1 Calculator

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

class Calculator extends JFrame implements ActionListener {
   private JTextField textField;
   private String s0, s1, s2;

   Calculator() {
      s0 = s1 = s2 = "";
      textField = new JTextField(16);
      textField.setEditable(false);
   }

   public static void main(String args[]) {
      SwingUtilities.invokeLater(() -> {
         Calculator calculator = new Calculator();
         calculator.createAndShowGUI();
      });
   }

   private void createAndShowGUI() {
      try {
         UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
      } catch (Exception e) {
         System.err.println(e.getMessage());
      }

      // Print your name at the top
      JLabel nameLabel = new JLabel("Om Singh SYCS 46", SwingConstants.CENTER);
```

```
nameLabel.setFont(new Font("Arial", Font.BOLD, 16));

JButton b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bs, bd, bm, be, beq, beq1;
b0 = new JButton("0");
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");
beq1 = new JButton("=");
ba = new JButton("+");
bs = new JButton("-");
bd = new JButton("/");
bm = new JButton("*");
beq = new JButton("C");
be = new JButton(".");

JButton[] buttons = {b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bs, bd, bm, beq, be, beq1};
for (JButton button : buttons) {
   button.addActionListener(this);
}

JPanel panel = new JPanel();
panel.setLayout(new GridLayout(6, 4));  // Adjusted layout to fit name label
panel.add(nameLabel);
panel.add(textField);
panel.add(ba);
```

```
    panel.add(bs);
    panel.add(bm);
    panel.add(bd);
    panel.add(b1);
    panel.add(b2);
    panel.add(b3);
    panel.add(b4);
    panel.add(b5);
    panel.add(b6);
    panel.add(b7);
    panel.add(b8);
    panel.add(b9);
    panel.add(b0);
    panel.add(be);
    panel.add(beq);
    panel.add(beq1);
    panel.setBackground(Color.PINK);

    getContentPane().add(panel);
    setSize(400, 400);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    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;
        }
        textField.setText(s0 + s1 + s2);
    } else if (s.charAt(0) == 'C') {
        s0 = s1 = s2 = "";
        textField.setText(s0 + s1 + s2);
    } else if (s.charAt(0) == '=') {
        double result = 0;
        try {
            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("/")) {
                if (Double.parseDouble(s2) == 0) {
                    throw new ArithmeticException("Cannot divide by zero");
                }
                result = Double.parseDouble(s0) / Double.parseDouble(s2);
            } else if (s1.equals("*")) {
                result = Double.parseDouble(s0) * Double.parseDouble(s2);
            }
            textField.setText(s0 + s1 + s2 + "=" + result);
            s0 = Double.toString(result);
            s1 = s2 = "";
        } catch (NumberFormatException | ArithmeticException ex) {
            textField.setText("Error: " + ex.getMessage());
            s0 = s1 = s2 = "";
        }
    } else {
        if (s1.equals("") || s2.equals("")) {
            s1 = s;
```

```
        } else {
            double result = 0;
            try {
                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("/")) {
                    if (Double.parseDouble(s2) == 0) {
                        throw new ArithmeticException("Cannot divide by zero");
                    }
                    result = Double.parseDouble(s0) / Double.parseDouble(s2);
                } else if (s1.equals("*")) {
                    result = Double.parseDouble(s0) * Double.parseDouble(s2);
                }
                s0 = Double.toString(result);
                s1 = s;
                s2 = "";
            } catch (NumberFormatException | ArithmeticException ex) {
                textField.setText("Error: " + ex.getMessage());
                s0 = s1 = s2 = "";
            }
        }
        textField.setText(s0 + s1 + s2);
    }
  }
}
```

2 Write a program to implement interfaces.

```
interface Vehicle {
  void changeGear(int a);
```

```
    void speedUp(int a);
    void applyBrakes(int a);
}

class Bicycle implements Vehicle {
    int speed;
    int gear;

    public void changeGear(int newGear) {
        gear = newGear;
    }

    public void speedUp(int increment) {
        speed += increment; // Using shorthand operator
    }

    public void applyBrakes(int decrement) {
        speed -= decrement; // Using shorthand operator
    }

    public void printStates() {
        System.out.println("\nBicycle - Speed: " + speed + ", Gear: " + gear); // Improved output
format
    }
}

class Bike implements Vehicle {
    int speed;
    int gear;

    public void changeGear(int newGear) {
```

```
        gear = newGear;
    }

    public void speedUp(int increment) {
        speed += increment; // Using shorthand operator
    }

    public void applyBrakes(int decrement) {
        speed -= decrement; // Using shorthand operator
    }

    public void printStates() {
        System.out.println("\nBike - Speed: " + speed + ", Gear: " + gear); // Improved output format
    }
}

public class GFG1 {
    public static void main(String[] args) {
        Bicycle bicycle = new Bicycle();
        bicycle.changeGear(2);
        bicycle.speedUp(3);
        bicycle.applyBrakes(1);

