



**Bangladesh Open University**  
**School of Science & Technology**  
**Bachelor of Science in Computer Science and Engineering**

**Assignment**

Course Title: Object Oriented Programming

Course Code: CSE2137 & CSE 21P8

***Submitted By***

Safkat Jamil (Imon)

Student Id No.: 20-0-52-801-061

Session: 2020-2021

2<sup>nd</sup> Year 1<sup>st</sup> Semester 2022

Dhaka Regional Center

***Submitted To***

Samrat Kumar Dey

Lecturer (Computer Science)

School of Science and Technology

Bangladesh Open University

***Date of Submission***

30 December 2023

**Q.1:** Write a java program that works as a simple calculator. Use a Grid Layout to arrange Buttons for digits and for the +, -, \*, and % operations. Add a text field to display the result.

**Ans.:**

**Code:**

```
package simple.calculator;

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

public class SimpleCalculator {
    private JFrame frame;
    private JTextField textField;
    private JButton[] numberButtons = new JButton[10];
    private JButton[] functionButtons = new JButton[8];
    private JButton addButton, subButton, mulButton, divButton, eqButton, clrButton, dotButton,
    perButton, backspaceButton, signButton;
    private JPanel panel;

    private double num1 = 0, num2 = 0, result = 0;
    private char operator;

    public SimpleCalculator() {
        frame = new JFrame("Calculator");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(400, 550);
        frame.setLayout(null);

        textField = new JTextField();
        textField.setBounds(10, 10, 360, 80);
        textField.setFont(new Font("Times New Roman", Font.PLAIN, 30));
        textField.setEditable(false);
        textField.setHorizontalAlignment(JTextField.RIGHT); // Set text alignment to right

        panel = new JPanel();
        panel.setBounds(10, 100, 360, 400);
        panel.setLayout(new GridLayout(5, 4, 5, 5));

        addButton = new JButton("+");
        subButton = new JButton("-");
        mulButton = new JButton("*");
        divButton = new JButton("/");
        eqButton = new JButton("=");
        clrButton = new JButton("CE");
        dotButton = new JButton(".");
        perButton = new JButton("%");
        backspaceButton = new JButton("C");
        signButton = new JButton("/-");

        functionButtons[0] = addButton;
        functionButtons[1] = subButton;
        functionButtons[2] = mulButton;
        functionButtons[3] = divButton;
        functionButtons[4] = eqButton;
        functionButtons[5] = clrButton;
```

```

functionButtons[6] = backspaceButton;
functionButtons[7] = signButton;

for (int i = 0; i < 8; i++) {
    functionButtons[i].setFont(new Font("Times New Roman", Font.PLAIN, 18));
    functionButtons[i].setFocusPainted(false);
    functionButtons[i].addActionListener(new OperatorAction());
}

for (int i = 0; i < 10; i++) {
    numberButtons[i] = new JButton(String.valueOf(i));
    numberButtons[i].setFont(new Font("Times New Roman", Font.PLAIN, 18));
    numberButtons[i].setFocusPainted(false);
    numberButtons[i].addActionListener(new NumberAction());
}

clrButton.setFont(new Font("Times New Roman", Font.PLAIN, 18));
clrButton.setFocusPainted(false);
clrButton.addActionListener(new ClearAction());

dotButton.setFont(new Font("Times New Roman", Font.PLAIN, 18));
dotButton.setFocusPainted(false);
dotButton.addActionListener(new DotAction());

perButton.setFont(new Font("Times New Roman", Font.PLAIN, 18));
perButton.setFocusPainted(false);
perButton.addActionListener(new OperatorAction());

backspaceButton.setFont(new Font("Times New Roman", Font.PLAIN, 18));
backspaceButton.setFocusPainted(false);
backspaceButton.addActionListener(new BackspaceAction());

signButton.setFont(new Font("Times New Roman", Font.PLAIN, 18));
signButton.setFocusPainted(false);
signButton.addActionListener(new SignAction());

panel.add(clrButton);
panel.add(backspaceButton);
panel.add(signButton);
panel.add(perButton);
panel.add(numberButtons[7]);
panel.add(numberButtons[8]);
panel.add(numberButtons[9]);
panel.add(divButton);
panel.add(numberButtons[4]);
panel.add(numberButtons[5]);
panel.add(numberButtons[6]);
panel.add(mulButton);
panel.add(numberButtons[1]);
panel.add(numberButtons[2]);
panel.add(numberButtons[3]);
panel.add(subButton);
panel.add(numberButtons[0]);
panel.add(dotButton);
panel.add(eqButton);
panel.add(addButton);

frame.add(textField);
frame.add(panel);
frame.setVisible(true);

```

```

    }

    private class SignAction implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            String currentText = textField.getText();
            if (!currentText.isEmpty()) {
                double currentValue = Double.parseDouble(currentText);
                currentValue = -currentValue;
                textField.setText(String.valueOf(currentValue));
            }
        }
    }

    private class NumberAction implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            JButton button = (JButton) e.getSource();
            textField.setText(textField.getText() + button.getText());
        }
    }

    private class OperatorAction implements ActionListener {
        public void actionPerformed(ActionEvent e) {
            JButton button = (JButton) e.getSource();
            String buttonText = button.getText();

            if (!buttonText.equals("=")) {
                if (!textField.getText().isEmpty()) {
                    num1 = Double.parseDouble(textField.getText());
                    operator = buttonText.charAt(0);
                    textField.setText("");
                }
            } else {
                if (!textField.getText().isEmpty()) {
                    num2 = Double.parseDouble(textField.getText());
                    switch (operator) {
                        case '+':
                            result = num1 + num2;
                            break;
                        case '-':
                            result = num1 - num2;
                            break;
                        case '*':
                            result = num1 * num2;
                            break;
                        case '/':
                            if (num2 != 0) {
                                result = num1 / num2;
                            } else {
                                JOptionPane.showMessageDialog(null,
                                    "Error: Division by zero is not allowed.");
                            }
                            return;
                        case '%':
                            result = num1 % num2;
                            break;
                    }
                }
                textField.setText(String.valueOf(result));
            }
        }
    }

```

```

    }
}

private class ClearAction implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        textField.setText("");
        num1 = 0;
        num2 = 0;
        result = 0;
    }
}

private class DotAction implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        if (!textField.getText().contains(".")) {
            textField.setText(textField.getText() + ".");
        }
    }
}

private class BackspaceAction implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        String currentText = textField.getText();
        if (!currentText.isEmpty()) {
            textField.setText(currentText.substring(0, currentText.length() - 1));
        }
    }
}

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

```

**Input:**

77 + 33 =

**Output:**



**Q.2:** Write a java program to find prime numbers between 1 to n.

**Ans.:**

**Code:**

```
package fine.the.prime.number.between.pkg1.to.n;

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

public class FineThePrimeNumberBetween1ToN {

    private JFrame frame;
    private JTextField inputField;
    private JTextArea outputArea;

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            try {
                FineThePrimeNumberBetween1ToN window = new FineThePrimeNumberBetween1ToN();
                window.frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        });
    }

    public FineThePrimeNumberBetween1ToN() {
        frame = new JFrame();
        frame.setTitle("Prime Numbers Finder");
        frame.setBounds(100, 100, 400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();
        frame.getContentPane().add(panel, "Center");
        panel.setLayout(null);

        JLabel lblEnterN = new JLabel("Enter n:");
        lblEnterN.setBounds(10, 15, 60, 20);
        panel.add(lblEnterN);

        inputField = new JTextField();
        inputField.setBounds(80, 12, 100, 25);
        panel.add(inputField);

        JButton btnFindPrimes = new JButton("Find Primes");
        btnFindPrimes.setBounds(190, 12, 120, 25);
        panel.add(btnFindPrimes);

        outputArea = new JTextArea();
        outputArea.setEditable(false);
        JScrollPane scrollPane = new JScrollPane(outputArea);
        scrollPane.setBounds(10, 50, 360, 200);
        panel.add(scrollPane);

        btnFindPrimes.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                findAndDisplayPrimes();
            }
        });
    }

    private void findAndDisplayPrimes() {
        // Implementation of the findAndDisplayPrimes method
    }
}
```

```

    }
    });
}

private void findAndDisplayPrimes() {
    outputArea.setText(""); // Clear previous output

    try {
        int n = Integer.parseInt(inputField.getText());
        outputArea.append("Prime numbers between 1 and " + n + " are:\n");
        for (int i = 2; i <= n; i++) {
            if (isPrime(i)) {
                outputArea.append(i + " ");
            }
        }
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(frame, "Please enter a valid number for 'n'.", "Error",
JOptionPane.ERROR_MESSAGE);
    }
}

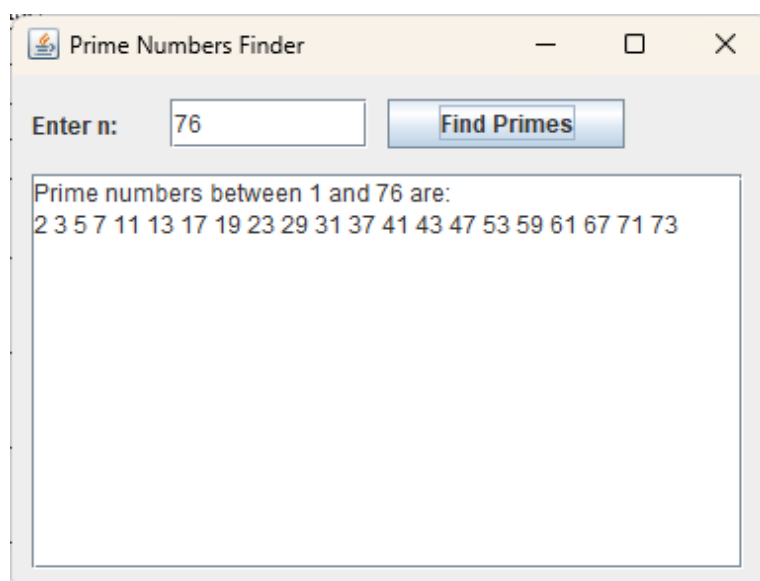
private boolean isPrime(int num) {
    if (num <= 1) {
        return false;
    }
    for (int i = 2; i <= Math.sqrt(num); i++) {
        if (num % i == 0) {
            return false;
        }
    }
    return true;
}
}

```

**Input:**

76

**Output:**



**Q.3:** Write a Java program that prints all real solutions to the quadratic equation  $ax^2+bx+c=0$ .

Read in a, b, c and use the quadratic formula.

