JAVA PROGRAMMING

LAB EXERCISES

EX.NO:01

CELSIUS TO FAHRENHEIT CONVERSION

AIM:

To write a Java program to read the temperature in Celsius and convert into Fahrenheit.

ALGORITHM:

```
1) Start the program.
2) Import the Scanner class to read input from the user.
3) Create a Scanner object for input.
4) Declare a variable to store the temperature in Celsius.
5) Declare a variable to store the temperature in Fahrenheit.
6) Ask the user to enter the temperature in Celsius.
7) Read the temperature in Celsius from the user.
8) Convert Celsius to Fahrenheit using the formula $F = (C \times 9/5) + 32$.
9) Display the temperature in Fahrenheit.
10) End the program.
```

PROGRAM:

```
import java.util.Scanner;

class EXNO01 {

   public static void main(String args[]) {

      float cel, far;

      Scanner s = new Scanner(System.in);

      System.out.println("Enter temperature in Celsius:");

      cel = s.nextFloat();

      far = (cel * 9 / 5) + 32;

      System.out.println("Temp in Fahrenheit: " + far);

   }

}
```

RESULT:

Thus, the above Java program is executed successfully and the temperature in Fahrenheit is displayed.

JAVA PROGRAMMING

LAB EXERCISES

EX.NO:02

LARGEST NUMBER USING CONDITIONAL OPERATOR

AIM:

To write a Java program to read 2 integers and find the largest number using conditional operator.

ALGORITHM:

1) Start the program.
2) Import the Scanner class.
3) Create a Scanner object.
4) Declare two integer variables.
5) Declare a variable to store the largest number.
6) Ask the user to enter two numbers.
7) Read the two numbers from the user.
8) Compare the numbers using the conditional operator.
9) Display the largest number.
10) End the program.

PROGRAM:

```
import java.util.Scanner;

public class EXNO02 {

  public static void main(String args[]) {

    int a, b, big;

    Scanner scan = new Scanner(System.in);

    System.out.println("Enter two numbers:");

    a = scan.nextInt();

    b = scan.nextInt();

    big = (a > b) ? a : b;

    System.out.println("Largest = " + big);

  }

}
```

RESULT:

Thus, the above Java program is executed successfully and the largest number is displayed.

LAB EXERCISES

EX.NO:03

IMPLEMENTING COMMAND LINE ARGUMENTS

AIM:

To write a Java program to implement command line arguments.

ALGORITHM:

1) Start the program.
2) Read the radius value from command line arguments.
3) Convert the argument from String to float.
4) Declare a variable to store the area of the circle.
5) Calculate the area using the formula Area = 3.14 * r * r.
6) Optionally, format the area to two decimal places.
7) Handle invalid or missing input if any.
8) Display the radius.
9) Display the area of the circle.
10) End the program.

PROGRAM:

```
class EXNO03 {
public static void main(String args[]) {
float radius, area;
radius = Float.parseFloat(args[0]);
System.out.println("The radius of a circle is: " + radius);
area = 3.14f * radius * radius;
System.out.println("The area of a circle is: " + area);
}
}
```

RESULT:

Thus, the above Java program is executed successfully and the radius and area of the circle are displayed.

JAVA PROGRAMMING

LAB EXERCISES

EX.NO:04

USING ARRAYS

AIM:

To write a Java program to find the sum and average of your tenth standard marks.

ALGORITHM:

1)Start the program.

2)Import the Scanner class to read input from the user (import java.util.Scanner;).

3)Declare an integer array to store marks for 5 subjects.

4)Initialize a variable sum to 0 to hold the total of marks.

5)Create a Scanner object to read input from the user.

6)Use a loop to read marks for each subject and store them in the array.

7)Add each mark to sum inside the loop to calculate total marks.

8)Calculate the average as average = sum / 5.0.

9)Display the total marks and average marks to the user.

10)End the program.

