1. Implement a program that demonstrates program structure of java with use of arithmetical and 21 logical implementations

```
// Demonstration of Java Program Structure with Arithmetic and Logical Operators
class ProgramStructureDemo {
    // main method - entry point of the program
    public static void main(String[] args) {
            // Arithmetic implementation
        int a = 21, b = 7;   // variables
        int sum = a + b;     // addition
        int diff = a - b;    // subtraction
        int product = a * b; // multiplication
        int quotient = a / b; // division
        int remainder = a % b; // modulus
        System.out.println("Arithmetic Operations:");
        System.out.println("a + b = " + sum);
        System.out.println("a - b = " + diff);
        System.out.println("a * b = " + product);
        System.out.println("a / b = " + quotient);
        System.out.println("a % b = " + remainder);
        // Logical implementation
        boolean condition1 = (a > b);     // true
        boolean condition2 = (a < 20);    // false
        System.out.println("\nLogical Operations:");
        System.out.println("a > b AND a < 20: " + (condition1 && condition2)); // AND
        System.out.println("a > b OR a < 20: " + (condition1 || condition2));  // OR
        System.out.println("NOT (a > b): " + (!condition1));                   // NOT
    }
}
```

*****OUTPUT*****

```
a + b = 28

a - b = 14

a * b = 147

a / b = 3
```

```
a % b = 0
Logical Operations:
a > b AND a < 20: false
a > b OR a < 20: true
NOT (a > b): false
```

2. Implement a program that demonstrates string operations using String and String Buffer class.

```
// Demonstration of String and StringBuffer Operations
class StringOperationsDemo {
  public static void main(String[] args) {
        // ---------- Using String class ----------
    String str1 = "Hello";
    String str2 = "World";
    System.out.println("String Operations:");
    System.out.println("Concatenation: " + str1.concat(" " + str2)); // Concatenation
    System.out.println("Length of str1: " + str1.length());          // Length
    System.out.println("Character at index 1 in str1: " + str1.charAt(1)); // Character extraction
    System.out.println("Substring of str2 (0,3): " + str2.substring(0, 3)); // Substring
    System.out.println("Uppercase str1: " + str1.toUpperCase());     // Uppercase
    System.out.println("Lowercase str2: " + str2.toLowerCase());     // Lowercase
    System.out.println("Check if str1 equals str2: " + str1.equals(str2)); // Comparison
    // ---------- Using StringBuffer class ----------
    StringBuffer sb = new StringBuffer("Hello");
    System.out.println("\nStringBuffer Operations:");
    sb.append(" World");                // Append
    System.out.println("After append: " + sb);
    sb.insert(6, "Java ");              // Insert
    System.out.println("After insert: " + sb);
    sb.replace(6, 10, "C++");           // Replace
    System.out.println("After replace: " + sb);
    sb.delete(6, 10);                   // Delete
    System.out.println("After delete: " + sb);
    sb.reverse();                       // Reverse
    System.out.println("After reverse: " + sb);
  }
}
```

*****OUTPUT*****

```
Concatenation: Hello World
Length of str1: 5
Character at index 1 in str1: e
Substring of str2 (0,3): Wor
Uppercase str1: HELLO
Lowercase str2: world
Check if str1 equals str2: false

StringBuffer Operations:
After append: Hello World
After insert: Hello Java World
After replace: Hello C++ World
After delete: Hello World
After reverse: dlroW olleH
```

```
}
```

3. Implement a program that demonstrates inner class and static fields.

```
// Demonstration of Inner Class and Static Fields
class OuterClass {
      // Static field (shared by all objects of OuterClass)
   static int objectCount = 0;
   // Constructor
   OuterClass() {
      objectCount++; // Increase count every time an object is created
   }
   // Inner class (non-static)
   class InnerClass {
      void displayMessage() {
         System.out.println("Hello from Inner Class!");
      }
   }
   // Method to show static field
   void showObjectCount() {
      System.out.println("Number of OuterClass objects created: " + objectCount);
   }
}
public class InnerClassDemo {
   public static void main(String[] args) {
            // Create OuterClass objects
      OuterClass obj1 = new OuterClass();
      OuterClass obj2 = new OuterClass();
      // Access static field using method
      obj1.showObjectCount(); // Shows 2 because two objects created
      // Create InnerClass object
      OuterClass.InnerClass inner = obj1.new InnerClass();
      inner.displayMessage(); // Calling method of inner class
   }
}
```