**Ans.:**

**Code:**

```
package quadratic.equation;

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

public class QuadraticEquation extends JFrame {

    private JTextField aField, bField, cField;

    public QuadraticEquation() {
        super("Quadratic Equation Solver");

        // Set layout manager
        setLayout(new GridLayout(4, 2, 5, 5));

        // Create components
        JLabel aLabel = new JLabel("Enter a:");
        aField = new JTextField();
        JLabel bLabel = new JLabel("Enter b:");
        bField = new JTextField();
        JLabel cLabel = new JLabel("Enter c:");
        cField = new JTextField();

        JButton calculateButton = new JButton("Calculate");
        calculateButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                calculateAndDisplay();
            }
        });

        // Add components to the frame
        add(aLabel);
        add(aField);
        add(bLabel);
        add(bField);
        add(cLabel);
        add(cField);
        add(new JLabel()); // Empty label for spacing
        add(calculateButton);

        // Set frame properties
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(300, 200);
        setLocationRelativeTo(null); // Center the frame
        setVisible(true);
    }

    private void calculateAndDisplay() {
        try {
```



```

// Get values from text fields
double a = Double.parseDouble(aField.getText());
double b = Double.parseDouble(bField.getText());
double c = Double.parseDouble(cField.getText());

// Calculate the discriminant
double discriminant = b * b - 4 * a * c;

// Check if the discriminant is non-negative
if (discriminant >= 0) {
    // Calculate the real solutions using the quadratic formula
    double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
    double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);

    // Display result in a new window
    displayResult(root1, root2);
} else {
    // If the discriminant is negative, no real solutions exist
    JOptionPane.showMessageDialog(this, "No real solutions. The discriminant is negative.",
        "Result", JOptionPane.INFORMATION_MESSAGE);
}
} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(this, "Invalid input. Please enter numeric values for a, b, and c.",
        "Error", JOptionPane.ERROR_MESSAGE);
}
}

private void displayResult(double root1, double root2) {
    JFrame resultFrame = new JFrame("Result");
    resultFrame.setLayout(new GridLayout(3, 1));

    JLabel label1 = new JLabel("Real Solutions:");
    JLabel label2 = new JLabel("Root 1: " + root1);
    JLabel label3 = new JLabel("Root 2: " + root2);

    resultFrame.add(label1);
    resultFrame.add(label2);
    resultFrame.add(label3);

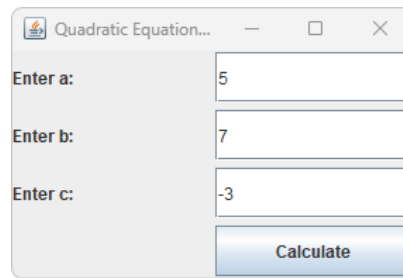
    resultFrame.setSize(300, 150);
    resultFrame.setLocationRelativeTo(this); // Center the result frame relative to the main frame
    resultFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    resultFrame.setVisible(true);
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new QuadraticEquation();
        }
    });
}
}

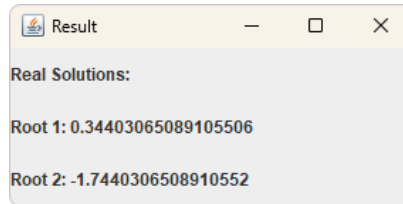
```

### Input:

Enter a: 5, b: 7, c: -3



**Output:**



**Q.4:** Create a base class Fruit which has name, taste, and size as its attributes. A method called eat() is created which describes the name of the fruit and its taste. Inherit the same in 2 other class Apple and Orange and override the eat() method to represent each fruit taste.

**Ans.:**

**Code:**

```
package fruit;

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

// Base class
class Fruit0 {
    protected String name;
    protected String taste;
    protected String size;

    // Constructor
    public Fruit0(String name, String taste, String size) {
        this.name = name;
        this.taste = taste;
        this.size = size;
    }

    // Method to describe eating the fruit
    public void eat() {
        JOptionPane.showMessageDialog(null, "Eating " + name + ". It tastes " + taste + ".");
    }
}

// Subclass Apple
class Apple extends Fruit0 {
    // Constructor
```

```

public Apple(String size) {
    super("Apple", "sweet", size);
}

// Override eat method to represent apple taste
@Override
public void eat() {
    JOptionPane.showMessageDialog(null, "Crunching on an apple. It tastes sweet and a bit tangy.");
}
}

// Subclass Orange
class Orange extends Fruit0 {
    // Constructor
    public Orange(String size) {
        super("Orange", "citrusy", size);
    }

    // Override eat method to represent orange taste
    @Override
    public void eat() {
        JOptionPane.showMessageDialog(null, "Peeling and enjoying an orange. It tastes citrusy and refreshing.");
    }
}

// Main GUI class
public class Fruit extends JFrame {

    private JComboBox<String> fruitComboBox;
    private JButton eatButton;

    public Fruit() {
        setTitle("Fruit Eating App");
        setSize(300, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout());

        // Create and populate the combo box
        String[] fruits = {"Apple", "Orange", "Unknown Fruit"};
        fruitComboBox = new JComboBox<>(fruits);
        add(fruitComboBox);

        // Create the Eat button
        eatButton = new JButton("Eat");
        add(eatButton);

        // Add action listener to the Eat button
        eatButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                eatSelectedFruit();
            }
        });
    }

    // Method to eat the selected fruit based on the combo box selection
    private void eatSelectedFruit() {
        String selectedFruit = (String) fruitComboBox.getSelectedItem();

        Fruit0 fruit;
    }
}

```

```

switch (selectedFruit) {
    case "Apple":
        fruit = new Apple("medium");
        break;
    case "Orange":
        fruit = new Orange("large");
        break;
    default:
        fruit = new Fruit0("Unknown Fruit", "unknown", "medium");
}

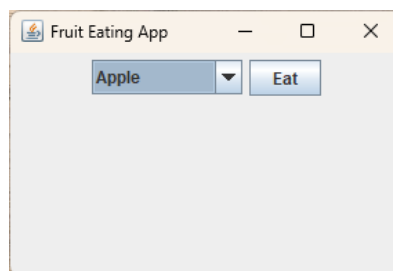
fruit.eat();
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new Fruit().setVisible(true);
        }
    });
}
}

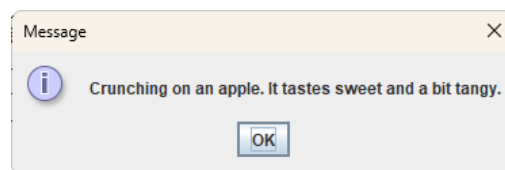
```

### Input-1:

Select "Apple" and click "Eat".

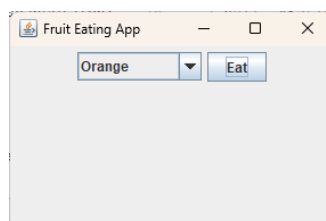


### Output-1:



### Input-2:

Select "Orange" and click "Eat".



## Output-2:



**Q.5:** Write a java program to illustrate the concept of class with method overloading.

**Ans.:**

**Code:**

```
package illustrate.overloading;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;

public class IllustrateOverloading extends JFrame {

    public IllustrateOverloading() {
        setTitle("Method Overloading");
        setSize(300, 150);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JButton addButton = new JButton("Add Two Integers");
        addButton.addActionListener(e -> {
            int num1 = getUserInput("Enter the first integer:");
            int num2 = getUserInput("Enter the second integer:");
            showResult(add(num1, num2));
        });

        JButton addThreeButton = new JButton("Add Three Integers");
        addThreeButton.addActionListener(e -> {
            int num1 = getUserInput("Enter the first integer:");
            int num2 = getUserInput("Enter the second integer:");
            int num3 = getUserInput("Enter the third integer:");
            showResult(add(num1, num2, num3));
        });

        JButton concatenateButton = new JButton("Concatenate Strings");
        concatenateButton.addActionListener(e -> {
            String str1 = JOptionPane.showInputDialog("Enter the first string:");
            String str2 = JOptionPane.showInputDialog("Enter the second string:");
            showResult(concatenate(str1, str2));
        });

        setLayout(new java.awt.FlowLayout());
        add(addButton);
        add(addThreeButton);
        add(concatenateButton);
    }

    // Method to add two integers
    public int add(int a, int b) {
        return a + b;
    }
}
```

```

// Method to add three integers
public int add(int a, int b, int c) {
    return a + b + c;
}

// Method to concatenate two strings
public String concatenate(String str1, String str2) {
    return str1 + str2;
}

private int getUserInput(String message) {
    String input = JOptionPane.showInputDialog(message);
    return Integer.parseInt(input);
}

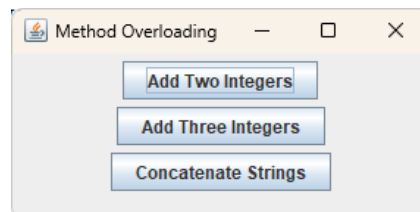
private void showResult(Object result) {
    JOptionPane.showMessageDialog(this, "Result: " + result, "Method Overloading Example",
JOptionPane.INFORMATION_MESSAGE);
}

public static void main(String[] args) {
    javax.swing.SwingUtilities.invokeLater(() -> {
        IllustrateOverloading gui = new IllustrateOverloading();
        gui.setVisible(true);
    });
}
}

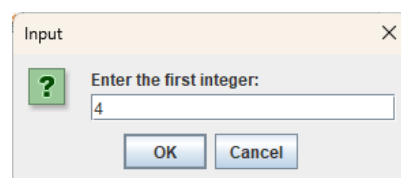
```

### Input-1:

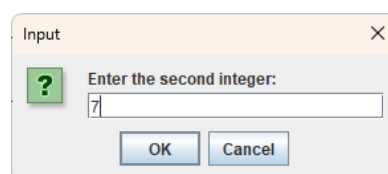
Select "Add Two Integers".



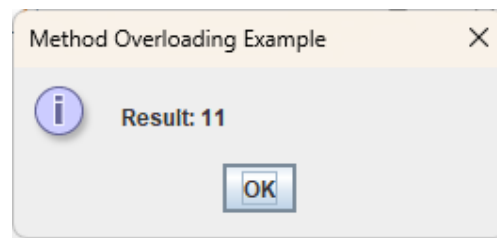
Enter the first integer "4".



Enter the second integer "7".