PROGRAM:

```
import java.util.Scanner;
class EX04 {
   public static void main(String args[]) {
      Scanner s = new Scanner(System.in);
      int marks[] = new int[5];
      int sum = 0;
      double aver;
      System.out.println("Enter your tenth standard marks one by one:");
      for (int i = 0; i < 5; i++) {
         System.out.print("Subject " + (i + 1) + ": ");
         marks[i] = s.nextInt();
         sum += marks[i];
      }
      aver = sum / 5.0;
```

```
        System.out.println("Total Marks: " + sum);
        System.out.println("Average Marks: " + aver);
    }
}
```

RESULT:

Thus, the above Java program is executed successfully and the total and average marks are displayed.

EX.NO:05

SORTING NAMES USING BUBBLE SORT

AIM:

To write a Java Program to sort 10 student names in alphabetical order using bubble sort.

ALGORITHM:

1)Start the program.

2)Import the Scanner class to read input from the user (import java.util.Scanner;).

3)Declare an array of strings to store 10 student names.

4)Read 10 student names from the user using a loop.

5)Implement bubble sort to arrange names in alphabetical order.

6)Compare names and swap if the current name is lexicographically greater than the next name.

7) Optionally, print a message indicating sorting is complete

8)Close the Scanner to prevent resource leak.

9) Display the sorted names to the user.

10)End the program.

PROGRAM:

```
import java.util.Scanner;
class EXNO05 {
    public static void main(String args[]) {
        Scanner s = new Scanner(System.in);
        String names[] = new String[10];
        System.out.println("Enter 10 student names:");
        for (int i = 0; i < names.length; i++) {
            System.out.print("Name " + (i + 1) + ": ");
            names[i] = s.nextLine();
        }
        for (int i = 0; i < names.length - 1; i++) {
            for (int j = 0; j < names.length - 1 - i; j++) {
                if (names[j].compareTo(names[j + 1]) > 0) {
                    String temp = names[j];
                    names[j] = names[j + 1];
                    names[j + 1] = temp;
                }
            }
        }
        System.out.println("\nStudent names in alphabetical order:");
        for (int i = 0; i < names.length; i++) {
            System.out.println(names[i]);
        }
    }
}
```

RESULT:

Thus, the above Java program is executed successfully and the student names are displayed in alphabetical order.

EX.NO:06

USING CONSTRUCTORS

AIM:

To write a Java program to collect student details using constructors.

JAVA PROGRAMMING

LAB EXERCISES

ALGORITHM:

1)Start the program.

2)Create a Student class with variables for name, age, roll number, and course.

3)Define a constructor to initialize student details.

4)Define a method display() to show student details.

5)In main, read the number of students from the user.

6)Use a loop to read each student’s details and create Student objects.

7) Optionally, display a message before and after showing all students.

8)Handle invalid or missing input if any.

9) Display all student details using the display() method.

10)End the program.

PROGRAM:

```
import java.util.Scanner;
class Student {
   String name;
   int age;
   String rollno;
   String course;
   Student(String n, int a, String r, String c) {
      name = n;
      age = a;
      rollno = r;
      course = c;
   }
   void display() {
      System.out.println("Name: " + name);
      System.out.println("Age: " + age);
      System.out.println("Roll Number: " + rollno);
      System.out.println("Course: " + course);
```

```
    }
}
public class EXNO06 {
    public static void main(String args[]) {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter the number of students: ");
        int n = s.nextInt();
        s.nextLine(); // Consume newline
        Student students[] = new Student[n];
        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for student " + (i + 1) + ":");
            System.out.print("Name: ");
            String name = s.nextLine();
            System.out.print("Age: ");
            int age = s.nextInt();
            s.nextLine(); // Consume newline
            System.out.print("Roll Number: ");
            String rollno = s.nextLine();
            System.out.print("Course: ");
            String course = s.nextLine();
            students[i] = new Student(name, age, rollno, course);
        }
        System.out.println("\nStudent Details:");
        for (int i = 0; i < n; i++) {
            students[i].display();
            System.out.println();
        }
    }
}
```

RESULT:

Thus, the above Java program is executed successfully and the details of all students are displayed.