OUTPUT

```
Number of OuterClass objects created: 2

Hello from Inner Class!
```

4. Implement a program that demonstrate inheritance, polymorphism.

```
// Base class (Parent class)
class Animal {
    // Method in parent class
    void sound() {
        System.out.println("Animal makes a sound");
    }
}
// Derived class (Child class) - Inheriting Animal
class Dog extends Animal {
    // Method Overriding - same method as parent but different behavior
    @Override
    void sound() {
        System.out.println("Dog barks");
    }
}
// Another derived class (Child class)
class Cat extends Animal {
    // Method Overriding - provides its own version of sound()
    @Override
    void sound() {
        System.out.println("Cat meows");
    }
}
// Main class
public class InheritancePolymorphismDemo {
    public static void main(String[] args) {
            // Polymorphism Example:
        // Parent class reference pointing to child class object
        Animal a1 = new Dog();
        Animal a2 = new Cat();
        // Calls the overridden method from child classes (runtime polymorphism)
        a1.sound();  // Output: Dog barks
        a2.sound();  // Output: Cat meows
        // Normal object creation of parent class
        Animal generic = new Animal();
        generic.sound(); // Output: Animal makes a sound
    }
}
```

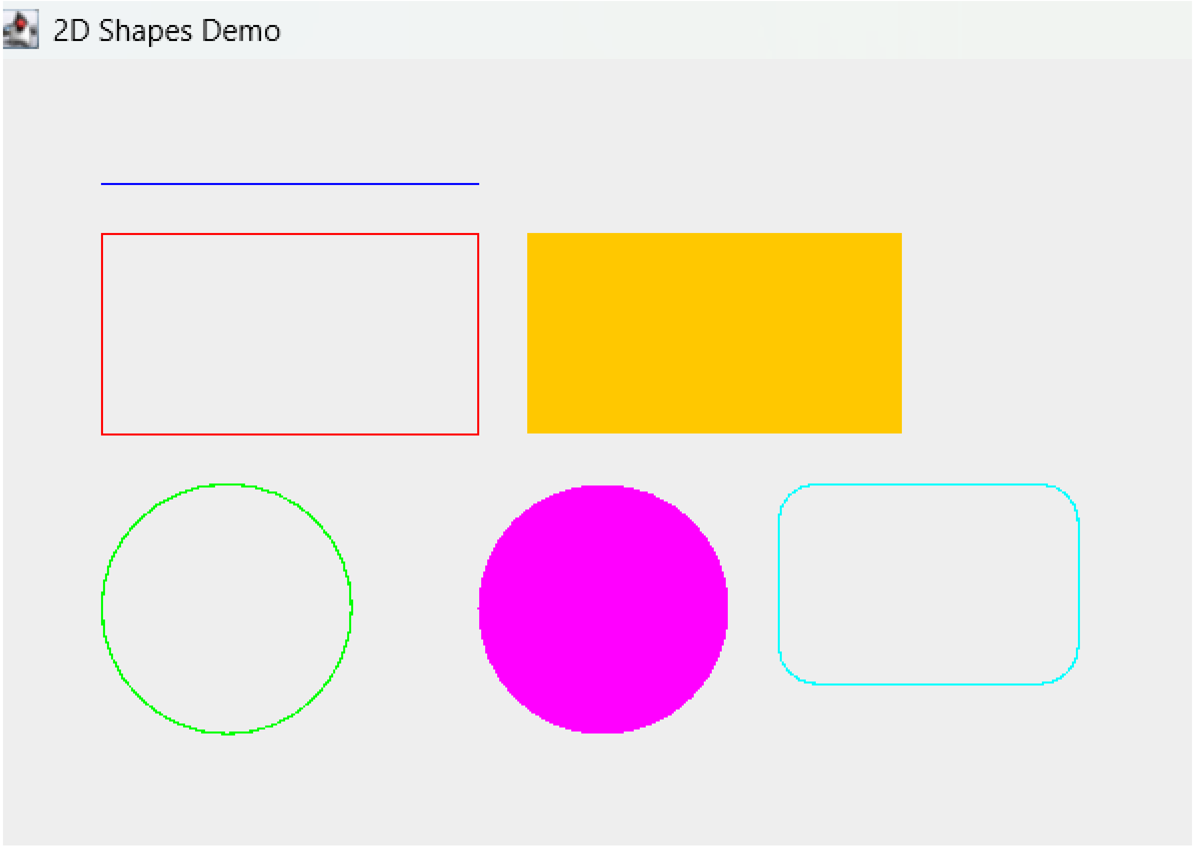
OUTPUT

```
Dog barks
Cat meows
Animal makes a sound
```

5. Implement a program that demonstrates 2D shapes on frames.

```
/*
Program: Demonstration of 2D Shapes on Frame
MCA Student
*/
// For Graphics
import java.awt.Color;
import java.awt.Graphics;
// For JFrame
import javax.swing.JFrame;
// Main class extending JFrame
public class ShapesOnFrame extends JFrame {
    // Constructor to set up the frame
    ShapesOnFrame() {
        setTitle("2D Shapes Demo");   // Title of frame
        setSize(500, 400);            // Width x Height
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);             // Make frame visible
    }
    // Override paint() method to draw shapes
    @Override
    public void paint(Graphics g) {
        super.paint(g);
        // Set drawing color
        g.setColor(Color.BLUE);
        // Draw Line
        g.drawLine(50, 80, 200, 80);
        // Draw Rectangle
        g.setColor(Color.RED);
        g.drawRect(50, 100, 150, 80);
        // Draw Filled Rectangle
        g.setColor(Color.ORANGE);
        g.fillRect(220, 100, 150, 80);
        // Draw Oval (Circle-like)
        g.setColor(Color.GREEN);
        g.drawOval(50, 200, 100, 100);
        // Draw Filled Oval
        g.setColor(Color.MAGENTA);
        g.fillOval(200, 200, 100, 100);
        // Draw Round Rectangle
        g.setColor(Color.CYAN);
        g.drawRoundRect(320, 200, 120, 80, 30, 30);
    }
    // Main method
    public static void main(String[] args) {
        new ShapesOnFrame();  // Create frame object
    }
}
```

