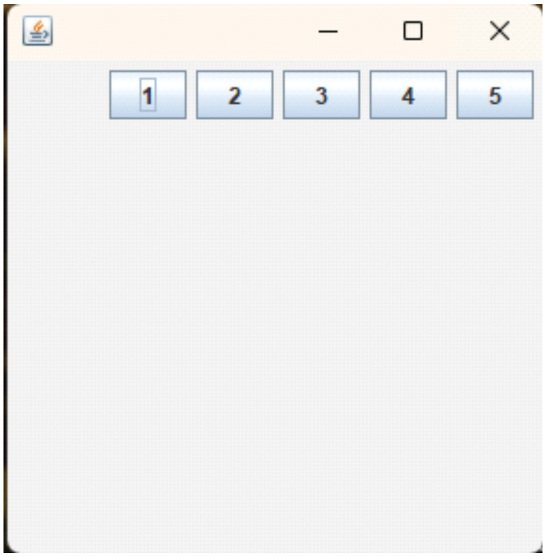
Practical no:-07 Layouts

Aim:a). Flow Layout Example

Code:

```
import java.awt.*;
import javax.swing.*;
public class MyFlowLayout
{
   JFrame f=new JFrame();
   MyFlowLayout()
   {
      JButton b1=new JButton("1");
      JButton b2=new JButton("2");
      JButton b3=new JButton("3");
      JButton b4=new JButton("4");
      JButton b5=new JButton("5");
// adding buttons to the frame
      f.add(b1); f.add(b2); f.add(b3); f.add(b4); f.add(b5);
// setting flow layout of right alignment
      f.setLayout(new FlowLayout(FlowLayout.RIGHT));
      f.setSize(300,300);
      f.setVisible(true);
   }
   public static void main(String[] args)
   {
      new MyFlowLayout();
   }
}
```

Output:



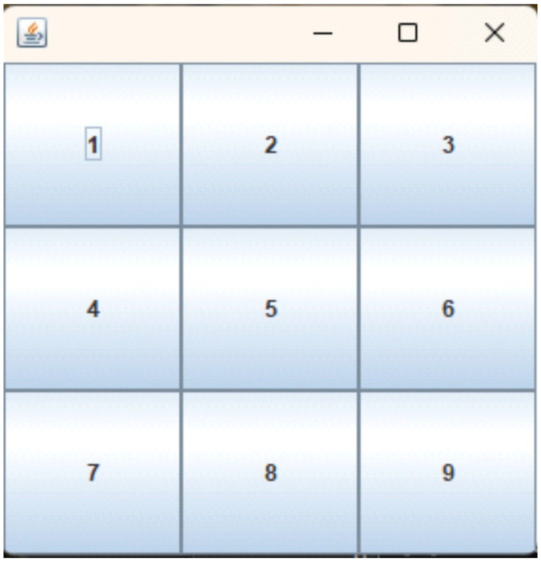
Aim:b). Grid layout Example

Code:

```
import java.awt.*;
import javax.swing.*;
public class MyGridLayout
{
   MyGridLayout(){
      JFrame f =new JFrame();
      JButton b1=new JButton("1");
      JButton b2=new JButton("2");
      JButton b3=new JButton("3");
      JButton b4=new JButton("4");
      JButton b5=new JButton("5");
      JButton b6=new JButton("6");
      JButton b7=new JButton("7");
      JButton b8=new JButton("8");
      JButton b9=new JButton("9");
// adding buttons to the frame
      f.add(b1); f.add(b2); f.add(b3);
      f.add(b4); f.add(b5); f.add(b6);
      f.add(b7); f.add(b8); f.add(b9);

      f.setLayout(new GridLayout(3,3));
// f.setLayout(new GridLayout(3, 3, 20, 25));
//Different Constructor
      f.setSize(300,300);
      f.setVisible(true);
   }
   public static void main(String[] args)
   {
      new MyGridLayout();
   }
```

}

Output:

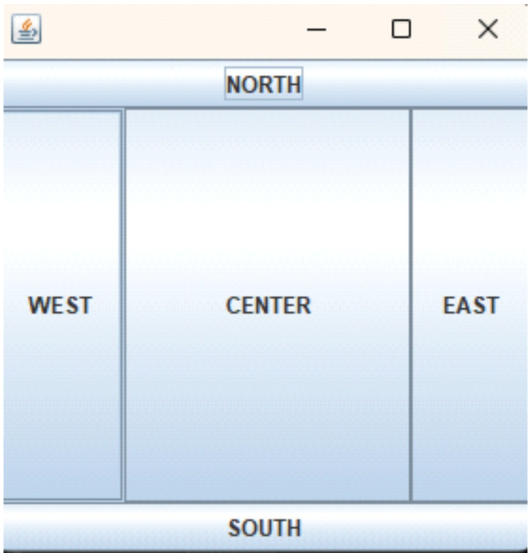


Aim:c). Border layout Example

Code:

```
import java.awt.*;
import javax.swing.*;
public class Border
{
   Border()
   {
      JFrame f = new JFrame();
      JButton b1 = new JButton("NORTH");;
      JButton b2 = new JButton("SOUTH");;
      JButton b3 = new JButton("EAST");;
      JButton b4 = new JButton("WEST");;
      JButton b5 = new JButton("CENTER");;

      f.add(b1, BorderLayout.NORTH);
      f.add(b2, BorderLayout.SOUTH);
      f.add(b3, BorderLayout.EAST);
      f.add(b4, BorderLayout.WEST);
      f.add(b5, BorderLayout.CENTER);
      f.setSize(300, 300);
      f.setVisible(true);
   }
   public static void main(String[] args) {
      new Border();
   } }
```

Output:



Practical no:-08 Events

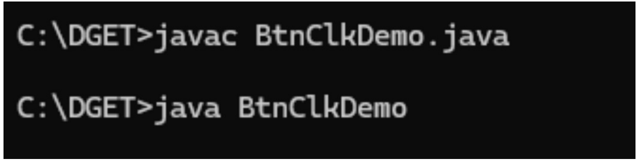
Aim:a). Action Event

I.Action Event with Button Click

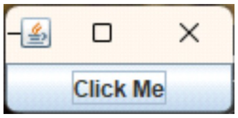
CODE:

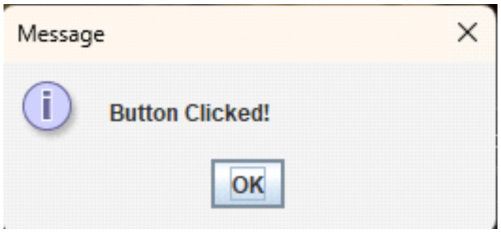
```
import javax.swing.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class BtnClkDemo {

public static void main(String[] args) {

JFrame frame = new JFrame("Button Click Demo");
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

JButton button = new JButton("Click Me");

button.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) { JOptionPane.showMessageDialog(frame, "Button
Clicked!"); }

});

frame.getContentPane().add(button);

frame.pack();

frame.setVisible(true);
}
}
```

Output:



```
C:\DGET>javac BtnClkDemo.java

C:\DGET>java BtnClkDemo
```





Button Clicked!

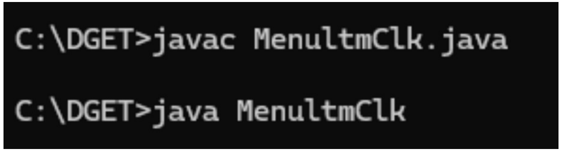
II.. Action Event with Menu Item

Code:

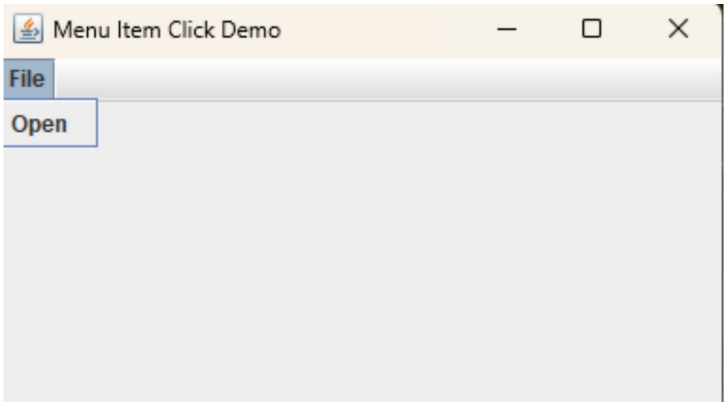
```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class MenuItmClk {
   public static void main(String[] args) {
      JFrame frame = new JFrame("Menu Item Click Demo");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // Create a menu bar
      JMenuBar menuBar = new JMenuBar();

      // Create "File" menu
      JMenu fileMenu = new JMenu("File");

      // Create "Open" menu item
      JMenuItem openItem = new JMenuItem("Open");

      // Add action listener to "Open" item
      openItem.addActionListener(new ActionListener() {
         @Override
         public void actionPerformed(ActionEvent e) {
            JOptionPane.showMessageDialog(frame, "File -> Open clicked!");
         }
      });

      // Add the "Open" item to the "File" menu
      fileMenu.add(openItem);

      // Add the "File" menu to the menu bar
      menuBar.add(fileMenu);
```