JAVA PROGRAMMING

LAB EXERCISES

EX.NO:07

USING METHOD OVERLOADING

AIM:

To write a Java program to calculate area of rectangle, triangle and square using method overloading.

ALGORITHM:

1)Start the program.

2)Create a class Area with three overloaded find Area methods:

- For rectangle (length × breadth).
- For triangle (0.5 × base × height).
- For square (side × side).

3)In the main method, create an object of the Area class.

4)Read the length and breadth of the rectangle.

5)Call find Area for rectangle and store the result.

6)Read the base and height of the triangle.

7)Call find Area for triangle and store the result.

8)Read the side of the square.

9)Display the areas of the rectangle, triangle, and square.

10)End the program.

PROGRAM:

```
import java.util.Scanner;
class Area {
   double findArea(double length, double breadth) {
      return length * breadth;
   }
   double findArea(double base, double height, boolean isTriangle) {
        return 0.5 * base * height;
    }
    double findArea(double side) {
        return side * side;
    }
}
class EXNO07 {
    public static void main(String args[]) {
        Scanner scan = new Scanner(System.in);
        Area c = new Area();
        System.out.println("Enter Length and Breadth of Rectangle:");
        double le = scan.nextDouble();
        double br = scan.nextDouble();
        double rect = c.findArea(le, br);
        System.out.println("Enter Base and Height of Triangle:");
        double ba = scan.nextDouble();
        double ht = scan.nextDouble();
        double tri = c.findArea(ba, ht, true);
        System.out.println("Enter Side of Square:");
        double s = scan.nextDouble();
        double square = c.findArea(s);
        System.out.println("Area of the rectangle: " + rect);
        System.out.println("Area of the triangle: " + tri);
        System.out.println("Area of the square: " + square);
    }
}
```

RESULT:

Thus, the above Java program is executed successfully and the areas of rectangle, triangle, and square are displayed.

JAVA PROGRAMMING

LAB EXERCISES

EX.NO:08

INHERITANCE AND OVERRIDING METHODS

AIM:

To write a Java program to create a class called Shape with methods called getPerimeter() and getArea(). Create a subclass called Circle that overrides the getPerimeter() and getArea() methods to calculate the area and perimeter of a circle.

ALGORITHM:

1)Start the program.

2)Create a base class Shape with methods getPerimeter() and getArea() returning 0.0.

3)Create a subclass Circle that extends Shape and has a radius variable.

4)Override getPerimeter() to calculate perimeter as 2 * π * r.

5)Override getArea() to calculate area as π * r * r.

6)In main, read the radius of the circle from the user and create a Circle object, Display the perimeter and area of the circle.

7)End the program.

PROGRAM:

```
import java.util.Scanner;
class Shape {
  double getPerimeter() {
    return 0.0;
  }
  double getArea() {
    return 0.0;
  }
}
class Circle extends Shape {
  double radius;


  Circle(double r) {
    radius = r;
  }
  @Override
  double getPerimeter() {
    return 2 * 22.0 / 7.0 * radius;
  }
  @Override
  double getArea() {
    return 22.0 / 7.0 * radius * radius;
  }
}
class EXNO8 {
  public static void main(String args[]) {
    Scanner scan = new Scanner(System.in);
    System.out.println("Enter Radius of Circle:");
    double rad = scan.nextDouble();
    Circle = new Circle(rad);
    System.out.println("Perimeter of the circle: " + circle.getPerimeter());
    System.out.println("Area of the circle: " + circle.getArea());
  }
}
```

RESULT:

Thus, the above Java program is executed successfully and the perimeter and area of the circle are displayed.