*****OUTPUT*****

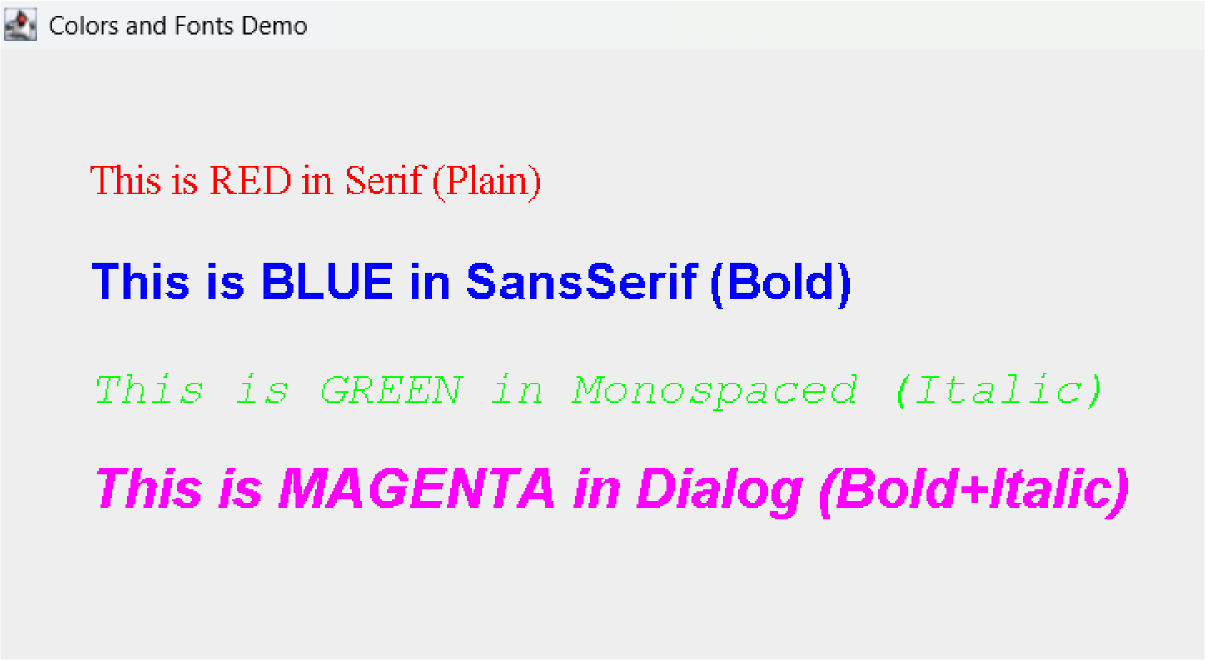


6. Implement a program that demonstrates color and fonts.

```
/*
 Program: Demonstration of Colors and Fonts in Java
 Author : MCA Student
*/
import java.awt.Color;
import java.awt.Font;
import java.awt.Graphics;
import javax.swing.JFrame;

// Main class extending JFrame
public class ColorsAndFontsDemo extends JFrame {
   // Constructor to set up the frame
   ColorsAndFontsDemo() {
      setTitle("Colors and Fonts Demo");   // Frame title
      setSize(500, 300);                   // Frame size
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      setVisible(true);                    // Show frame
   }
   // Override paint() method to draw text with colors and fonts
   @Override
   public void paint(Graphics g) {
      super.paint(g);
      // Set Font and Color for first text
      g.setFont(new Font("Serif", Font.PLAIN, 20)); // Serif font, normal style, size 20
      g.setColor(Color.RED);                        // Red color
      g.drawString("This is RED in Serif (Plain)", 50, 100);
      // Second text with Bold
      g.setFont(new Font("SansSerif", Font.BOLD, 24)); // SansSerif, bold, size 24
      g.setColor(Color.BLUE);                          // Blue color
      g.drawString("This is BLUE in SansSerif (Bold)", 50, 150);
```

```
        // Third text with Italic
        g.setFont(new Font("Monospaced", Font.ITALIC, 22)); // Monospaced, italic
        g.setColor(Color.GREEN);                        // Green color
        g.drawString("This is GREEN in Monospaced (Italic)", 50, 200);

        // Fourth text with Bold + Italic
        g.setFont(new Font("Dialog", Font.BOLD | Font.ITALIC, 26));
        g.setColor(Color.MAGENTA);
        g.drawString("This is MAGENTA in Dialog (Bold+Italic)", 50, 250);
    }
    // Main method
    public static void main(String[] args) {
        new ColorsAndFontsDemo(); // Create frame object
    }
}
```

