Ramaiah Institute of Technology
(Autonomous Institute, Affiliated to VTU)
Department of Computer Science & Engineering

Object Oriented Programming Laboratory (CSL37)

Semester:III **Week #: 04** **Section:A,B,C**

.

Method Overloading and Overriding

1. Write a Java program that demonstrates method overloading with methods to add two integers, two doubles, and two strings.
2. Create a class with a method to find the area of a rectangle using method overloading. The method should accept dimensions in both integer and double data types.
3. Develop a Java program that includes a class with a method to concatenate two strings. Overload the method to concatenate three strings as well.
4. Create a base class Shape with a method calculateArea. Derive two classes Circle and Rectangle from Shape and override the calculateArea method in each derived class.
5. Develop a Java program that demonstrates the use of method overriding with a base class Vehicle and two derived classes Car and Motorcycle. Override the start method in each derived class.
6. Write a program that includes an interface Drawable with a method draw(). Implement this interface in two classes Circle and Rectangle with their own versions of the draw method.

1.

```
public class MethodOverloadingDemo {
    // Method to add two integers
    static int add(int a, int b) {
        return a + b;
    }
    // Method to add two doubles
    static double add(double a, double b) {
        return a + b;
    }
    // Method to concatenate two strings
    static String concatenate(String s1, String s2) {
        return s1 + s2;
    }
    // Method to concatenate three strings
    static String concatenate(String s1, String s2, String s3) {
        return s1 + s2 + s3;
    }
    public static void main(String[] args) {
        // Testing method overloading
        System.out.println("Sum of integers: " + add(5, 10));
        System.out.println("Sum of doubles: " + add(3.5, 2.5));
        System.out.println("Concatenation of two strings: " + concatenate("Hello, ", "World!"));
        System.out.println("Concatenation of three strings: " + concatenate("I", " Love", " Java"));
    }
}
```

Output :

```
Sum of integers: 15
Sum of doubles: 6.0
Concatenation of two strings: Hello, World!
Concatenation of three strings: I Love Java
```

2.

```
        public class AreaCalculator {
    // Method to calculate area of a rectangle with integer dimensions
    static int calculateArea(int length, int width) {
        return length * width;
    }
    // Method to calculate area of a rectangle with double dimensions
    static double calculateArea(double length, double width) {
        return length * width;
    }
    public static void main(String[] args) {
        // Testing method overloading for rectangle area
        System.out.println("Area of rectangle with integer dimensions: " + calculateArea(5, 10));
        System.out.println("Area of rectangle with double dimensions: " + calculateArea(3.5, 2.5));
    }
}
```

output:

```
Area of rectangle with integer dimensions: 50
```

Area of rectangle with double dimensions: 8.75

3.

```
public class StringConcatenationDemo {
    // Method to concatenate two strings
    static String concatenate(String s1, String s2) {
        return s1 + s2;
    }

    // Method to concatenate three strings
    static String concatenate(String s1, String s2, String s3) {
        return s1 + s2 + s3;
    }

    public static void main(String[] args) {
        // Testing method overloading for string concatenation
        System.out.println("Concatenation of two strings: " + concatenate("Hello, ", "World!"));
        System.out.println("Concatenation of three strings: " + concatenate("I", " Love", " Java"));
    }
}
```

output:

```
Concatenation of two strings: Hello, World!
Concatenation of three strings: I Love Java
```

4.

```
class Shape {
    double calculateArea() {
        return 0.0;
    }
}

class Circle extends Shape {
    double radius;

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

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

class Rectangle extends Shape {
    double length;
    double width;

    Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    @Override
    double calculateArea() {
        return length * width;
    }
}

public class ShapeDemo {
    public static void main(String[] args) {
        // Testing method overriding with Shape, Circle, and Rectangle
        Shape circle = new Circle(5.0);
        Shape rectangle = new Rectangle(4.0, 6.0);

        System.out.println("Area of Circle: " + circle.calculateArea());
        System.out.println("Area of Rectangle: " + rectangle.calculateArea());
    }
}
```

output:

```
Area of Circle: 78.53981633974483
Area of Rectangle: 24.0
```

5.

```
class Vehicle {
    void start() {
        System.out.println("Vehicle is starting...");
    }
}

class Car extends Vehicle {
    @Override
    void start() {
        System.out.println("Car is starting...");
    }
}

class Motorcycle extends Vehicle {
    @Override
    void start() {
        System.out.println("Motorcycle is starting...");
    }
}

public class VehicleDemo {
    public static void main(String[] args) {
        // Testing method overriding with Vehicle, Car, and Motorcycle
        Vehicle vehicle = new Vehicle();
        Car car = new Car();
        Motorcycle motorcycle = new Motorcycle();

        vehicle.start();
        car.start();
        motorcycle.start();
    }
}
```

Output:
Vehicle is starting...
Car is starting...
Motorcycle is starting…

6.

```
interface Drawable {
    void draw();
}

class Circle implements Drawable {
    @Override
    public void draw() {
        System.out.println("Drawing Circle");
    }
}

class Rectangle implements Drawable {
    @Override
    public void draw() {
        System.out.println("Drawing Rectangle");
    }
}

public class DrawingDemo {
    public static void main(String[] args) {
        // Testing interface implementation with Circle and Rectangle
        Drawable circle = new Circle();
        Drawable rectangle = new Rectangle();

        circle.draw();
        rectangle.draw();
    }
}
```

Output:
Drawing Circle
Drawing Rectangle