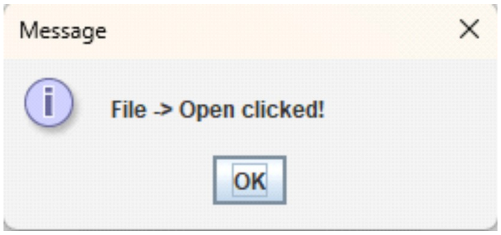
```
        // Set the menu bar for the frame
        frame.setJMenuBar(menuBar);

        // Set the size and make the frame visible
        frame.setSize(400, 300);
        frame.setVisible(true);
    }
}
```

Output:



```
C:\DGET>javac MenuItmClk.java

C:\DGET>java MenuItmClk
```





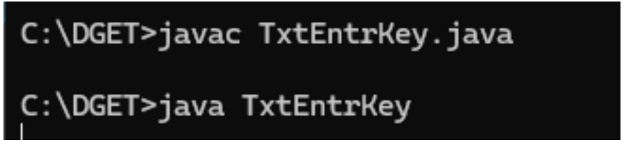
File -> Open clicked!

III. Action Event with text Field Enter Key Event

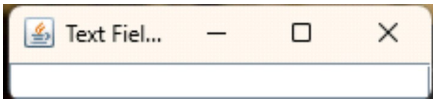
CODE

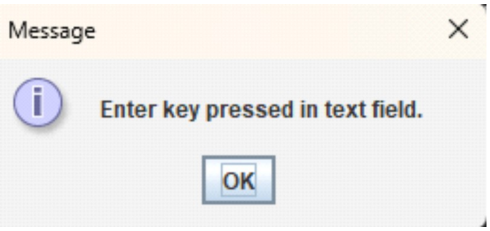
```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class TxtEntrKey {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Text Field Enter Key Demo");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Corrected the syntax here
        JTextField textField = new JTextField(20);
        textField.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(frame, "Enter key pressed in text field.");
            }
        });

        frame.getContentPane().add(textField);
        frame.pack();
        frame.setVisible(true);
    }
}
```

Output:



```
C:\DGET>javac TxtEntrKey.java

C:\DGET>java TxtEntrKey
```





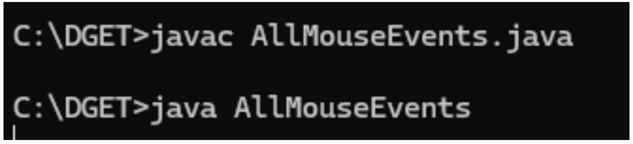
Enter key pressed in text field.

Aim:b). MouseEvent :Program to demostrate MouseClick , MouseMotion and mouseEnter/Exit events.

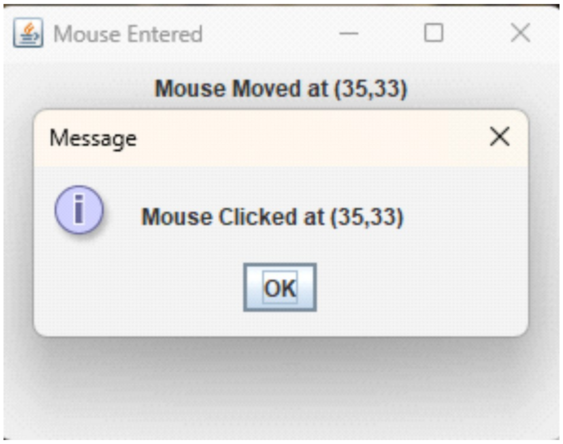
Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;

public class AllMouseEvents {
    public static void main(String[] args) {
        // Run on the Event Dispatch Thread
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("MouseClick Event Demo");
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