OUTPUT



Colors and Fonts Demo

This is RED in Serif (Plain)

This is BLUE in SansSerif (Bold)

This is GREEN in Monospaced (Italic)

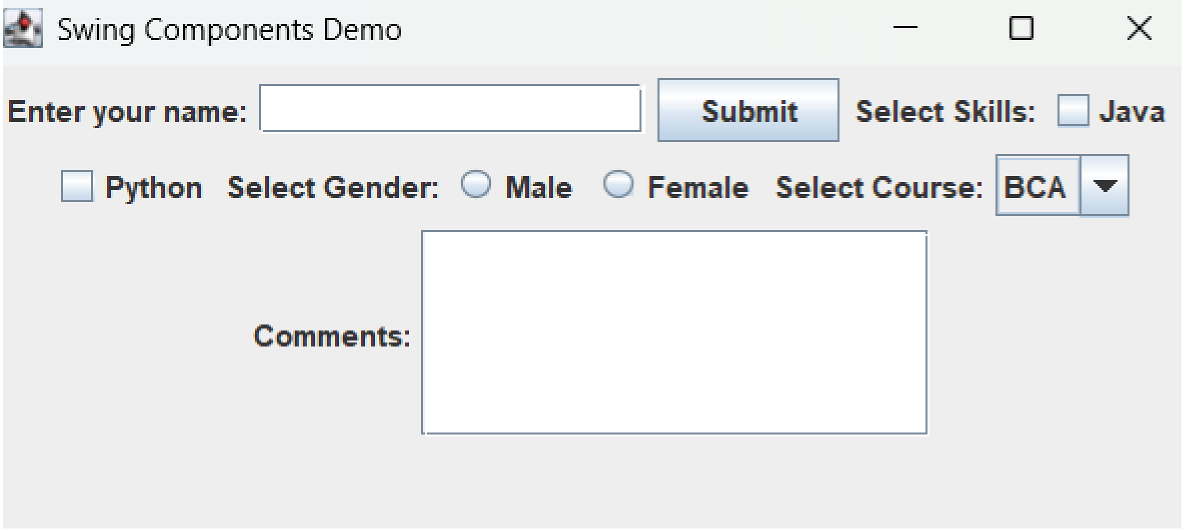
This is MAGENTA in Dialog (Bold+Italic)

7. Implement a program to illustrate use of various swing components

```
/*
 Program: Demonstration of various Swing components
 Author : MCA Student

 ***Explanation of Components***
JLabel → Displays text ("Enter your name:").
JTextField → Single-line input.
JButton → A button labeled "Submit".
JCheckBox → Multiple options (Java, Python).
JRadioButton + ButtonGroup → One option must be selected (Male, Female).
JComboBox → Drop-down menu (BCA, MCA, MBA, MSc).
JTextArea → Multi-line input.
JScrollPane → Adds scrollbars to text area.
*/

// Layout managers
import java.awt.FlowLayout;
// Import Swing classes
import javax.swing.ButtonGroup;
import javax.swing.JButton;
import javax.swing.JCheckBox;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JRadioButton;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
```

```
public class SwingComponentsDemo extends JFrame {
    // Constructor
    SwingComponentsDemo() {
        // Set frame properties
        setTitle("Swing Components Demo");
        setSize(500, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());  // Simple layout manager
        // JLabel
        JLabel label = new JLabel("Enter your name:");

        // JTextField
        JTextField textField = new JTextField(15);

        // JButton
        JButton button = new JButton("Submit");

        // JCheckBox
        JCheckBox check1 = new JCheckBox("Java");
        JCheckBox check2 = new JCheckBox("Python");

        // JRadioButton + ButtonGroup
        JRadioButton male = new JRadioButton("Male");
        JRadioButton female = new JRadioButton("Female");
        ButtonGroup genderGroup = new ButtonGroup();
        genderGroup.add(male);
        genderGroup.add(female);

        // JComboBox
        String courses[] = {"BCA", "MCA", "MBA", "MSc"};
        JComboBox<String> comboBox = new JComboBox<>(courses);
        // JTextArea
        JTextArea textArea = new JTextArea(5, 20);

        // JScrollPane for text area
        JScrollPane scrollPane = new JScrollPane(textArea);

        // Add components to frame
        add(label);
        add(textField);
        add(button);
        add(new JLabel("Select Skills:"));
        add(check1);
        add(check2);
        add(new JLabel("Select Gender:"));
        add(male);
        add(female);
        add(new JLabel("Select Course:"));
        add(comboBox);
        add(new JLabel("Comments:"));
        add(scrollPane);
        setVisible(true);
    }
    // Main method
    public static void main(String[] args) {
        new SwingComponentsDemo(); // Create frame object
    }
}
```