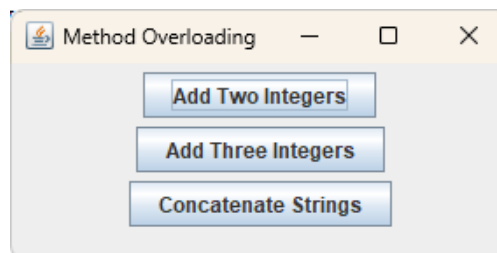


### Output-1:

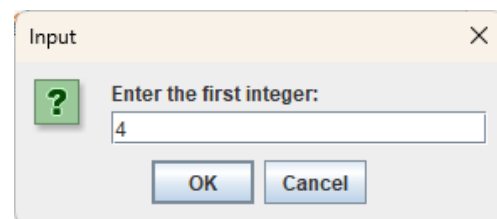


### Input-2:

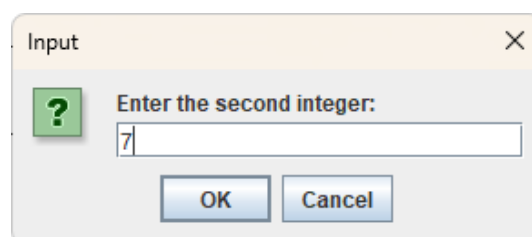
Select "Add Three Integers".



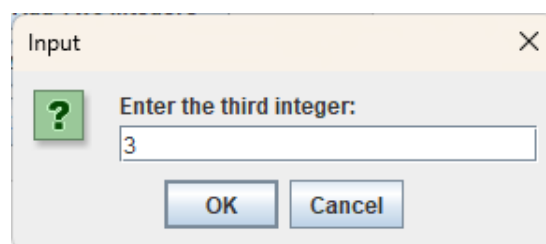
Enter the first integer "4".



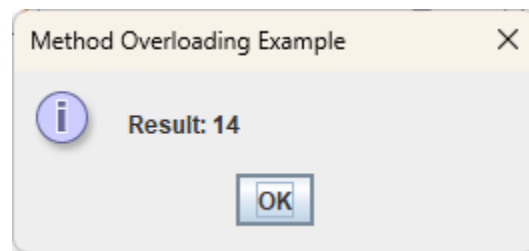
Enter the second integer "7".



Enter the third integer "3".

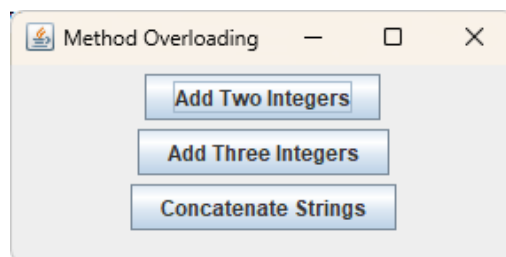


**Output-2:**

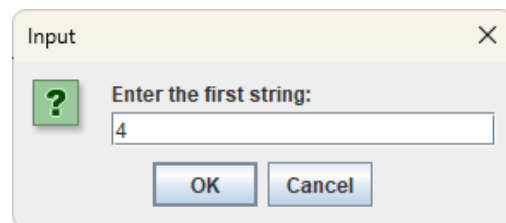


**Input-3:**

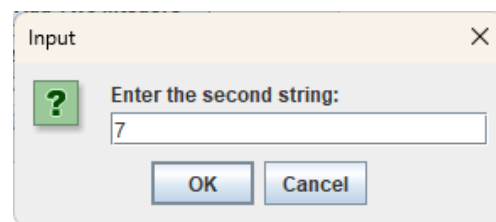
Select "Concatenate Strings".



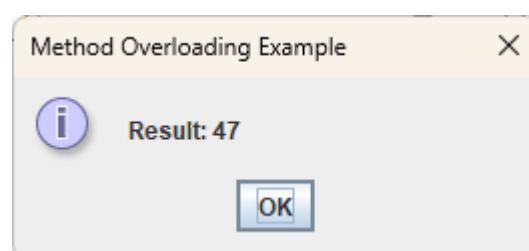
Enter the first string "4".



Enter the second string "7".



**Output-3:**





**Q.6:** Write a Java Program to create an abstract class named Shape that contains two integers and an empty method named print Area(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method print Area () that prints the area of the given shape.

**Ans.:**

**Code:**

```
package shape;

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

abstract class Shape {
    protected int dimension1;
    protected int dimension2;

    public Shape(int dimension1, int dimension2) {
        this.dimension1 = dimension1;
        this.dimension2 = dimension2;
    }

    public abstract void printArea();
}

class Rectangle extends Shape {
    public Rectangle(int length, int width) {
        super(length, width);
    }

    @Override
    public void printArea() {
        int area = dimension1 * dimension2;
        JOptionPane.showMessageDialog(null, "Area of Rectangle: " + area);
    }
}

class Triangle extends Shape {
    public Triangle(int base, int height) {
        super(base, height);
    }

    @Override
    public void printArea() {
        int area = (int) (0.5 * dimension1 * dimension2);
        JOptionPane.showMessageDialog(null, "Area of Triangle: " + area);
    }
}

class Circle extends Shape {
    public Circle(int radius) {
        super(radius, 0); // Assuming dimension1 is the radius and dimension2 is not used
    }

    @Override
```

```

    public void printArea() {
        int area = (int) (Math.PI * dimension1 * dimension1);
        JOptionPane.showMessageDialog(null, "Area of Circle: " + area);
    }
}

public class ThreeTypeShape {

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

            String[] shapeOptions = {"Select", "Rectangle", "Triangle", "Circle"};
            JComboBox<String> shapeComboBox = new JComboBox<>(shapeOptions);
            JTextField dimension1Field = new JTextField(10);
            JTextField dimension2Field = new JTextField(10);
            JLabel label1 = new JLabel("Dimension 1:");
            JLabel label2 = new JLabel("Dimension 2:");
            JButton calculateButton = new JButton("Calculate");
            calculateButton.setEnabled(false); // Disable initially

            JPanel panel = new JPanel(new GridLayout(5, 1));
            panel.add(new JLabel("Select Shape:"));
            panel.add(shapeComboBox);
            panel.add(label1);
            panel.add(dimension1Field);
            panel.add(label2);
            panel.add(dimension2Field);

            JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER));
            buttonPanel.add(calculateButton);

            frame.add(panel, BorderLayout.CENTER);
            frame.add(buttonPanel, BorderLayout.SOUTH);

            calculateButton.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                    String selectedShape = (String) shapeComboBox.getSelectedItem();
                    int dimension1 = Integer.parseInt(dimension1Field.getText());
                    int dimension2 = Integer.parseInt(dimension2Field.getText());

                    if (!selectedShape.equals("Select")) {
                        Shape shape = createShape(selectedShape, dimension1, dimension2);
                        if (shape != null) {
                            shape.printArea();
                        }
                    }
                }
            });

            shapeComboBox.addActionListener(new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                    String selectedShape = (String) shapeComboBox.getSelectedItem();
                    label1.setText(getLabelName(selectedShape, 1));
                    label2.setText(getLabelName(selectedShape, 2));
                    dimension2Field.setVisible(!selectedShape.equals("Circle"));
                    calculateButton.setEnabled(!selectedShape.equals("Select"));
                }
            });
        });
    }
}

```

```

    }
    });

    frame.setSize(220, 180);
    frame.setLocationRelativeTo(null);
    frame.setVisible(true);
    });
}

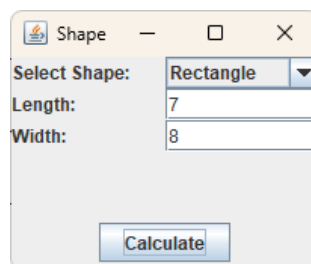
private static Shape createShape(String shapeType, int dimension1, int dimension2) {
    switch (shapeType) {
        case "Rectangle":
            return new Rectangle(dimension1, dimension2);
        case "Triangle":
            return new Triangle(dimension1, dimension2);
        case "Circle":
            return new Circle(dimension1);
        default:
            return null;
    }
}

private static String getLabelName(String shapeType, int dimensionNumber) {
    switch (shapeType) {
        case "Rectangle":
            return (dimensionNumber == 1) ? "Length:" : "Width:";
        case "Triangle":
            return (dimensionNumber == 1) ? "Base:" : "Height:";
        case "Circle":
            return (dimensionNumber == 1) ? "Radius:" : "";
        default:
            return "";
    }
}
}
}

```

### Input-1:

Select Shape "Rectangle". After that, input Length: 7 and Width: 8.

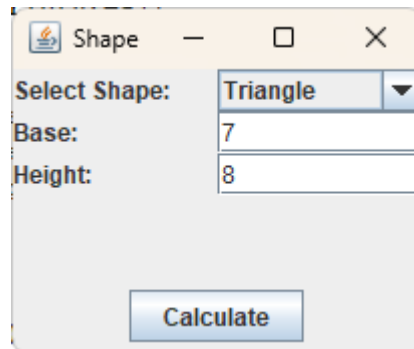


### Output-1:

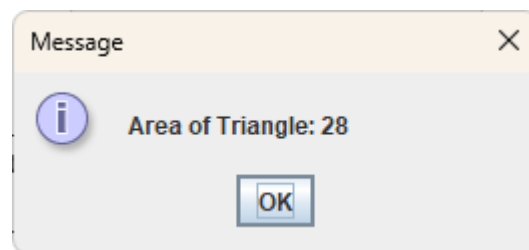


**Input-2:**

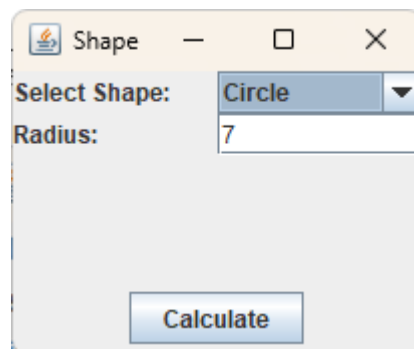
Select Shape "Triangle". After that, input Base: 7 and Height: 8.



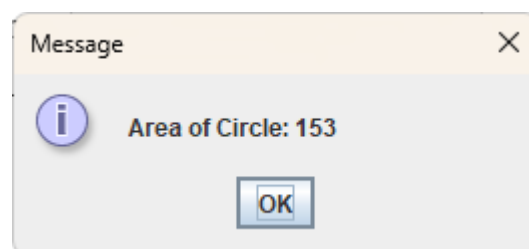
A screenshot of a Java Swing window titled "Shape". It contains a dropdown menu labeled "Select Shape:" with "Triangle" selected. Below it are two text input fields: "Base:" with the value "7" and "Height:" with the value "8". At the bottom is a "Calculate" button.