            // Create a JPanel and JLabel
            JPanel panel = new JPanel();
            JLabel jl = new JLabel();
            panel.add(jl);
            panel.setPreferredSize(new Dimension(300, 200));

            // Handling Mouse Click Event
            panel.addMouseListener(new MouseAdapter() {
                @Override
                public void mouseClicked(MouseEvent e) {
                    JOptionPane.showMessageDialog(frame, "Mouse Clicked at (" + e.getX() + "," + e.getY()
+ ")");
                }
            });

            // Handling Mouse Motion Event
            panel.addMouseMotionListener(new MouseAdapter() {
                @Override
                public void mouseMoved(MouseEvent e) {
                    jl.setText("Mouse Moved at (" + e.getX() + "," + e.getY() + ")");
                }
            });

            // Handling Mouse Enter and Exit Event
            panel.addMouseListener(new MouseAdapter() {
                @Override
                public void mouseEntered(MouseEvent e) {
                    frame.setTitle("Mouse Entered");
                }

                @Override
                public void mouseExited(MouseEvent e) {
                    frame.setTitle("Mouse Exited");
                }
            });

            // Add panel to frame
            frame.add(panel);
            frame.pack();
            frame.setVisible(true);
        });
    }
}
```

Output:



```
C:\DGET>javac AllMouseEvents.java

C:\DGET>java AllMouseEvents
```



Mouse Moved at (35,33)

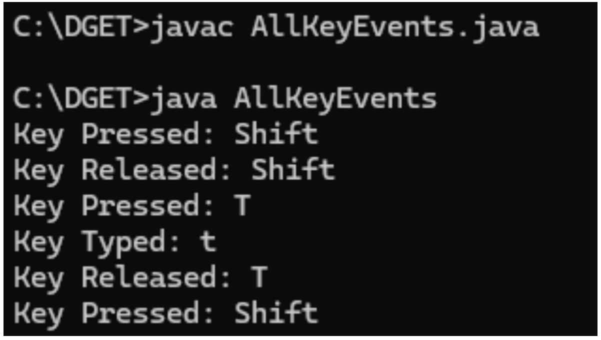
Mouse Clicked at (35,33)

Aim:c). KeyEvent: Program to demostrates how to handle KeyEvents (key presses and key releases):

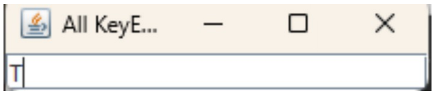
Code:

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

public class AllKeyEvents {
    public static void main(String[] args) {
        JFrame frame = new JFrame("All KeyEvents");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Create a JTextField
        JTextField textField = new JTextField(20);

        // Add the text field to the frame
        frame.add(textField);

        // Add a KeyListener to the text field
        textField.addKeyListener(new KeyAdapter() {
            @Override
            public void keyTyped(KeyEvent e) {
                System.out.println("Key Typed: " + e.getKeyChar());
            }

            @Override
            public void keyPressed(KeyEvent e) {
                System.out.println("Key Pressed: " + KeyEvent.getKeyText(e.getKeyCode()));
            }

            @Override
            public void keyReleased(KeyEvent e) {
                System.out.println("Key Released: " + KeyEvent.getKeyText(e.getKeyCode()));
            }
        });

        // Pack the frame and make it visible
        frame.pack();
        frame.setVisible(true);
    }
}
```

Output:



```
C:\DGET>javac AllKeyEvents.java

C:\DGET>java AllKeyEvents
Key Pressed: Shift
Key Released: Shift
Key Pressed: T
Key Typed: t
Key Released: T
Key Pressed: Shift
```



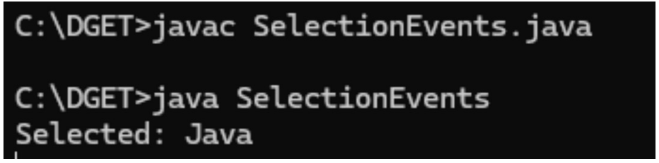
```
T
```

Aim:c). SelectionEvent

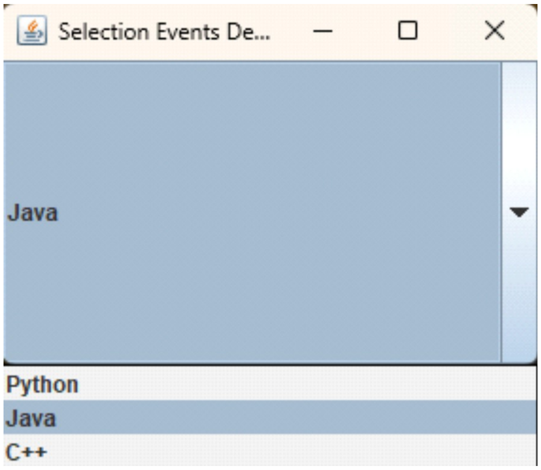
Code:

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

public class SelectionEvents {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            JFrame frame = new JFrame("Selection Events Demo");  // Fixed string
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

            // Correctly defined items
            String[] items = {"Python", "Java", "C++"};

            // Correct type for comboBox and initialization
            JComboBox<String> comboBox = new JComboBox<>(items);

            // Adding ActionListener to comboBox
            comboBox.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                    String selectedItem = (String) comboBox.getSelectedItem();
                    System.out.println("Selected: " + selectedItem);  // Fixed string concatenation
                }
            });

            // Adding comboBox to frame with BorderLayout.CENTER
            frame.add(comboBox, BorderLayout.CENTER);

            // Setting frame size and visibility
            frame.setSize(300, 200);
            frame.setVisible(true);
        });
    }
}
```

Output:



```
C:\DGET>javac SelectionEvents.java

C:\DGET>java SelectionEvents
Selected: Java
```

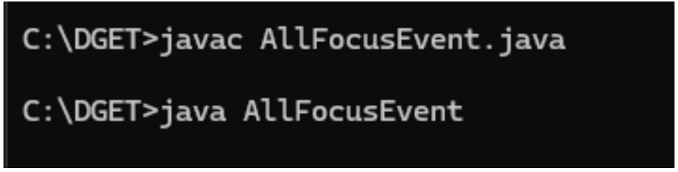


Aim:e). Focus Event:

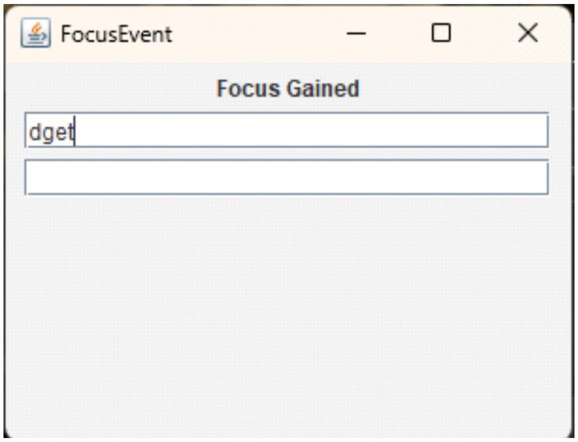
Code:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.FocusEvent;
import java.awt.event.FocusListener;
public class AllFocusEvent
{
    public static void main(String[] args)
{
    SwingUtilities.invokeLater(() -> {
        JFrame frame = new JFrame("FocusEvent");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        panel.setPreferredSize(new Dimension(300,200));
        JLabel label = new JLabel();
        JTextField textField1= new JTextField(25);
        JTextField textField2= new JTextField(25);
        textField1.addFocusListener(new FocusListener()
        {
            @Override
            public void focusGained(FocusEvent e)
        {
            label.setText("Focus Gained");
        }
            @Override
            public void focusLost(FocusEvent e)
        {
            label.setText("Focus Lost");
        }
        });
        panel.add(label);
        panel.add(textField1);
        panel.add(textField2);
        frame.add(panel);
        frame.pack();
        frame.setVisible(true);
    });
}
}
```

Output:

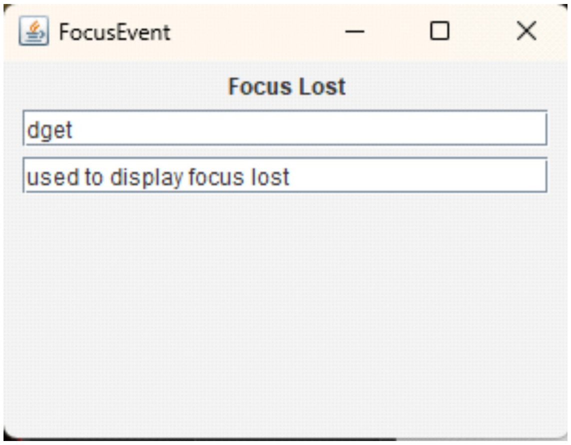


```
C:\DGET>javac AllFocusEvent.java

C:\DGET>java AllFocusEvent
```