OUTPUT



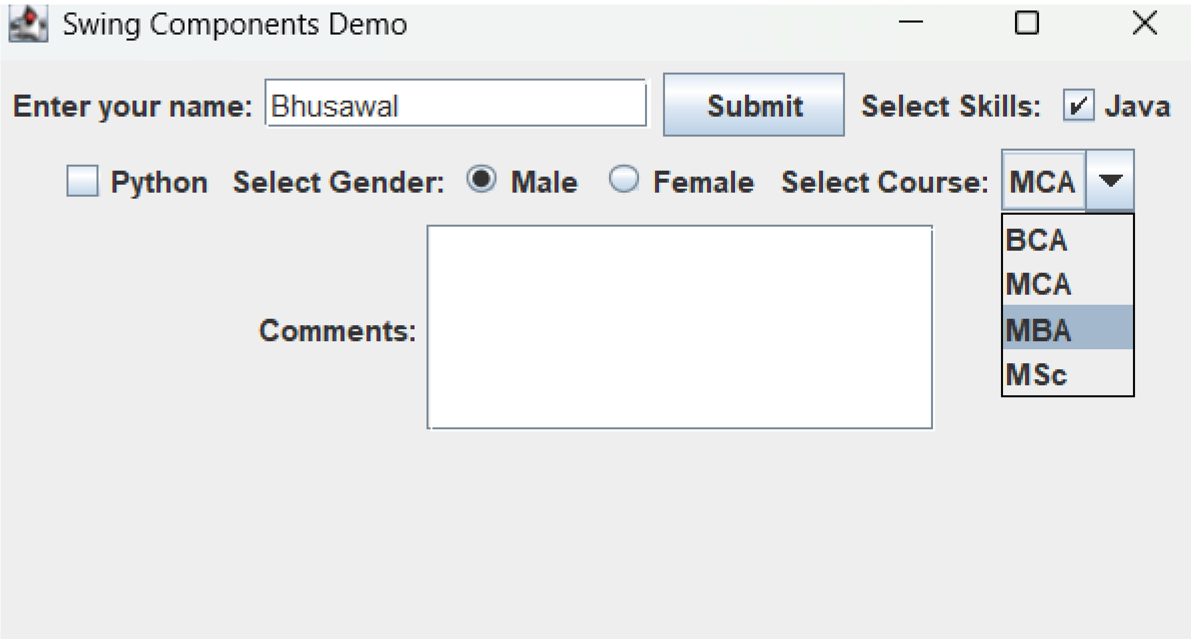
Enter your name:

Select Skills: □ Java □ Python

Select Gender: ○ Male ○ Female

Select Course: BCA

Comments:



Enter your name: Bhusawal

Select Skills: ☑ Java □ Python

Select Gender: ● Male ○ Female

Select Course: MCA

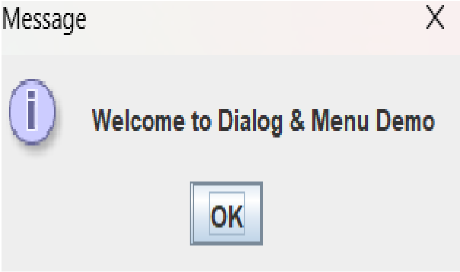
Comments:

8. Implement a program that demonstrates use of dialog box and menus.

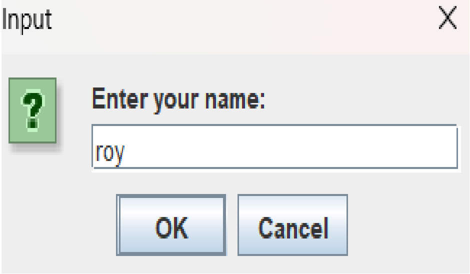
```
/*
 Program: Demonstration of Dialog Boxes and Menus
 Author : MCA Student
 */

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.JFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
public class DialogAndMenuDemo extends JFrame implements ActionListener {
   JMenuItem exitItem, aboutItem;

   // Constructor
   DialogAndMenuDemo() {
      setTitle("Dialog Box and Menu Demo");
      setSize(400, 300);
      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      // ---------- Menu Bar ----------
      JMenuBar menuBar = new JMenuBar();

      // File menu
      JMenu fileMenu = new JMenu("File");
      exitItem = new JMenuItem("Exit");
      exitItem.addActionListener(this); // Add event
      fileMenu.add(exitItem);
```

```
        // Help menu
        JMenu helpMenu = new JMenu("Help");
        aboutItem = new JMenuItem("About");
        aboutItem.addActionListener(this); // Add event
        helpMenu.add(aboutItem);

        // Add menus to menu bar
        menuBar.add(fileMenu);
        menuBar.add(helpMenu);
        setJMenuBar(menuBar);

        // ---------- Dialog Boxes ----------
        // Message Dialog
        JOptionPane.showMessageDialog(this, "Welcome to Dialog & Menu Demo", "Message", JOptionPane.INFORMATION_MESSAGE);

        // Input Dialog
        String name = JOptionPane.showInputDialog(this, "Enter your name:");
        JOptionPane.showMessageDialog(this, "Hello, " + name + "!", "Greetings", JOptionPane.INFORMATION_MESSAGE);

        // Confirm Dialog
        int response = JOptionPane.showConfirmDialog(this, "Do you like Java?", "Question", JOptionPane.YES_NO_OPTION);
        if(response == JOptionPane.YES_OPTION) {
            JOptionPane.showMessageDialog(this, "Great! Keep learning.", "Info", JOptionPane.INFORMATION_MESSAGE);
        } else {
            JOptionPane.showMessageDialog(this, "No worries! Explore other languages.", "Info", JOptionPane.INFORMATION_MESSAGE);
        }

        setVisible(true);
    }
    // Handle menu item clicks
    public void actionPerformed(ActionEvent e) {
        if(e.getSource() == exitItem) {
            System.exit(0); // Close application
        }
        if(e.getSource() == aboutItem) {
            JOptionPane.showMessageDialog(this, "Dialog & Menu Demo\nAuthor: MCA Student",
"About", JOptionPane.INFORMATION_MESSAGE);
        }
    }

    // Main method
    public static void main(String[] args) {
        new DialogAndMenuDemo();
    }
}
```

OUTPUT

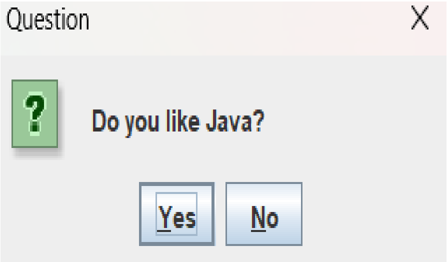


Welcome to Dialog & Menu Demo

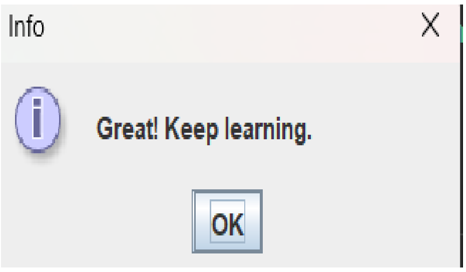


Enter your name:

roy



Do you like Java?

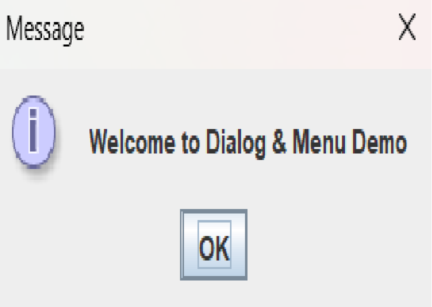


Great! Keep learning.

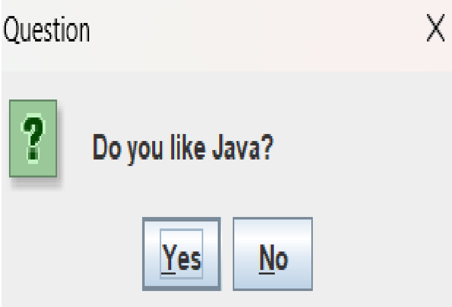
9. Implement a program that demonstrates event handling for various types of events.

```
/*
 Program: Demonstration of Event Handling
 Author : MCA Student
*/
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class EventHandlingDemo extends JFrame implements ActionListener, KeyListener,
MouseListener, WindowListener {

    JButton button;
    // Constructor
    EventHandlingDemo() {
        setTitle("Event Handling Demo");
        setSize(500, 400);
        setLayout(new FlowLayout());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // ---------- Button (ActionEvent) ----------
        button = new JButton("Click Me");
        button.addActionListener(this);  // Register ActionListener
        add(button);

        // ---------- TextField (KeyEvent) ----------
        JTextField textField = new JTextField(20);
        textField.addKeyListener(this);  // Register KeyListener
        add(new JLabel("Type something:"));
        add(textField);

        // ---------- Panel (MouseEvent) ----------
        JPanel panel = new JPanel();
        panel.setBackground(Color.LIGHT_GRAY);
        panel.setPreferredSize(new Dimension(200, 100));
        panel.addMouseListener(this);    // Register MouseListener
        add(new JLabel("Click inside panel:"));
        add(panel);

        // ---------- WindowEvent ----------
        addWindowListener(this);         // Register WindowListener

        setVisible(true);
    }

    // --------- ActionEvent ---------
    public void actionPerformed(ActionEvent e) {
        if(e.getSource() == button) {
            JOptionPane.showMessageDialog(this, "Button clicked!", "Action Event",
JOptionPane.INFORMATION_MESSAGE);
        }
    }

    // --------- KeyEvent ---------
    public void keyTyped(KeyEvent e) {
        System.out.println("Key Typed: " + e.getKeyChar());
    }
    public void keyPressed(KeyEvent e) {
        System.out.println("Key Pressed: " + e.getKeyChar());
    }
    public void keyReleased(KeyEvent e) {
        System.out.println("Key Released: " + e.getKeyChar());
    }

    // --------- MouseEvent ---------
    public void mouseClicked(MouseEvent e) {
        JOptionPane.showMessageDialog(this, "Mouse clicked at (" + e.getX() + "," + e.getY() + ")",
"Mouse Event", JOptionPane.INFORMATION_MESSAGE);
    }
    public void mousePressed(MouseEvent e) {}
    public void mouseReleased(MouseEvent e) {}
    public void mouseEntered(MouseEvent e) {}
    public void mouseExited(MouseEvent e) {}

    // --------- WindowEvent ---------
    public void windowOpened(WindowEvent e) {
        System.out.println("Window Opened");
    }
    public void windowClosing(WindowEvent e) {
        System.out.println("Window Closing");
    }
    public void windowClosed(WindowEvent e) {
        System.out.println("Window Closed");
    }
    public void windowIconified(WindowEvent e) {}
    public void windowDeiconified(WindowEvent e) {}
    public void windowActivated(WindowEvent e) {}
    public void windowDeactivated(WindowEvent e) {}

    // Main method
    public static void main(String[] args) {
        new EventHandlingDemo();
    }
}
```