**Output-2:****Input-3:**

Select Shape "Circle". After that, input Radius: 7.



A screenshot of a Java Swing window titled "Shape". It contains a dropdown menu labeled "Select Shape:" with "Circle" selected. Below it is a text input field labeled "Radius:" with the value "7". At the bottom is a "Calculate" button.

**Output-3:**

**Q.7:** Develop a java application with Employee class with Emp\_name, Emp\_id, Address, Mail\_id, Mobile\_no as members. Inherit the classes, Programmer, Assistant Professor, Associate Professor and Professor from employee class. Add Basic Pay (BP) as the member of all the inherited classes with 97% of BP as Dearness allowance (DA), 10% of BP as House Rent Allowance (HRA), 12% of BP as Provident Fund (PF), 0.1% of BP for staff club fund. Generate pay slips for the employees with their gross and net salary.

**Ans.:**

**Code:**

```
package employee;

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

class Employee {
    String emp_name, emp_id, address, mail_id, mobile_no;
}

class Programmer extends Employee {
    int basicPay;

    Programmer(int basicPay) {
        this.basicPay = basicPay;
    }
}

class AssistantProfessor extends Employee {
    int basicPay;

    AssistantProfessor(int basicPay) {
        this.basicPay = basicPay;
    }
}

class AssociateProfessor extends Employee {
    int basicPay;

    AssociateProfessor(int basicPay) {
        this.basicPay = basicPay;
    }
}

class Professor extends Employee {
    int basicPay;

    Professor(int basicPay) {
        this.basicPay = basicPay;
    }
}

class PayrollResultWindow extends JFrame {
```

```

private JTextArea resultArea;

public PayrollResultWindow(String result, int totalEntries) {
    setTitle("Payroll Result");
    setSize(400, 400);

    // Center the frame on the screen
    Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
    int screenWidth = screenSize.width;
    int screenHeight = screenSize.height;
    int frameWidth = getWidth();
    int frameHeight = getHeight();
    setLocation((screenWidth - frameWidth) / 2, (screenHeight - frameHeight) / 2);

    setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

    resultArea = new JTextArea(result);
    resultArea.setEditable(false);

    JScrollPane scrollPane = new JScrollPane(resultArea);
    add(scrollPane, BorderLayout.CENTER);

    // Add total entries section
    JLabel totalLabel = new JLabel("Total Entries: " + totalEntries);
    totalLabel.setHorizontalAlignment(JLabel.CENTER);
    add(totalLabel, BorderLayout.SOUTH);
}

}

public class EmployeeSalary extends JFrame {
    private JTextField empNameField, empIdField, addressField, mailIdField, mobileNoField, basicPayField;
    private JComboBox<String> employeeTypeComboBox;
    private JTextArea resultArea;
    private ArrayList<String> history;

    public EmployeeSalary() {
        setTitle("Payroll System");
        setSize(380, 255);

        // Center the frame on the screen
        Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
        int screenWidth = screenSize.width;
        int screenHeight = screenSize.height;
        int frameWidth = getWidth();
        int frameHeight = getHeight();
        setLocation((screenWidth - frameWidth) / 2, (screenHeight - frameHeight) / 2);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BoxLayout(getContentPane(), BoxLayout.Y_AXIS));

        history = new ArrayList<>();

        employeeTypeComboBox = new JComboBox<>(new String[]{"Programmer", "Assistant Professor",
"Associate Professor", "Professor"});
        add(createComboBoxPanel("Employee Type:", employeeTypeComboBox));

        empNameField = createTextField();
        empIdField = createTextField();
        addressField = createTextField();
        mailIdField = createTextField();

```

```

mobileNoField = createTextField();
basicPayField = createTextField();

JButton calculateButton = new JButton("Calculate Salary");
calculateButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        calculateSalary();
    }
});

JButton showHistoryButton = new JButton("Show History");
showHistoryButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        showHistory();
    }
});

resultArea = new JTextArea();
resultArea.setEditable(false);

JPanel inputPanel = new JPanel();
inputPanel.setLayout(new GridLayout(6, 2, 5, 5));
inputPanel.add(new JLabel("Employee Name:"));
inputPanel.add(empNameField);
inputPanel.add(new JLabel("Employee ID:"));
inputPanel.add(empIdField);
inputPanel.add(new JLabel("Address:"));
inputPanel.add(addressField);
inputPanel.add(new JLabel("Mail ID:"));
inputPanel.add(mailIdField);
inputPanel.add(new JLabel("Mobile No:"));
inputPanel.add(mobileNoField);
inputPanel.add(new JLabel("Basic Pay:"));
inputPanel.add(basicPayField);

JPanel buttonPanel = new JPanel();
buttonPanel.setLayout(new FlowLayout(FlowLayout.CENTER));
buttonPanel.add(calculateButton);
buttonPanel.add(showHistoryButton);

add(inputPanel);
add(buttonPanel);
add(resultArea);
}

private JTextField createTextField() {
    JTextField jTextField = new JTextField(20);
    jTextField.setAlignmentX(Component.LEFT_ALIGNMENT);
    return jTextField;
}

private JPanel createComboBoxPanel(String label, JComboBox<String> comboBox) {
    JPanel panel = new JPanel();
    panel.setLayout(new FlowLayout(FlowLayout.LEFT));
    panel.add(new JLabel(label));
    panel.add(comboBox);
    panel.setAlignmentX(Component.LEFT_ALIGNMENT);
    return panel;
}

```

```

}

private void calculateSalary() {
    try {
        String empType = (String) employeeTypeComboBox.getSelectedItem();
        String empName = empNameField.getText();
        String empId = empIdField.getText();
        String address = addressField.getText();
        String mailId = mailIdField.getText();
        String mobileNo = mobileNoField.getText();
        int basicPay = Integer.parseInt(basicPayField.getText());

        history.add("Employee Type: " + empType + ", Employee Name: " + empName + ", Employee ID: " + empId +
            ", Address: " + address + ", Mail ID: " + mailId + ", Mobile No: " + mobileNo + ", Basic Pay: " + basicPay);

        Employee employee = null;

        switch (empType) {
            case "Programmer":
                employee = new Programmer(basicPay);
                break;
            case "Assistant Professor":
                employee = new AssistantProfessor(basicPay);
                break;
            case "Associate Professor":
                employee = new AssociateProfessor(basicPay);
                break;
            case "Professor":
                employee = new Professor(basicPay);
                break;
        }

        // Display the pay slip
        double da = (0.97 * basicPay);
        double hra = (0.10 * basicPay);
        double pf = (0.12 * basicPay);
        double staffClubFund = (0.001 * basicPay);

        double grossSalary = basicPay + da + hra;
        double netSalary = grossSalary - pf - staffClubFund;

        String result = "Pay Slip\n" +
            "Employee Type: " + empType + "\n" +
            "Employee Name: " + empName + "\n" +
            "Employee ID: " + empId + "\n" +
            "Address: " + address + "\n" +
            "Mail ID: " + mailId + "\n" +
            "Mobile No: " + mobileNo + "\n" +
            "Basic Pay: ₹" + basicPay + "\n" +
            "Dearness Allowance (DA): ₹" + da + "\n" +
            "House Rent Allowance (HRA): ₹" + hra + "\n" +
            "Provident Fund (PF): ₹" + pf + "\n" +
            "Staff Club Fund: ₹" + staffClubFund + "\n" +
            "Gross Salary: ₹" + grossSalary + "\n" +
            "Net Salary: ₹" + netSalary + "\n";

        resultArea.setText(result);
    }
}

```



```

        // Show result in a new window
        int totalEntries = history.size();
        new PayrollResultWindow(result, totalEntries).setVisible(true);

    } catch (NumberFormatException e) {
        resultArea.setText("Error: Please enter valid numeric values for Basic Pay.");
    }
}

private void showHistory() {
    StringBuilder historyText = new StringBuilder("Input History:\n");

    for (String entry : history) {
        historyText.append(entry).append("\n");
    }

    JTextArea historyArea = new JTextArea(historyText.toString());
    historyArea.setEditable(false);

    JScrollPane scrollPane = new JScrollPane(historyArea);

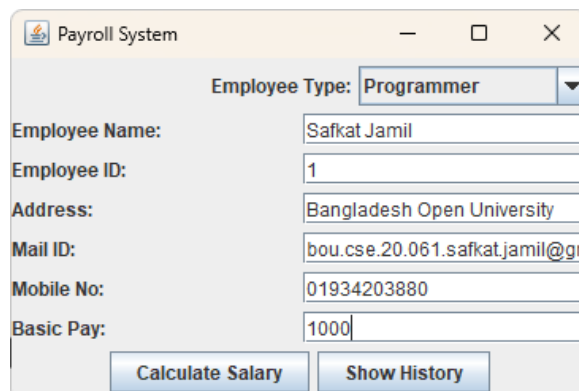
    JFrame historyFrame = new JFrame("Input History");
    historyFrame.setSize(400, 400);
    historyFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    historyFrame.add(scrollPane);
    historyFrame.setVisible(true);
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new EmployeeSalary().setVisible(true);
        }
    });
}
}

```

### Input-1:

Select Employee Type "Programmer". Now, input Employee Name: Safkat Jamil, Employee ID: 1, Address: Bangladesh Open University, Mail ID: bou.cse.20.061.safkat.jamil@gmail.com, Mobile No: 01934203880, Basic Pay: 1000

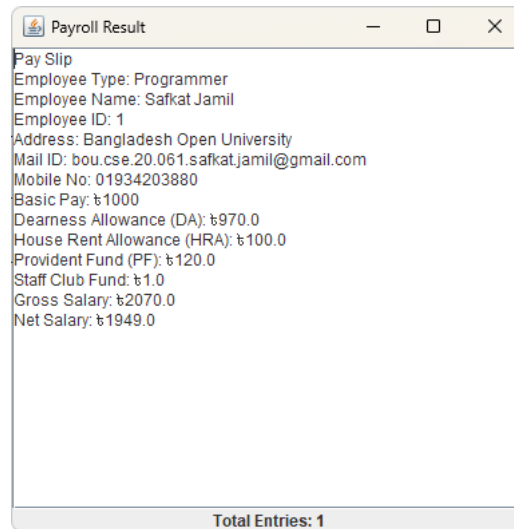


The screenshot shows a Java Swing window titled "Payroll System". It contains a form with the following fields and values:

- Employee Type:** A dropdown menu with "Programmer" selected.
- Employee Name:** Text field containing "Safkat Jamil".
- Employee ID:** Text field containing "1".
- Address:** Text field containing "Bangladesh Open University".
- Mail ID:** Text field containing "bou.cse.20.061.safkat.jamil@gmail.com".
- Mobile No:** Text field containing "01934203880".
- Basic Pay:** Text field containing "1000".

