

## CSYE 6200 Assignment-5

### Problem 1:

#### Problem Description:

Write a program which will create an Octagon object with a side value 9. Then display the objects area and its perimeter. In the same program, create a clone of the object and compare the two objects and print the result of comparison. While creating Octagon class, extend from GeometricObject and implement Comparable and Cloneable interfaces to understand the interface concepts.

#### Analysis:

#### Algorithm:

- *Main()* – This method contains the Octagon object creation with a side value 9. It also contains the function calls of *getArea()* and *getPerimeter()* from the Octagon object.
- *Class Octagon* – The class implementation starts with extending the GeometricObject and implementation of Comparable and Cloneable interfaces. It has one constructor which initializes the octagon side and a *compareTo* method which takes an object as a parameter and compares it with the current instance. The following statements contain all the method inside this class.
  - *compareTo()*
    - If area of two Octagon objects is similar, returns 0.
    - If area of current instance is greater, returns 1.
    - If the current Octagon instance is less, returns -1.
  - *getArea()*
    - Returns the area of the Octagon based on the side value
    - Formula:  $\text{area} = (2 + 4/\sqrt{2}) * \text{side} * \text{side}$
  - *getPerimeter()*
    - Returns the perimeter of the Octagon based on the side value
    - Formula:  $\text{perimeter} = 8 * \text{side}$
  - *Clone()*
    - Creates a clone or a copy when requested
    - Raises a CloneNotSupportedException

#### Learnings:

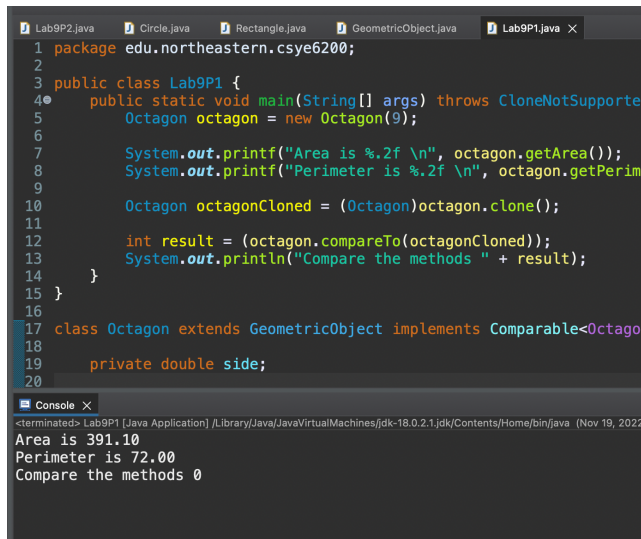
Learned how to implement interfaces in Java.

Learned how to create a clone of an object with the help of cloneable interface.

## Source Code:

```
1. package edu.northeastern.csye6200;
2.
3. public class Lab9P1 {
4.     public static void main(String[] args) throws CloneNotSupportedException {
5.         Octagon octagon = new Octagon(9);
6.
7.         System.out.printf("Area is %.2f \n", octagon.getArea());
8.         System.out.printf("Perimeter is %.2f \n", octagon.getPerimeter());
9.
10.        Octagon octagonCloned = (Octagon)octagon.clone();
11.
12.        int result = (octagon.compareTo(octagonCloned));
13.        System.out.println("Compare the methods " + result);
14.    }
15. }
16.
17. class Octagon extends GeometricObject implements Comparable<Octagon>, Cloneable {
18.
19.     private double side;
20.
21.     Octagon(double side) {
22.         this.side = side;
23.     }
24.
25.     @Override
26.     public int compareTo(Octagon o) {
27.         if(getArea() == o.getArea()) {
28.             return 0;
29.         }
30.         else if (getArea() > o.getArea()) {
31.             return 1;
32.         }
33.         else {
34.             return -1;
35.         }
36.     }
37.
38.     @Override
39.     public double getArea() {
40.         return (2 + 4 / Math.sqrt(2)) * side * side;
41.     }
42.
43.     @Override
44.     public double getPerimeter() {
45.         return 8 * side;
46.     }
47.
48.     @Override
49.     public Object clone() throws CloneNotSupportedException {
50.         return super.clone();
51.     }
52.
53. }
54.
```

