Object Oriented Programming using Java Laboratory (DJS23FLES201)
Academic Year 2023-24

EXPERIMENT NO. 7

NAME- SHAIKH ARSHAD AJIJ

ROLL NO.- B007

SAP ID – 60019230064

BRANCH- CSE-ICB

BATCH- B1

AIM / OBJECTIVE:

To implement Constructors and constructor overloading

- WAOOP to count the no. of objects created of a class using constructors.

  Code-

```
class MyClass {
    private static int objectCount = 0;

    // Constructor
    public MyClass() {
        // Increment the object count whenever a new object is created
        objectCount++;
    }

    // Method to get the count of objects created
    public static int getObjectCount() {
        return objectCount;
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        // Creating objects of MyClass
        MyClass obj1 = new MyClass();
        MyClass obj2 = new MyClass();
        MyClass obj3 = new MyClass();

        // Getting the count of objects created
        int count = MyClass.getObjectCount();
        System.out.println("Number of objects created: " + count);
    }
}
```

Output-

```
C:\Users\Arshad\Desktop\study\java>java Main
Number of objects created: 3

C:\Users\Arshad\Desktop\study\java>
```

- WAP to display area of square and rectangle using the concept of overloaded constructor (use parameterized, non-parameterized and copy constructor).

Code-

```
class Shape {
    double area;

    // Non-parameterized constructor
    public Shape() {
        area = 0;
    }
```

```
    // Parameterized constructor for square
    public Shape(double side) {
        area = side * side;
    }

    // Parameterized constructor for rectangle
    public Shape(double length, double width) {
        area = length * width;
    }

    // Copy constructor
    public Shape(Shape shape) {
        this.area = shape.area;
    }

    // Method to display area
    public void displayArea() {
        System.out.println("Area: " + area);
    }
}

public class Main {
    public static void main(String[] args) {
        // Using non-parameterized constructor
        Shape shape1 = new Shape();
        System.out.println("Area of shape 1 (l,b=0):");
        shape1.displayArea();

        // Using parameterized constructor for square
        Shape shape2 = new Shape(5);
```

```
        System.out.println("\nArea of shape 2 (Square):");
        shape2.displayArea();

        // Using parameterized constructor for rectangle
        Shape shape3 = new Shape(4, 6);
        System.out.println("\nArea of shape 3 (Rectangle):");
        shape3.displayArea();

        // Using copy constructor
        Shape shape4 = new Shape(shape3);
        System.out.println("\nArea of shape 4 (Copy of shape 3):");
        shape4.displayArea();
    }
}
```

Output-

```
C:\Users\Arshad\Desktop\study\java>java Main
Area of shape 1 (l,b=0):
Area: 0.0

Area of shape 2 (Square):
Area: 25.0

Area of shape 3 (Rectangle):
Area: 24.0

Area of shape 4 (Copy of shape 3):
Area: 24.0
```

CONCLUSION:

The first program demonstrated how to count the number of objects created from a class using constructors. By incrementing a static counter within the constructor, we could keep track of the number of instances created.

The second program showcased overloaded constructors to calculate the area of both squares and rectangles. By providing different constructors based on the number and types of

parameters, we could create instances of the Shape class representing squares or rectangles and calculate their areas accordingly.

Website References: javapoint.com