At the bottom of the form, there are two buttons: "Calculate Salary" and "Show History".

### Output-1:



Payroll Result

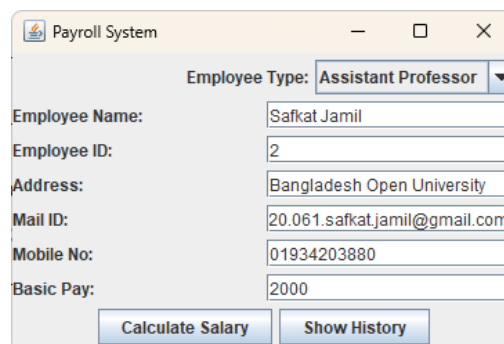
Pay Slip

Employee Type: Programmer  
Employee Name: Safkat Jamil  
Employee ID: 1  
Address: Bangladesh Open University  
Mail ID: bou.cse.20.061.safkat.jamil@gmail.com  
Mobile No: 01934203880  
Basic Pay: ₳1000  
Dearness Allowance (DA): ₳970.0  
House Rent Allowance (HRA): ₳100.0  
Provident Fund (PF): ₳120.0  
Staff Club Fund: ₳1.0  
Gross Salary: ₳2070.0  
Net Salary: ₳1949.0

Total Entries: 1

### Input-2:

Select Employee Type "Assistant Professor". Now, input Employee Name: Safkat Jamil, Employee ID: 2, Address: Bangladesh Open University, Mail ID: bou.cse.20.061.safkat.jamil@gmail.com, Mobile No: 01934203880, Basic Pay: 2000



Payroll System

Employee Type: Assistant Professor

Employee Name: Safkat Jamil

Employee ID: 2

Address: Bangladesh Open University

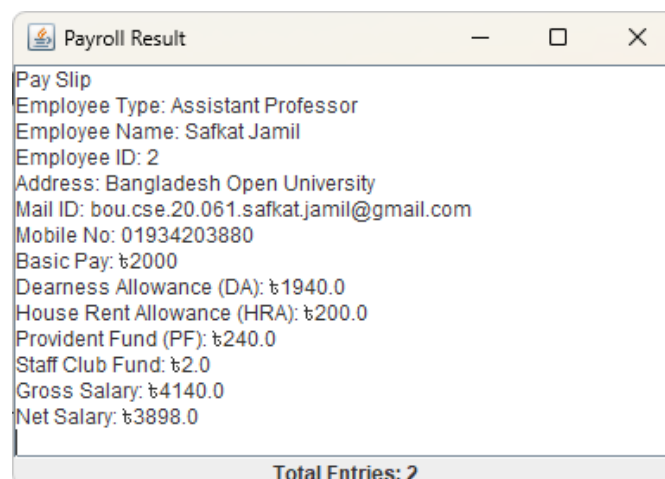
Mail ID: 20.061.safkat.jamil@gmail.com

Mobile No: 01934203880

Basic Pay: 2000

Calculate Salary Show History

### Output-2:



Payroll Result

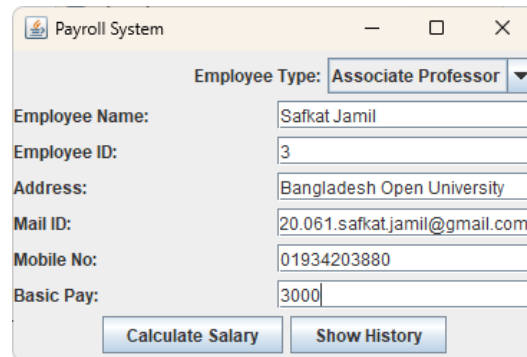
Pay Slip

Employee Type: Assistant Professor  
Employee Name: Safkat Jamil  
Employee ID: 2  
Address: Bangladesh Open University  
Mail ID: bou.cse.20.061.safkat.jamil@gmail.com  
Mobile No: 01934203880  
Basic Pay: ₳2000  
Dearness Allowance (DA): ₳1940.0  
House Rent Allowance (HRA): ₳200.0  
Provident Fund (PF): ₳240.0  
Staff Club Fund: ₳2.0  
Gross Salary: ₳4140.0  
Net Salary: ₳3898.0

Total Entries: 2

### Input-3:

Select Employee Type "Associate Professor". Now, input Employee Name: Safkat Jamil, Employee ID: 3, Address: Bangladesh Open University, Mail ID: bou.cse.20.061.safkat.jamil@gmail.com, Mobile No: 01934203880, Basic Pay: 3000

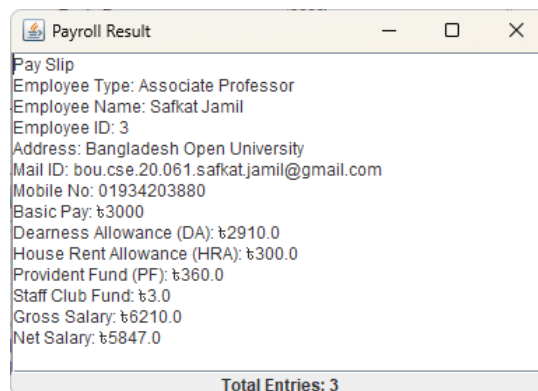


The screenshot shows a window titled "Payroll System". It contains a form with the following fields and values:

Field	Value
Employee Type	Associate Professor
Employee Name	Safkat Jamil
Employee ID	3
Address	Bangladesh Open University
Mail ID	20.061.safkat.jamil@gmail.com
Mobile No	01934203880
Basic Pay	3000

At the bottom of the form are two buttons: "Calculate Salary" and "Show History".

### Output-3:



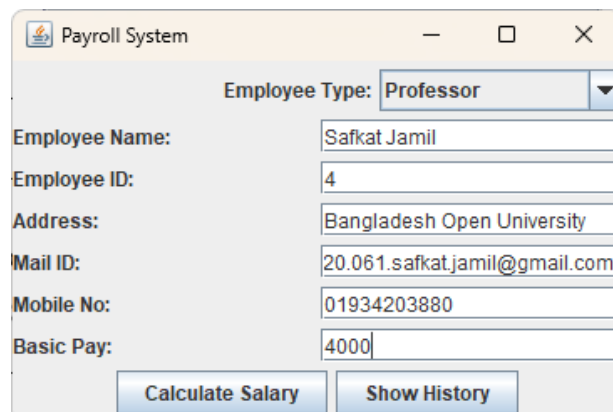
The screenshot shows a window titled "Payroll Result". It displays a "Pay Slip" with the following information:

Pay Slip  
Employee Type: Associate Professor  
Employee Name: Safkat Jamil  
Employee ID: 3  
Address: Bangladesh Open University  
Mail ID: bou.cse.20.061.safkat.jamil@gmail.com  
Mobile No: 01934203880  
Basic Pay: ₳3000  
Dearness Allowance (DA): ₳2910.0  
House Rent Allowance (HRA): ₳300.0  
Provident Fund (PF): ₳360.0  
Staff Club Fund: ₳3.0  
Gross Salary: ₳6210.0  
Net Salary: ₳5847.0

At the bottom of the window, it says "Total Entries: 3".

### Input-4:

Select Employee Type " Professor". Now, input Employee Name: Safkat Jamil, Employee ID: 4, Address: Bangladesh Open University, Mail ID: bou.cse.20.061.safkat.jamil@gmail.com, Mobile No: 01934203880, Basic Pay: 4000

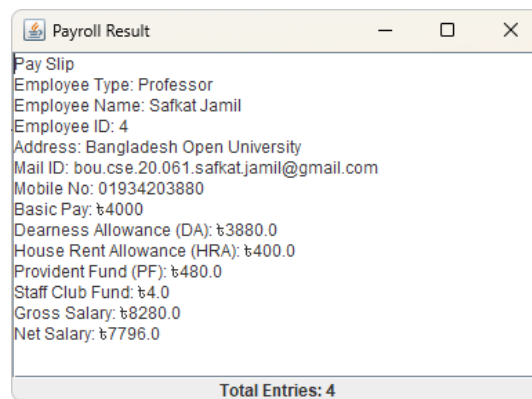


The screenshot shows a window titled "Payroll System". It contains a form with the following fields and values:

Field	Value
Employee Type	Professor
Employee Name	Safkat Jamil
Employee ID	4
Address	Bangladesh Open University
Mail ID	20.061.safkat.jamil@gmail.com
Mobile No	01934203880
Basic Pay	4000

At the bottom of the form are two buttons: "Calculate Salary" and "Show History".

#### Output-4:



**Q.8:** Write a program that creates a user interface to perform integer divisions. The user enters two numbers in the text fields, Num1 and Num2. The division of Num1 and Num2 is displayed in the Result field when the Divide button is clicked. If Num1 or Num2 were not an integer, the program would throw a `NumberFormatException`. If Num2 were Zero, the program would throw an `Arithmetic Exception` Display the exception in a message dialog box.