## Screenshot:



```
1 package edu.northeastern.csye6200;
2
3 public class Lab9P1 {
4     public static void main(String[] args) throws CloneNotSupportedException {
5         Octagon octagon = new Octagon(9);
6
7         System.out.printf("Area is %.2f \n", octagon.getArea());
8         System.out.printf("Perimeter is %.2f \n", octagon.getPerimeter());
9
10        Octagon octagonCloned = (Octagon)octagon.clone();
11
12        int result = (octagon.compareTo(octagonCloned));
13        System.out.println("Compare the methods " + result);
14    }
15 }
16
17 class Octagon extends GeometricObject implements Comparable<Octagon> {
18     private double side;
19 }
20
```

Console X

```
<terminated> Lab9P1 [Java Application] /Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java (Nov 19, 2022)
Area is 391.10
Perimeter is 72.00
Compare the methods 0
```

## Problem 2:

### Problem Description:

Write a program to create an array of five geometric objects such as Square, Rectangle and Circle, while square being created thrice with a difference values. For each object display its area and print the way to color that object. Make sure that the required geometric objects implement Colorable interface which contain a void method *howToColor*. Also, the program should be able to display the area of the geometric object and the way to color it. If the object is not colorable, displaying the area of the object is enough.

### Analysis:

### Algorithm:

*Main()* – The main method contains the declaration and the initialization of the geometric objects array. Square with different sides 2, 5 and 4.5. Circle with radius 5 and the Rectangle object with the width and height 3 and 4 respectively. After the initialization, area and the way to color the object gets printed.

*Interface Colorable* – Colorable interface has been declared with a void method *howToColor*. That means every object that implements this interface must implement its methods.

*Class Square* – This class extends *GeometricObject* and implements *Colorable* interface. Contains constructor to initialize its side. It also overrides the super class *getArea* and *getPerimeter* methods.

- *getArea()* – returns the area of the square as per the side value.
  - Formula:  $\text{side} * \text{side}$

- *getPerimeter()* – returns the perimeter of the square as per the side value.
  - Formula: side \* 4

## Learnings:

This program creates a way to strengthen my knowledge on interfaces.

## Source Code:

```
1. package edu.northeastern.csye6200;
2.
3. public class Lab9P2 {
4.     public static void main(String[] args) {
5.         GeometricObject[] geometricObjects = new GeometricObject[5];
6.         geometricObjects[0] = new Square(2);
7.         geometricObjects[1] = new Circle(5);
8.         geometricObjects[2] = new Square(5);
9.         geometricObjects[3] = new Rectangle(3, 4);
10.        geometricObjects[4] = new Square(4.5);
11.
12.        for (int i = 0; i < geometricObjects.length; i++) {
13.            System.out.printf("Area is %.2f \n", geometricObjects[i].getArea());
14.            if (geometricObjects[i] instanceof Colorable) {
15.                System.out.println(((Colorable)geometricObjects[i]).howToColor());
16.            }
17.            System.out.printf("\n");
18.        }
19.    }
20. }
21.
22. interface Colorable {
23.     String howToColor();
24. }
25.
26. class Square extends GeometricObject implements Colorable {
27.
28.     private double side;
29.
30.     Square(double side) {
31.         this.side = side;
32.     }
33.
34.     @Override
35.     public double getArea() {
36.         return side * side;
37.     }
38.
39.     @Override
40.     public double getPerimeter() {
41.         return side * 4;
42.     }
43.
44.     @Override
45.     public String howToColor() {
46.         return "Color all four sides";
47.     }
48. }
49. }
50.
```

## Rectangle Class:

```
1. package edu.northeastern.csye6200;
2.
3. public class Rectangle extends GeometricObject{
4.
5.     private double width;
6.     private double height;
7.
8.     public Rectangle(double width, double height) {
9.         this.width = width;
10.        this.height = height;
11.    }
12.
13.    @Override
14.    public double getArea() {
15.        return width * height;
16.    }
17.
18.    @Override
19.    public double getPerimeter() {
20.        return 2 * width + 2 * height;
21.    }
22. }
23.
```