EX.NO:09

IMPLEMENTING INTERFACE

AIM:

To write a Java program to create an interface Shape with the getArea() method. Create three classes Rectangle, Circle, and Triangle that implement the Shape Interface. Implement the getArea() method for each of the three classes.

ALGORITHM:

```
1)Start the program.
2)Create an interface Shape with method getArea().
3)Create classes Rectangle, Triangle, and Circle implementing Shape.
4)Define getArea() for each class according to its shape.
5)In main, read necessary dimensions from the user, Create objects of each class and calculate their areas.
6)Display the areas of rectangle, triangle, and circle.
7)End the program.
```

PROGRAM:

```
import java.util.Scanner;
interface Shape {
   double getArea();
}
class Rectangle implements Shape {
   double length, breadth;

   Rectangle(double l, double b) {
      length = l;
      breadth = b;
   }
   public double getArea() {
      return length * breadth;
   }
}
```

```
class Circle implements Shape {
   double radius;

   Circle(double r) {
      radius = r;
   }
   public double getArea() {
      return 22.0 / 7.0 * radius * radius;
   }
}
class Triangle implements Shape {
   double base, height;
   Triangle(double b, double h) {
      base = b;
      height = h;
   }
   public double getArea() {
      return 0.5 * base * height;
   }
}
class EXNO09 {
   public static void main(String args[]) {
      Scanner scan = new Scanner(System.in);
      System.out.println("Enter Length and Breadth of Rectangle:");
      double le = scan.nextDouble();
      double br = scan.nextDouble();
      Rectangle rect = new Rectangle(le, br);
      System.out.println("Enter Base and Height of Triangle:");
      double ba = scan.nextDouble();
      double ht = scan.nextDouble();
      Triangle tri = new Triangle(ba, ht);
      System.out.println("Enter Radius of Circle:");
```

```
        double r = scan.nextDouble();
        Circle cir = new Circle(r);
        System.out.println("Area of Rectangle: " + rect.getArea());
        System.out.println("Area of Triangle: " + tri.getArea());
        System.out.println("Area of Circle: " + cir.getArea());
    }
}
```

RESULT:

Thus, the above Java program is executed successfully and the areas of rectangle, triangle, and circle are displayed.

EX.NO:10

USING SWING COMPONENTS

AIM:

To write a Java program to create a panel with three buttons, labeled Red, Blue and Yellow, so that clicking each button results in the background color changing to the appropriate color.

ALGORITHM:

1)Start the program.

2)Import Swing and AWT packages for GUI components.

3)Create a JFrame as the main window.

4)Create a JPanel and three JButtons labeled Red, Blue, and Yellow.

5)Add buttons to the panel.

6)Define an ActionListener to change the panel’s background color based on the button clicked.

7)Add the panel to the frame and make the frame visible.

PROGRAM:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
public class EXNO10 {
    public static void main(String args[]) {
        JFrame frame = new JFrame("Color Changer");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel panel = new JPanel();
        JButton redButton = new JButton("Red");
        JButton blueButton = new JButton("Blue");
        JButton yellowButton = new JButton("Yellow");
        panel.add(redButton);
        panel.add(blueButton);
        panel.add(yellowButton);
        ActionListener colorChanger = new ActionListener() {
            public void actionPerformed(ActionEvent e) {
                if (e.getSource() == redButton) {
                    panel.setBackground(Color.RED);
                } else if (e.getSource() == blueButton) {
                    panel.setBackground(Color.BLUE);
                } else if (e.getSource() == yellowButton) {
                    panel.setBackground(Color.YELLOW);
                }
            }
        };
        redButton.addActionListener(colorChanger);
        blueButton.addActionListener(colorChanger);
        yellowButton.addActionListener(colorChanger);
        frame.add(panel);
        frame.setVisible(true);
    }
}
```

RESULT:

Thus, the above Java program is executed successfully and clicking each button changes the background color of the panel accordingly.