**Ans.:**

**Code:**

```
package integer.division;

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

public class IntegerDivision extends JFrame {

    private JTextField num1Field, num2Field;
    private JFrame resultFrame;
    private JTextField resultField;

    public IntegerDivision() {
        // Set up the main frame
        setTitle("Integre Division");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(270, 140);

        // Center the main frame on the screen
        setLocationRelativeTo(null);

        // Create components for the main frame
        JLabel num1Label = new JLabel("Num1:");
        num1Field = new JTextField(10);

        JLabel num2Label = new JLabel("Num2:");
        num2Field = new JTextField(10);
```

```

JButton divideButton = new JButton("Divide");

// Set layout for the main frame
setLayout(new GroupLayout(getContentPane()));

GroupLayout layout = (GroupLayout) getContentPane().getLayout();
layout.setAutoCreateGaps(true);
layout.setAutoCreateContainerGaps(true);

layout.setHorizontalGroup(layout.createSequentialGroup()
    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.LEADING)
        .addComponent(num1Label)
        .addComponent(num2Label))
    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
        .addComponent(num1Field)
        .addComponent(num2Field)
        .addComponent(divideButton))
);

layout.setVerticalGroup(layout.createSequentialGroup()
    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
        .addComponent(num1Label)
        .addComponent(num1Field))
    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
        .addComponent(num2Label)
        .addComponent(num2Field))
    .addComponent(divideButton)
);

// Add action listener to the divide button
divideButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        try {
            int num1 = Integer.parseInt(num1Field.getText());
            int num2 = Integer.parseInt(num2Field.getText());

            if (num2 == 0) {
                throw new ArithmeticException("Cannot divide by zero");
            }

            int result = num1 / num2;

            // Display the result in a new window
            displayResult(result);
        } catch (NumberFormatException ex) {
            showErrorMessage("Please enter valid integers for Num1 and Num2");
        } catch (ArithmeticException ex) {
            showErrorMessage("Cannot divide by zero");
        }
    }
});
}

private void displayResult(int result) {
    // Set up the result frame
    resultFrame = new JFrame("Result");
    resultFrame.setSize(200, 100);
    resultFrame.setLayout(new FlowLayout());
}

```

```

resultFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

// Center the result frame on the screen
resultFrame.setLocationRelativeTo(null);

// Create components for the result frame
resultField = new JTextField(String.valueOf(result), 10);
resultField.setEditable(false);

resultFrame.add(new JLabel("Result: "));
resultFrame.add(resultField);

// Make the result frame visible
resultFrame.setVisible(true);
}

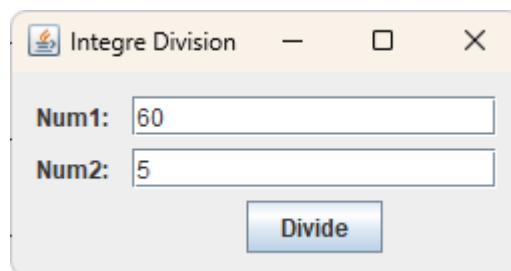
private void showErrorMessage(String message) {
    JOptionPane.showMessageDialog(this, message, "Error", JOptionPane.ERROR_MESSAGE);
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new IntegerDivision().setVisible(true);
        }
    });
}
}

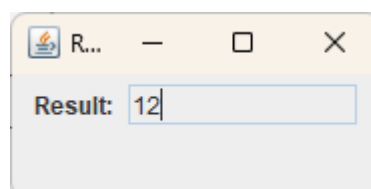
```

### Input-1:

Input Num1: 60 and Num2: 5.

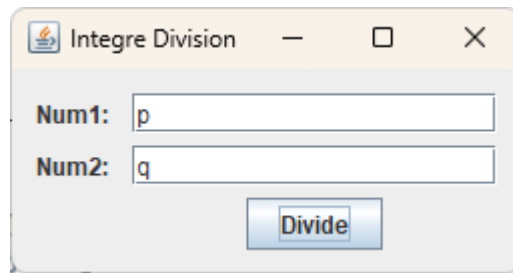


### Output-1:

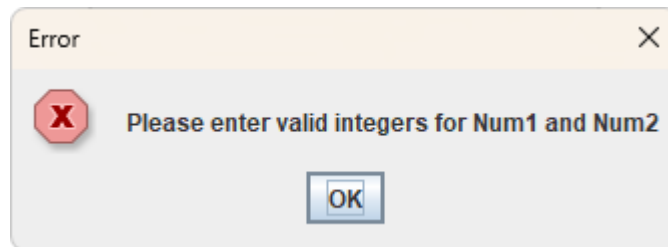


### Input-2:

Input Num1: p and Num2: q.

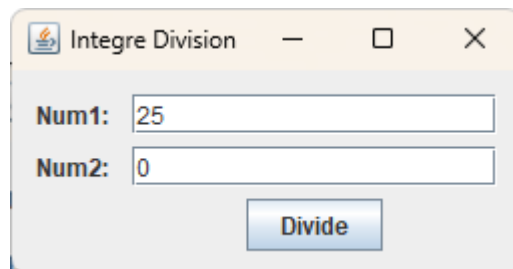


**Output-2:**

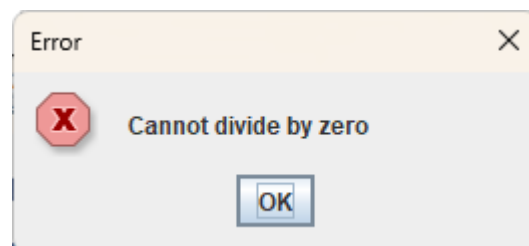


**Input-3:**

Input Num1: 25 and Num2: 0.



**Output-3:**



**Q.9:** Write a Java program that handles all mouse events and shows the event name at the center of the window when a mouse event is fired. (Use adapter classes).

**Ans.:**

**Code:**

```
package mouse.event;

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

import javax.swing.*;

public class MouseEvents extends JFrame {

    private JLabel eventLabel;

    public MouseEvents() {
        super("Mouse Event");

        eventLabel = new JLabel("Event Name");
        eventLabel.setHorizontalAlignment(JLabel.CENTER);
        add(eventLabel);

        addMouseListener(new MyMouseAdapter());

        setSize(400, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setVisible(true);
    }

    private class MyMouseAdapter extends MouseAdapter {
        @Override
        public void mouseClicked(MouseEvent e) {
            displayEventName("Mouse Clicked");
        }

        @Override
        public void mousePressed(MouseEvent e) {
            displayEventName("Mouse Pressed");
        }

        @Override
        public void mouseReleased(MouseEvent e) {
            displayEventName("Mouse Released");
        }

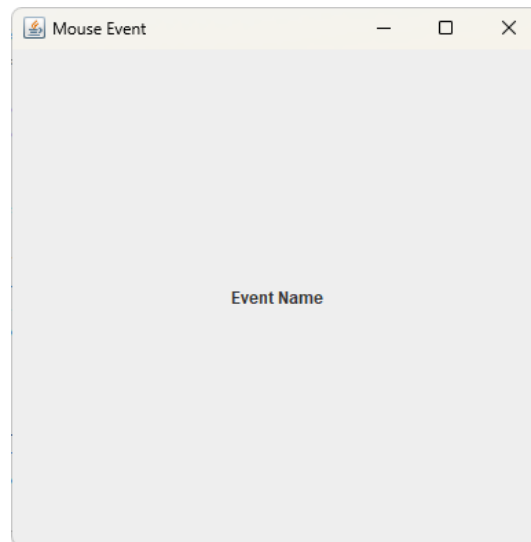
        @Override
        public void mouseEntered(MouseEvent e) {
            displayEventName("Mouse Entered");
        }

        @Override
        public void mouseExited(MouseEvent e) {
            displayEventName("Mouse Exited");
        }
    }
}
```

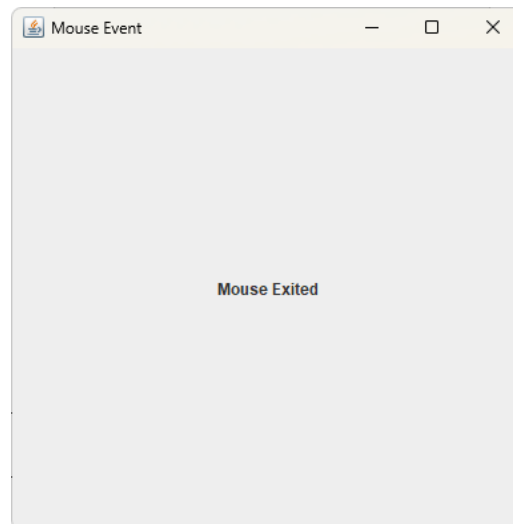


```
private void displayEventName(String eventName) {  
    eventLabel.setText(eventName);  
}  
  
public static void main(String[] args) {  
    SwingUtilities.invokeLater(() -> new MouseEvents());  
}  
}
```

**Input:**



**Output:**



**Q.10:** Develop a java application to implement currency converter (Dollar to BDT, EURO to BDT, Yen to BDT and vice versa), distance converter (meter to KM, miles to KM and vice versa), time converter (hours to minutes, seconds and vice versa) using packages.

**Ans.:**

At first create currency package.

**Code:**

```
//Currency Package
//converter.currency.CurrencyConverter
package converter.currency;

public class CurrencyConverter {
    private static double dollarToBDTRate = 110.50;
    private static double euroToBDTRate = 122.51;
    private static double yenToBDTRate = 0.78;

    public static void setConversionRates(double dollarRate, double euroRate, double yenRate) {
        dollarToBDTRate = dollarRate;
        euroToBDTRate = euroRate;
        yenToBDTRate = yenRate;
    }

    public static double convertDollarToBDT(double amount) {
        return amount * dollarToBDTRate;
    }

    public static double convertEuroToBDT(double amount) {
        return amount * euroToBDTRate;
    }

    public static double convertYenToBDT(double amount) {
        return amount * yenToBDTRate;
    }

    public static double convertBDTToDollar(double amount) {
        return amount / dollarToBDTRate;
    }

    public static double convertBDTToEuro(double amount) {
        return amount / euroToBDTRate;
    }

    public static double convertBDTToYen(double amount) {
        return amount / yenToBDTRate;
    }
}
```

After creating currency package, create distance package.

**Code:**

```
// Distance Package
//converter.distance.DistanceConverter
package converter.distance;

public class DistanceConverter {
    public static double convertMeterToKM(double distance) {
        return distance / 1000.0;
    }

    public static double convertKMToMeter(double distance) {
        return distance * 1000.0;
    }

    public static double convertMilesToKM(double distance) {
        return distance * 1.60934;
    }

    public static double convertKMToMiles(double distance) {
        return distance / 1.60934;
    }
}
```

After creating distance package, create time package.

**Code:**

```
// Time Package
// converter.time.TimeConverter
package converter.Time;

public class TimeConverter {
    public static double convertHoursToMinutes(double hours) {
        return hours * 60.0;
    }

    public static double convertHoursToSeconds(double hours) {
        return hours * 3600.0;
    }

    public static double convertMinutesToHours(double minutes) {
        return minutes / 60.0;
    }

    public static double convertSecondsToHours(double seconds) {
        return seconds / 3600.0;
    }
}
```

Now, after creating currency package, distance package & time package, creating main package. Note, conversion rate already given in the code on each package excluding main package.