## GeometricObject class:

```
1. package edu.northeastern.csye6200;
2.
3. // GeometricObject.java: The abstract GeometricObject class
4.
5. public abstract class GeometricObject {
6.
7.     private String color = "white";
8.     private boolean filled;
9.
10.    /** Default construct */
11.    protected GeometricObject() {
12.    }
13.
14.    /** Construct a geometric object */
15.    protected GeometricObject(String color, boolean filled) {
16.        this.color = color;
17.        this.filled = filled;
18.    }
19.
20.    /** Getter method for color */
21.    public String getColor() {
22.        return color;
23.    }
24.
25.    /** Setter method for color */
26.    public void setColor(String color) {
27.        this.color = color;
28.    }
29.
30.    /**
31.     * Getter method for filled. Since filled is boolean, so, the get method
32.     * name is isFilled
33.     */
34. }
```

```

34. public boolean isFilled() {
35.     return filled;
36. }
37.
38. /** Setter method for filled */
39. public void setFilled(boolean filled) {
40.     this.filled = filled;
41. }
42.
43. /** Abstract method findArea */
44. public abstract double getArea();
45.
46. /** Abstract method getPerimeter */
47. public abstract double getPerimeter();
48. }
49.

```

Circle Class:

```

1. package edu.northeastern.csye6200;
2.
3. public class Circle extends GeometricObject{
4.     private double radius;
5.
6.     /**Default constructor*/
7.     public Circle() {
8.         this(1.0);
9.     }
10.
11.     /**Construct circle with a specified radius*/
12.     public Circle(double radius) {
13.         this(radius, "white", false);
14.     }
15.
16.     /**Construct a circle with specified radius, filled, and color*/
17.     public Circle(double radius, String color, boolean filled) {
18.         super(color, filled);
19.         this.radius = radius;
20.     }
21.
22.     /**Return radius*/
23.     public double getRadius() {
24.         return radius;
25.     }
26.
27.     /**Set a new radius*/
28.     public void setRadius(double radius) {
29.         this.radius = radius;
30.     }
31.
32.     /**Implement the getArea method defined in GeometricObject*/
33.     public double getArea() {
34.         return radius*radius*Math.PI;
35.     }
36.
37.     /**Implement the getPerimeter method defined in GeometricObject*/
38.     public double getPerimeter() {
39.         return 2*radius*Math.PI;
40.     }
41.
42.     /**Override the equals() method defined in the Object class*/
43.     public boolean equals(Circle circle) {
44.         return this.radius == circle.getRadius();

```

```

45.     }
46.
47.     @Override
48.     public String toString() {
49.         return "[Circle] radius = " + radius;
50.     }
51. }
52.

```

## Screenshot:

The screenshot shows an IDE with several tabs: Lab9P2.java, Circle.java, Rectangle.java, GeometricObject.java, and Lab9P1.java. The Lab9P2.java tab is active, displaying the following code:

```

1 package edu.northeastern.csye6200;
2
3 public class Lab9P2 {
4     public static void main(String[] args) {
5         GeometricObject[] geometricObjects = new GeometricObject[5];
6         geometricObjects[0] = new Square(2);
7         geometricObjects[1] = new Circle(5);
8         geometricObjects[2] = new Square(5);
9         geometricObjects[3] = new Rectangle(3, 4);
10        geometricObjects[4] = new Square(4.5);
11
12        for (int i = 0; i < geometricObjects.length; i++) {
13            System.out.printf("Area is %.2f \n", geometricObjects[i].getArea());
14            if (geometricObjects[i] instanceof Colorable) {
15                System.out.println(((Colorable)geometricObjects[i]).getColor());
16            }
17            System.out.printf("\n");
18        }
19    }
20 }

```

Below the code editor, the Console window shows the output of the program:

```

<terminated> Lab9P2 [Java Application] /Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java (Nov 19, 2022, 7:44:10 AM)
Area is 4.00
Color all four sides

Area is 78.54

Area is 25.00
Color all four sides

Area is 12.00

Area is 20.25
Color all four sides

```