Focus Gained

dget



Focus Lost

dget

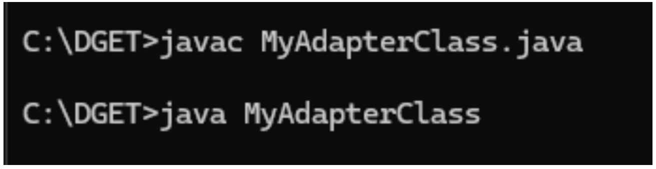
used to display focus lost

Practical no:-09   Demostrate the use of Adapter Class in event hangling

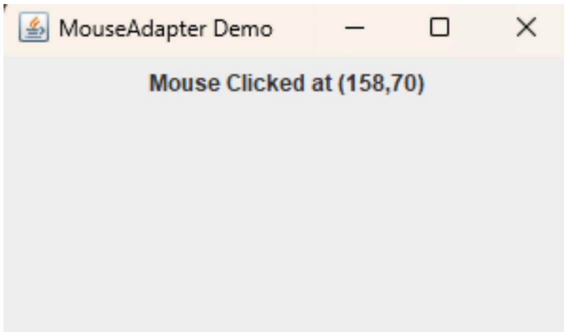
Code:

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

public class MyAdapterClass{
  public static void main(String[] args) {
    SwingUtilities.invokeLater(()->{
      JFrame frame = new JFrame("MouseAdapter Demo");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      JLabel label = new JLabel();
      JPanel panel = new JPanel();
      panel.setPreferredSize(new Dimension(300,200));

      panel.addMouseListener(new MouseAdapter() {
        @Override
        public void mouseClicked(MouseEvent e) {
          label.setText("Mouse Clicked at ("+e.getX()+","+e.getY()+")");
        }
      });
      panel.add(label);
      frame.add(panel);
      frame.pack();
      frame.setVisible(true);
    });
  }

}
```

Output:



```
C:\DGET>javac MyAdapterClass.java

C:\DGET>java MyAdapterClass
```



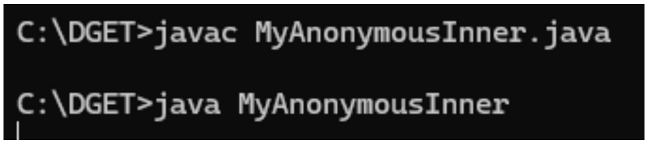
```
Mouse Clicked at (158,70)
```

Practical no:-10 Demostrate the use of Anonymous Inner Class in event hangling

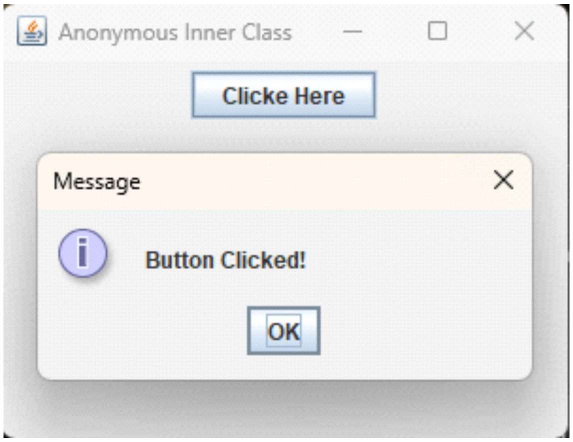
Code:

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

public class MyAnonymousInner{
  public static void main(String[] args) {
    SwingUtilities.invokeLater(()->{
      JFrame frame = new JFrame("Anonymous Inner Class");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      JPanel panel = new JPanel();
      panel.setPreferredSize(new Dimension(300,200));

      JButton button= new JButton("Clicke Here");
      button.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
          JOptionPane.showMessageDialog(frame,"Button Clicked!");
        }
      });
      panel.add(button);
      frame.add(panel);
      frame.pack();
      frame.setVisible(true);
    });
  }

}
```

Output:



```
C:\DGET>javac MyAnonymousInner.java

C:\DGET>java MyAnonymousInner
```



Message

Button Clicked!