### Code:

```
// Main Package
package converter;

import converter.currency.CurrencyConverter;
import converter.distance.DistanceConverter;
import converter.time.TimeConverter;

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

public class Converter extends JFrame {
    private JComboBox<String> conversionTypeComboBox;
    private JComboBox<String> specificConversionComboBox;
    private JTextField inputField;
    private JButton convertButton;

    public Converter() {
        setTitle("Converter");
        setSize(250, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Center the window on the screen
        setLocationRelativeTo(null);

        setLayout(new GridLayout(4, 1));

        String[] conversionTypes = {"Currency", "Distance", "Time"};
        conversionTypeComboBox = new JComboBox<>(conversionTypes);

        specificConversionComboBox = new JComboBox<>();

        inputField = new JTextField();
        convertButton = new JButton("Convert");

        conversionTypeComboBox.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                updateSpecificConversionOptions();
            }
        });

        add(conversionTypeComboBox);
        add(specificConversionComboBox);
        add(inputField);
        add(convertButton);

        updateSpecificConversionOptions();

        convertButton.addActionListener(new ActionListener() {
            @Override
```

```

        public void actionPerformed(ActionEvent e) {
            convert();
        }
    });
}

private void updateSpecificConversionOptions() {
    String selectedConversionType = (String) conversionTypeComboBox.getSelectedItem();
    String[] specificConversionOptions;

    switch (selectedConversionType) {
        case "Currency":
            specificConversionOptions = new String[]{
                "Dollar to BDT", "Euro to BDT", "Yen to BDT",
                "BDT to Dollar", "BDT to Euro", "BDT to Yen"
            };
            break;
        case "Distance":
            specificConversionOptions = new String[]{"Meter to KM", "KM to Meter", "Miles to KM", "KM to Miles"};
            break;
        case "Time":
            specificConversionOptions = new String[]{"Hours to Minutes", "Hours to Seconds", "Minutes to Hours", "Seconds to Hours"};
            break;
        default:
            specificConversionOptions = new String[]{};
            break;
    }

    specificConversionComboBox.setModel(new DefaultComboBoxModel<>(specificConversionOptions));
}

private void convert() {
    String selectedConversion = (String) specificConversionComboBox.getSelectedItem();
    double inputValue = Double.parseDouble(inputField.getText());
    double result = 0;

    switch (selectedConversion) {
        case "Dollar to BDT":
            result = CurrencyConverter.convertDollarToBDT(inputValue);
            break;
        case "Euro to BDT":
            result = CurrencyConverter.convertEuroToBDT(inputValue);
            break;
        case "Yen to BDT":
            result = CurrencyConverter.convertYenToBDT(inputValue);
            break;
        case "BDT to Dollar":
            result = CurrencyConverter.convertBDTToDollar(inputValue);
            break;
        case "BDT to Euro":
            result = CurrencyConverter.convertBDTToEuro(inputValue);
            break;
        case "BDT to Yen":
            result = CurrencyConverter.convertBDTToYen(inputValue);
            break;
        case "Meter to KM":
            result = DistanceConverter.convertMeterToKM(inputValue);
            break;
    }
}

```

```

        case "KM to Meter":
            result = DistanceConverter.convertKMTToMeter(inputValue);
            break;
        case "Miles to KM":
            result = DistanceConverter.convertMilesToKM(inputValue);
            break;
        case "KM to Miles":
            result = DistanceConverter.convertKMTToMiles(inputValue);
            break;
        case "Hours to Minutes":
            result = TimeConverter.convertHoursToMinutes(inputValue);
            break;
        case "Hours to Seconds":
            result = TimeConverter.convertHoursToSeconds(inputValue);
            break;
        case "Minutes to Hours":
            result = TimeConverter.convertMinutesToHours(inputValue);
            break;
        case "Seconds to Hours":
            result = TimeConverter.convertSecondsToHours(inputValue);
            break;
        default:
            break;
    }

    // Display result in a new window
    showResult(result);
}

private void showResult(double result) {
    JFrame resultFrame = new JFrame("Result");
    resultFrame.setSize(300, 100);
    resultFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

    // Center the result window on the screen
    resultFrame.setLocationRelativeTo(null);

    JLabel resultLabel = new JLabel("Result: " + result);
    resultFrame.add(resultLabel);

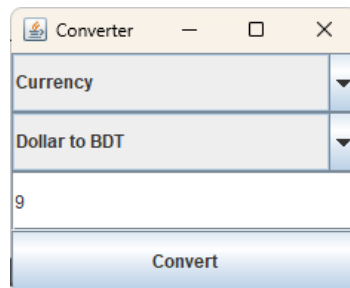
    resultFrame.setVisible(true);
}

public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            Converter converterGUI = new Converter();
            // Center the window on the screen
            converterGUI.setLocationRelativeTo(null);
            converterGUI.setVisible(true);
        }
    });
}
}

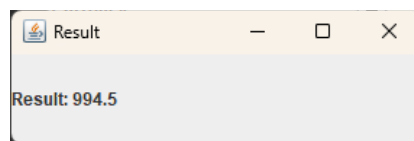
```

**Input-1:**

Select "Currency". After that select "Dollar to BDT". Input 9.



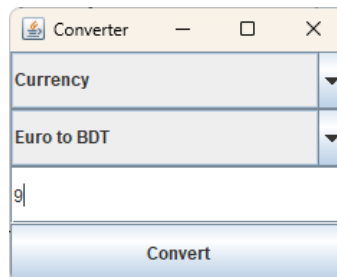
The screenshot shows a window titled "Converter" with a standard Windows title bar (minimize, maximize, close buttons). Inside the window, there are two dropdown menus. The first dropdown is labeled "Currency" and the second is labeled "Dollar to BDT". Below these menus is a text input field containing the number "9". At the bottom of the window is a blue button labeled "Convert".

**Output-1:**

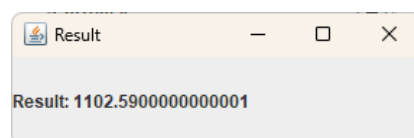
The screenshot shows a window titled "Result" with a standard Windows title bar. Inside the window, the text "Result: 994.5" is displayed.

**Input-2:**

Select "Currency". After that select "Euro to BDT". Input 9.



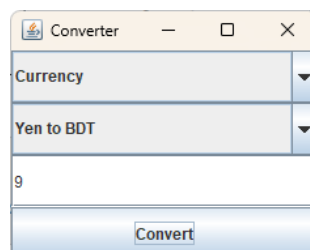
The screenshot shows a window titled "Converter" with a standard Windows title bar. Inside the window, there are two dropdown menus. The first dropdown is labeled "Currency" and the second is labeled "Euro to BDT". Below these menus is a text input field containing the number "9". At the bottom of the window is a blue button labeled "Convert".

**Output-2:**

The screenshot shows a window titled "Result" with a standard Windows title bar. Inside the window, the text "Result: 1102.5900000000001" is displayed.

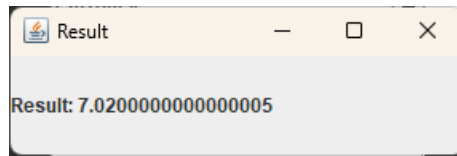
**Input-3:**

Select "Currency". After that select "Yen to BDT". Input 9.



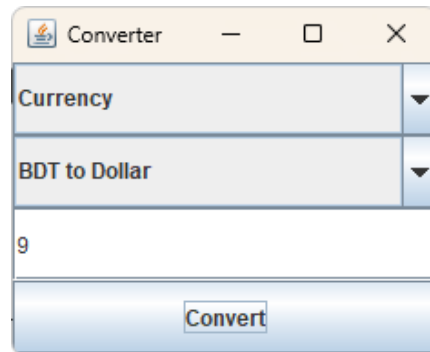
The screenshot shows a window titled "Converter" with a standard Windows title bar. Inside the window, there are two dropdown menus. The first dropdown is labeled "Currency" and the second is labeled "Yen to BDT". Below these menus is a text input field containing the number "9". At the bottom of the window is a blue button labeled "Convert".

### Output-3:



### Input-4:

Select "Currency". After that select "BDT to Dollar". Input 9.

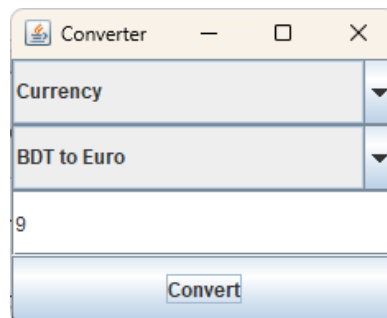


### Output-4:

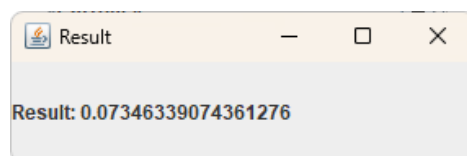


### Input-5:

Select "Currency". After that select "BDT to Euro". Input 9.



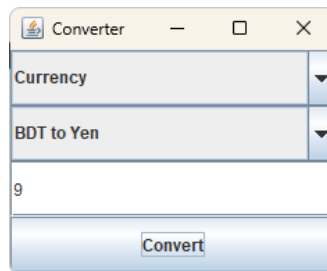
### Output-5:



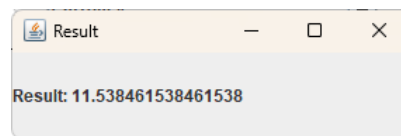


**Input-6:**

Select "Currency". After that select "BDT to Yen". Input 9.



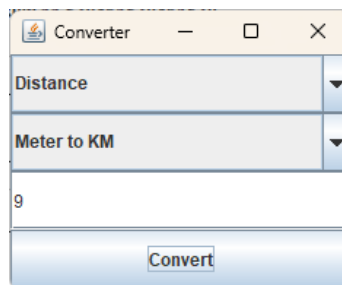
The screenshot shows a window titled "Converter" with a standard Windows interface (minimize, maximize, close buttons). Inside the window, there are two dropdown menus. The first dropdown is labeled "Currency" and the second is labeled "BDT to Yen". Below these menus is a text input field containing the number "9". At the bottom of the window is a button labeled "Convert".

**Output-6:**

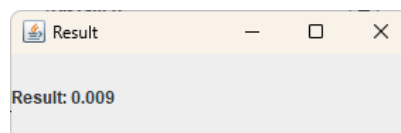
The screenshot shows a window titled "Result" with a standard Windows interface. Inside the window, the text "Result: 11.538461538461538" is displayed.