        System.out.println("\nBicycle present state: ");
        bicycle.printStates();

        Bike bike = new Bike();
        bike.changeGear(1);
        bike.speedUp(4);
        bike.applyBrakes(3);
```

```
        System.out.println("\nBike present state: ");
        bike.printStates();
    }
}
```

3 inheritance and method overriding

```
class Car {
    public Car() {
        System.out.println("Class Car");
    }

    public void vehicletype() {
        System.out.println("Vehicle type: Car");
    }
}

class Maruti extends Car {
    public Maruti() {
        System.out.println("Class Maruti");
    }

    public void brand() {
        System.out.println("Brand is Maruti");
    }

    public void speed1() {
        System.out.println("Max. Speed: 90 kmph");
    }
}

class Maruti800 extends Maruti {
```

```
  public Maruti800() {
    System.out.println("Class Maruti800");
  }

  public void speed2() {
    System.out.println("Max. Speed: 80 kmph");
  }
}

class Main {
  public static void main(String[] args) {
    Maruti800 mm = new Maruti800();
    mm.vehicletype();
    mm.brand();
    mm.speed1();
    mm.speed2();
  }
}
```

4 Abstract classes and methods

```
abstract class Shape {
  abstract double calculateArea();
  abstract double calculatePerimeter();
}

class Circle extends Shape {
  private double radius;

  public Circle(double radius) {
    this.radius = radius;
  }
```

```
    double calculateArea() {
        return Math.PI * radius * radius;
    }

    double calculatePerimeter() {
        return 2 * Math.PI * radius;
    }
}

class Triangle extends Shape {
    private double side1;
    private double side2;
    private double side3;

    public Triangle(double side1, double side2, double side3) {
        this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;
    }

    double calculateArea() {
        double s = (side1 + side2 + side3) / 2;
        return Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }

    double calculatePerimeter() {
        return side1 + side2 + side3;
    }
}
```

```
public class Main2 {
    public static void main(String[] args) {
        double r = 4.0;
        Circle circle = new Circle(r);

        double ts1 = 3.0, ts2 = 4.0, ts3 = 5.0;
        Triangle triangle = new Triangle(ts1, ts2, ts3);

        System.out.println("Radius of the Circle: " + r);
        System.out.println("Area of the Circle: " + circle.calculateArea());
        System.out.println("Perimeter of the Circle: " + circle.calculatePerimeter());
        System.out.println("\nSides of the Triangle are: " + ts1 + ", " + ts2 + ", " + ts3);
        System.out.println("Area of the Triangle: " + triangle.calculateArea());
        System.out.println("Perimeter of the Triangle: " + triangle.calculatePerimeter());
    }
}
```

5 list, set, map interfaces

```
import java.util.*;

public class CollectionInterfacesDemo {
    public static void main(String[] args) {
        List<String> list = new ArrayList<>();
        list.add("Apple");
        list.add("Banana");
        list.add("Cherry");
        list.add(1, "Mango");

        System.out.println("Om Singh 46");
        System.out.println("List elements: " + list);
        System.out.println("Element at index 2: " + list.get(2));
```

```
list.set(2, "Orange");
System.out.println("List after modification: " + list);

list.remove("Mango");
System.out.println("List after removing 'Mango': " + list);
System.out.println("Is 'Apple' in the list? " + list.contains("Apple"));
System.out.println("Size of the list: " + list.size());

Set<String> set = new HashSet<>();
set.add("Dog");
set.add("Cat");
set.add("Bird");
set.add("Cat"); // Duplicate, will not be added

System.out.println("\nSet elements: " + set);
set.remove("Bird");
System.out.println("Set after removing 'Bird': " + set);
System.out.println("Is 'Cat' in the set? " + set.contains("Cat"));
System.out.println("Size of the set: " + set.size());

Map<Integer, String> map = new HashMap<>();
map.put(1, "John");
map.put(2, "Jane");
map.put(3, "Doe");

System.out.println("\nMap elements: " + map);
System.out.println("Value for key 2: " + map.get(2));

map.put(2, "Mary");
System.out.println("Map after updating key 2: " + map);
```

```
        map.remove(3);
        System.out.println("Map after removing key 3: " + map);
        System.out.println("Is key 1 present in the map? " + map.containsKey(1));
        System.out.println("Is value 'Jane' present in the map? " + map.containsValue("Jane"));
        System.out.println("Size of the map: " + map.size());
    }
}
```

6 Swing components to design form