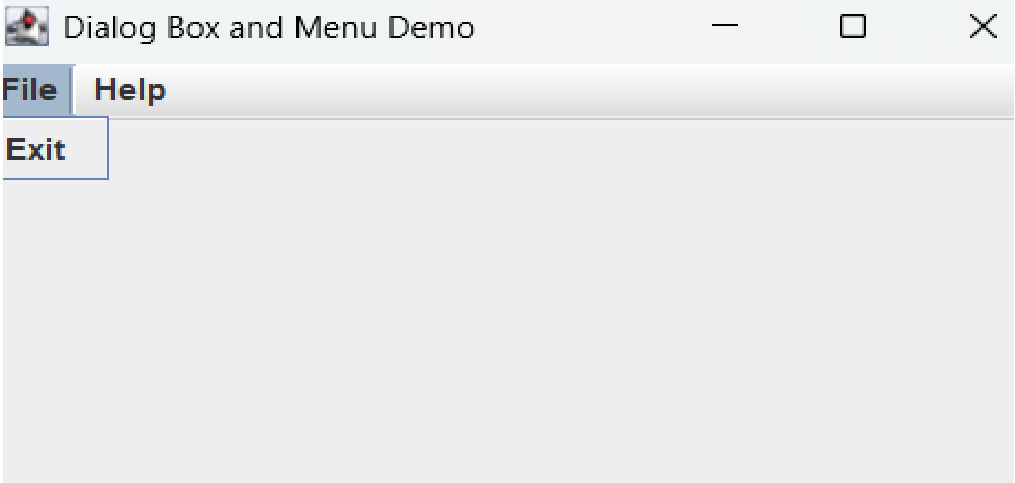
OUTPUT


Message

Welcome to Dialog & Menu Demo


Question

Do you like Java?


Dialog Box and Menu Demo

10. Implement a program to illustrate multithreading.

```
/*
 Program: Demonstration of Multithreading
 Author : MCA Student
 Description: This program shows two threads running simultaneously.
*/
class ThreadOne extends Thread {
   public void run() {
      // Loop to show thread activity
      for (int i = 1; i <= 5; i++) {
         System.out.println("ThreadOne: Count " + i);
         try {
            Thread.sleep(500);  // Pause for 0.5 seconds
         } catch (InterruptedException e) {
            System.out.println(e);
         }
      }
   }
}
class ThreadTwo implements Runnable {
   public void run() {
      // Loop to show thread activity
      for (int i = 1; i <= 5; i++) {
         System.out.println("ThreadTwo: Count " + i);
         try {
            Thread.sleep(700);  // Pause for 0.7 seconds
         } catch (InterruptedException e) {
            System.out.println(e);
         }
      }
   }
}
```

```
public class MCA_MultithreadingDemo {
    public static void main(String[] args) {
        System.out.println("Main Thread Started");

        // Create thread objects
        ThreadOne t1 = new ThreadOne();              // Thread by extending Thread
        Thread t2 = new Thread(new ThreadTwo());     // Thread by implementing Runnable

        // Start both threads
        t1.start();
        t2.start();
        System.out.println("Main Thread Running Simultaneously with Other Threads");
    }
}
```

*****OUTPUT*****

```
Main Thread Started
Main Thread Running Simultaneously with Other Threads
ThreadOne: Count 1
ThreadTwo: Count 1
ThreadOne: Count 2
ThreadTwo: Count 2
ThreadOne: Count 3
ThreadTwo: Count 3
ThreadOne: Count 4
ThreadOne: Count 5
ThreadTwo: Count 4
ThreadTwo: Count 5
```