**Input-7:**

Select "Distance". After that select "Meter to KM". Input 9.



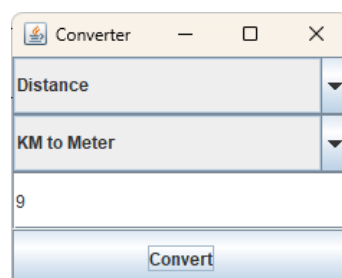
The screenshot shows a window titled "Converter" with a standard Windows interface. Inside the window, there are two dropdown menus. The first dropdown is labeled "Distance" and the second is labeled "Meter to KM". Below these menus is a text input field containing the number "9". At the bottom of the window is a button labeled "Convert".

**Output-7:**

The screenshot shows a window titled "Result" with a standard Windows interface. Inside the window, the text "Result: 0.009" is displayed.

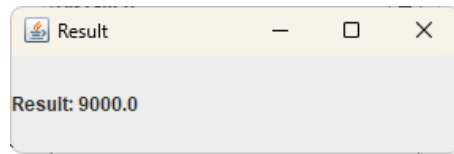
**Input-8:**

Select "Distance". After that select "KM to Meter". Input 9.



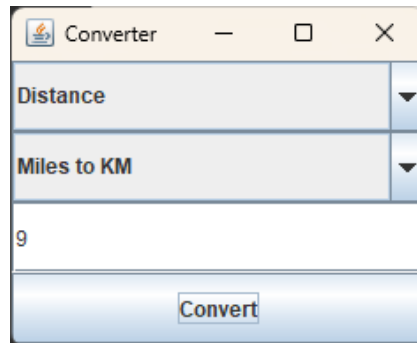
The screenshot shows a window titled "Converter" with a standard Windows interface. Inside the window, there are two dropdown menus. The first dropdown is labeled "Distance" and the second is labeled "KM to Meter". Below these menus is a text input field containing the number "9". At the bottom of the window is a button labeled "Convert".

### Output-8:

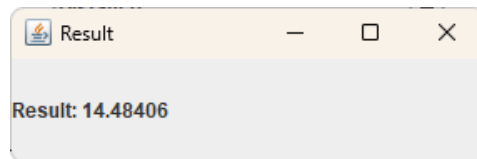


### Input-9:

Select "Distance". After that select "Miles to KM". Input 9.

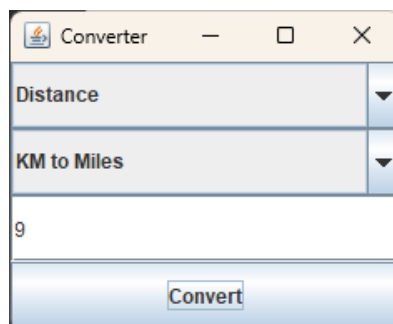


### Output-9:

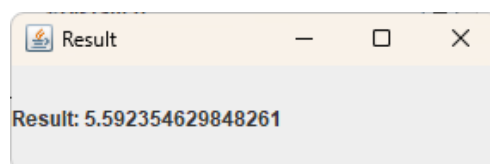


### Input-10:

Select "Distance". After that select "KM to Miles". Input 9.

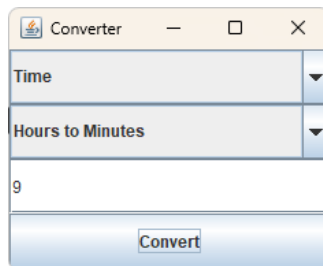


### Output-10:

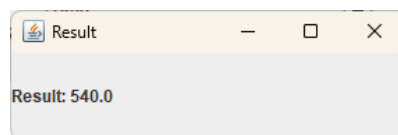


**Input-11:**

Select "Time". After that select "Hours to Minutes". Input 9.



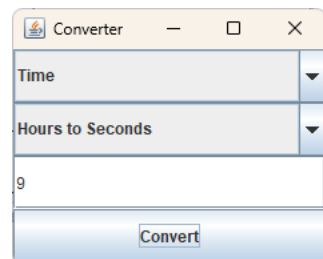
The screenshot shows a window titled "Converter". It has two dropdown menus: the first is set to "Time" and the second is set to "Hours to Minutes". Below these is a text input field containing the number "9". At the bottom is a "Convert" button.

**Output-11:**

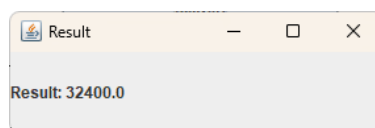
The screenshot shows a window titled "Result". It displays the text "Result: 540.0".

**Input-12:**

Select "Time". After that select "Hours to Seconds". Input 9.



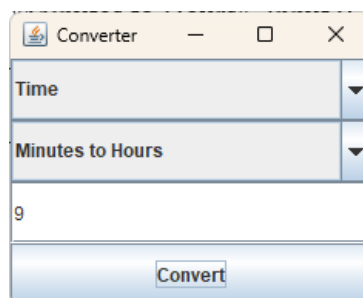
The screenshot shows a window titled "Converter". It has two dropdown menus: the first is set to "Time" and the second is set to "Hours to Seconds". Below these is a text input field containing the number "9". At the bottom is a "Convert" button.

**Output-12:**

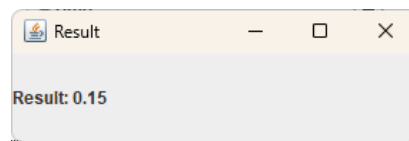
The screenshot shows a window titled "Result". It displays the text "Result: 32400.0".

**Input-13:**

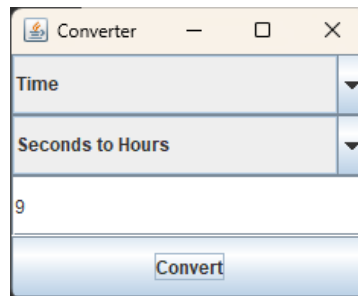
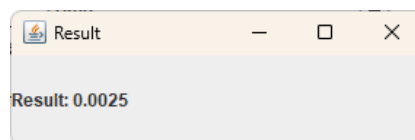
Select "Time". After that select "Minutes to Hours". Input 9.



The screenshot shows a window titled "Converter". It has two dropdown menus: the first is set to "Time" and the second is set to "Minutes to Hours". Below these is a text input field containing the number "9". At the bottom is a "Convert" button.

**Output-13:****Input-14:**

Select "Time". After that select "Seconds to Hours". Input 9.

**Output-14:**

**Q.11:** Write a JAVA program to implement Interface using extends keyword.

**Ans.:**

**Code:**

```
package keywords;

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

// Define an interface
interface Animal {
    void makeSound();
}

// Implement the interface using the extends keyword
class Dog implements Animal {
    @Override
    public void makeSound() {
        JOptionPane.showMessageDialog(null, "Woof! Woof!");
    }
}
```

```

class Cat implements Animal {
    @Override
    public void makeSound() {
        JOptionPane.showMessageDialog(null, "Meow! Meow!");
    }
}

```

```

class Other implements Animal {
    @Override
    public void makeSound() {
        JOptionPane.showMessageDialog(null, "Unknown!");
    }
}

```

```

// Main GUI class
public class Interface extends JFrame {

    public Interface() {
        super("Animal Sounds");

        // Create buttons for each animal
        JButton dogButton = new JButton("Dog");
        JButton catButton = new JButton("Cat");
        JButton otherButton = new JButton("Other");

        // Add action listeners to the buttons
        dogButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Dog myDog = new Dog();
                myDog.makeSound();
            }
        });

        catButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Cat myCat = new Cat();
                myCat.makeSound();
            }
        });

        otherButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                Other myOther = new Other();
                myOther.makeSound();
            }
        });

        // Create a panel and add buttons to it
        JPanel panel = new JPanel();
        panel.setLayout(new GridLayout(3, 1));
        panel.add(dogButton);
        panel.add(catButton);
        panel.add(otherButton);

        // Set up the frame

```

```

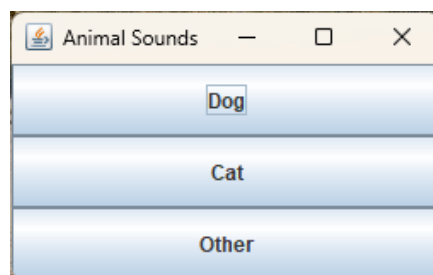
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        add(panel, BorderLayout.CENTER);
        pack();
        setLocationRelativeTo(null); // Center the frame
        setVisible(true);
    }

    public static void main(String[] args) {
        // Create the GUI frame
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new Interface();
            }
        });
    }
}

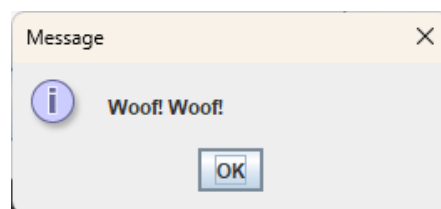
```

### Input-1:

Select "Dog".

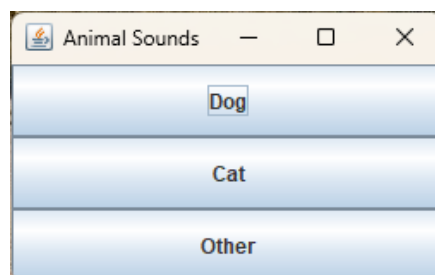


### Output-1:

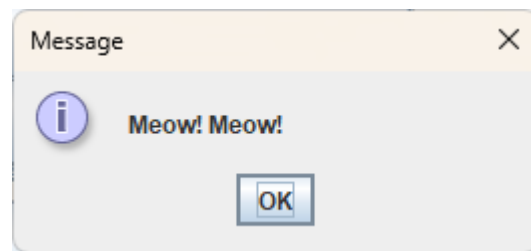


### Input-2:

Select "Cat".

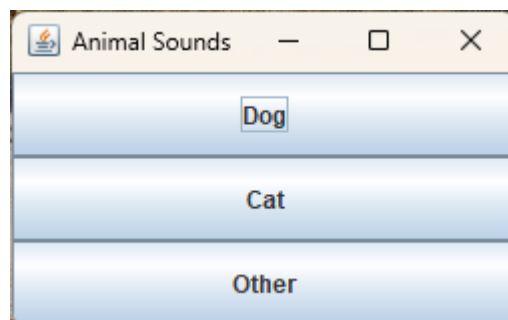


**Output-2:**



**Input-3:**

Select "Other".



**Output-3:**