```
import javax.swing.*;

public class Form {
    public static void main(String[] args) {
        // Create the main frame
        JFrame f = new JFrame("Form Example");
        f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        f.setSize(400, 700); // Adjusted size for better visibility
        f.setResizable(false); // Optional: make the frame non-resizable
        f.setLayout(null); // Set layout to null for absolute positioning

        // Form Title
        JLabel lb = new JLabel("Form Example");
        lb.setBounds(150, 20, 100, 50);
        f.add(lb);

        // Name field
        JLabel lb1 = new JLabel("Name:");
        lb1.setBounds(100, 100, 100, 30);
        JTextField tf = new JTextField("Enter name.");
        tf.setBounds(150, 100, 150, 30);
        f.add(lb1);
```

```
f.add(tf);

// Password field
JLabel lb2 = new JLabel("Password: ");
lb2.setBounds(100, 150, 80, 30);
JPasswordField pf = new JPasswordField();
pf.setBounds(200, 150, 150, 30);
f.add(lb2);
f.add(pf);

// Fruits selection
JLabel lb3 = new JLabel("Fruits:");
lb3.setBounds(100, 200, 100, 30);
JCheckBox cb1 = new JCheckBox("Mango");
cb1.setBounds(100, 230, 100, 30);
JCheckBox cb2 = new JCheckBox("Papaya");
cb2.setBounds(100, 260, 100, 30);
JCheckBox cb3 = new JCheckBox("Orange");
cb3.setBounds(100, 290, 100, 30);
f.add(lb3);
f.add(cb1);
f.add(cb2);
f.add(cb3);

// Gender selection
JLabel lb4 = new JLabel("Gender :");
lb4.setBounds(100, 320, 100, 30);
JRadioButton r1 = new JRadioButton("Male");
JRadioButton r2 = new JRadioButton("Female");
r1.setBounds(100, 350, 100, 30);
r2.setBounds(100, 380, 100, 30);
```

```
        ButtonGroup bg = new ButtonGroup();
        bg.add(r1);
        bg.add(r2);
        f.add(lb4);
        f.add(r1);
        f.add(r2);

        // Submit button
        JButton bt = new JButton("Submit");
        bt.setBounds(150, 430, 100, 50);
        f.add(bt);

        // Set frame visibility
        f.setVisible(true);
    }
}
```

7 Swing for Simple Menu

```
import javax.swing.*;

public class SimpleMenuExample {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Om Singh 46");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 300);

        JMenuBar menuBar = new JMenuBar();

        JMenu menu1 = new JMenu("Menu 1");
        JMenuItem item1_1 = new JMenuItem("Item 1-1");
        JMenuItem item1_2 = new JMenuItem("Item 1-2");
```

```
        menu1.add(item1_1);
        menu1.add(item1_2);

        JMenu menu2 = new JMenu("Menu 2");
        JMenuItem item2_1 = new JMenuItem("Item 2-1");
        JMenuItem item2_2 = new JMenuItem("Item 2-2");
        menu2.add(item2_1);
        menu2.add(item2_2);

        menuBar.add(menu1);
        menuBar.add(menu2);

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

8 Servlet – Html

HTML

```
<html>
  <head>
    <title>Login Form</title>
  </head>
  <body>
    <form action="LoginServlet" method="get">
      Enter User ID: <input type="text" name="txtId"><br>
      Enter Password: <input type="password" name="txtPass"><br>
      <input type="reset">
      <input type="submit" value="Click to Login">
    </form>
  </body>
```

```
</html>
```

Servlet

```
package log;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class LoginServlet extends HttpServlet {

  // Handles the GET request from the HTML form
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {

    response.setContentType("text/html;charset=UTF-8");

    // Using try-with-resources to automatically close PrintWriter
    try (PrintWriter out = response.getWriter()) {
      out.println("<html><head><title>Servlet LoginServlet</title></head>");
      String uname = request.getParameter("txtId");
      String upass = request.getParameter("txtPass");

      // Simple authentication check
      if ("admin".equals(uname) && "12345".equals(upass)) {
        out.println("<body bgcolor='blue'>");
        out.println("<h1>Welcome !!! " + uname + "</h1>");
```

```
        } else {
            out.println("<body bgcolor='red'>");
            out.println("<h1>Login Failed !!!</h1>");
        }
        out.println("</body></html>");
    }
}

// Handles POST requests, though unused in this form setup
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    processRequest(request, response);
}

// Process request method is not explicitly used in this setup but can handle shared code
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        out.println("<!DOCTYPE html>");
        out.println("<html><head><title>Servlet LoginServlet</title></head><body>");
        out.println("<h1>Servlet LoginServlet at " + request.getContextPath() + "</h1>");
        out.println("</body></html>");
    }
}

@Override
public String getServletInfo() {
    return "Simple login authentication servlet";
}
```

```
}
```