11. Implement a program to illustrate exception handling.

```
/*
 Program: Demonstration of Exception Handling
 Author : MCA Student
 Description: This program illustrates try, catch, finally, and multiple exceptions.
*/
import java.util.Scanner;
public class ExceptionHandlingDemo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        try {
            // Take input from user
            System.out.print("Enter first number: ");
            int a = sc.nextInt();

            System.out.print("Enter second number: ");
            int b = sc.nextInt();

            // May cause ArithmeticException
            int result = a / b;
            System.out.println("Result of division = " + result);

            // Example of ArrayIndexOutOfBoundsException
            int arr[] = {10, 20, 30};
            System.out.println("Accessing 4th element: " + arr[3]);
        } catch (ArithmeticException e) {
            System.out.println("Error: Division by zero is not allowed.");
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: Array index is out of bounds.");
        } catch (Exception e) {
            System.out.println("General Exception: " + e);
        } finally {
            // Always executes
            System.out.println("Finally block executed. Closing resources...");
        }
        System.out.println("Program continues after exception handling...");
        sc.close();
    }
}
```

*****OUTPUT*****

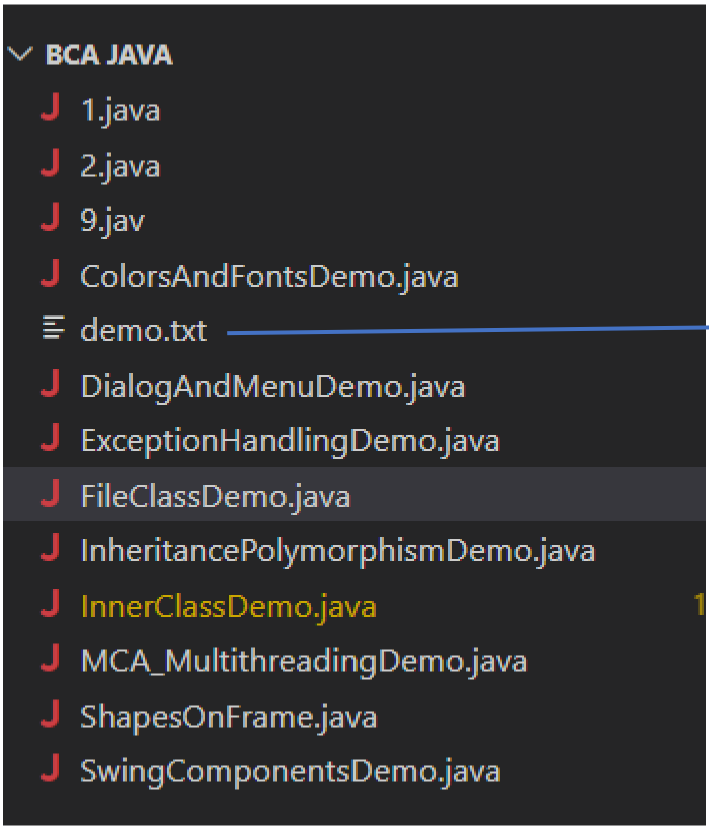
```
Enter first number: 4
Enter second number: 2
Result of division = 2
```

12. Implement a program to demonstrate use of File class,

```
/*
 Program: Demonstration of File Class in Java
 Author : MCA Student
 Description: This program shows how to use File class methods.
*/
import java.io.File;
import java.io.IOException;
public class FileClassDemo {
    public static void main(String[] args) {
        try {
            // Create File object
            File file = new File("demo.txt");

            // Create new file (if it doesn't exist)
            if (file.createNewFile()) {
                System.out.println("File created: " + file.getName());
            } else {
                System.out.println("File already exists.");
            }
            // File Information
            System.out.println("Absolute Path : " + file.getAbsolutePath());
            System.out.println("Can Write?    : " + file.canWrite());
            System.out.println("Can Read?     : " + file.canRead());
            System.out.println("Is File?      : " + file.isFile());
            System.out.println("Is Directory? : " + file.isDirectory());
            System.out.println("File Size     : " + file.length() + " bytes");
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
    }}
```

*****OUTPUT*****

```
File created: demo.txt
Absolute Path : C:\Users\OneDrive\Desktop\Nahata\BCA JAVA\demo.txt
Can Write?    : true
Can Read?     : true
Is File?      : true
Is Directory? : false
File Size     : 0 bytes
```



BCA JAVA
1.java
2.java
9.jav
ColorsAndFontsDemo.java
demo.txt
DialogAndMenuDemo.java
ExceptionHandlingDemo.java
FileClassDemo.java
InheritancePolymorphismDemo.java
InnerClassDemo.java 1
MCA_MultithreadingDemo.java
ShapesOnFrame.java
SwingComponentsDemo.java

USE OF FILE CLASS