

Assignment 5 Lab 9

Author: Aashay Pawar

NUID: 002134382

Problem 1

Write a class named Octagon that extends GeometricObject and implements the Comparable and Cloneable interfaces. Assume that all eight sides of the octagon are of equal length. The area can be computed using the following formula:

$$area = (2 + 4/\sqrt{2}) * side * side$$

Write a test program that creates an Octagon object with side value 9 and displays its area and perimeter. Create a new object using the clone() method and compare the two objects using the compareTo() method.

Expected results:

Area is 391.10 Perimeter is 72.0 Compare the methods 0
--

Problem Description: The problem involves implementing a class called Octagon that extends the GeometricObject class and implements the Comparable and Cloneable interfaces. The class represents an octagon with all eight sides of equal length. The Octagon class contains methods to calculate the area and perimeter of the octagon using specific formulas.

The GeometricObject class is an abstract class that serves as a base for geometric objects such as circles, rectangles, and squares. It provides common properties like color and filled status, as well as abstract methods to calculate area and perimeter. The Circle and Rectangle classes are also provided as examples of how to extend the GeometricObject class.

A test program (Lab9P1) is included to create an Octagon object with a side value of 9, display its area and perimeter, create a new object using the clone() method, and compare the two objects using the compareTo() method. The expected result is to display the area as 391.10, perimeter as 72.0, and the comparison result as 0.

Analysis:

1. Define the GeometricObject abstract class with properties for color and filled status, and abstract methods to calculate area and perimeter.
2. Create the Circle and Rectangle classes that extend the GeometricObject class and implement their specific getArea() and getPerimeter() methods.
3. Define the Colorable interface with a method howToColor() to represent objects that can be colored.
4. Implement the Square class that extends GeometricObject and implements the Colorable interface.
5. Create the Octagon class that extends GeometricObject, implements Comparable<Octagon>, and overrides getArea(), getPerimeter(), compareTo(), and clone() methods.

6. In the Lab9P1 test program, create an Octagon object with a side value of 9 and display its area and perimeter.
7. Use the clone() method to create a copy of the Octagon object and compare the two objects using the compareTo() method, displaying the comparison result.

Code:

```
// Author: Aashay Pawar
// NUID: 002134382
// Date: 2023-07-31
// Description: Problem 1

package edu.northeastern.csye6200;

public class Lab9P1 {
    public static void main(String[] args) {
        // Create an Octagon object with side value 9
        Octagon octagon = new Octagon(9);

        // Display the area and perimeter of the Octagon
        System.out.println("Area is " + octagon.getArea());
        System.out.println("Perimeter is " + octagon.getPerimeter());

        // Create a new object using the clone() method
        Octagon clonedOctagon = (Octagon) octagon.clone();

        // Compare the two objects using the compareTo() method
        int comparisonResult = octagon.compareTo(clonedOctagon);
        System.out.println("Compare the methods " + comparisonResult);
    }
}

class Octagon extends GeometricObject implements Comparable<Octagon>, Cloneable {
    private double side;

    // Constructor
    public Octagon(double side) {
        this.side = side;
    }

    // Getter and setter for side
    public double getSide() {
        return side;
    }

    public void setSide(double side) {
        this.side = side;
    }

    // Calculate and override the getArea() method
    @Override
```

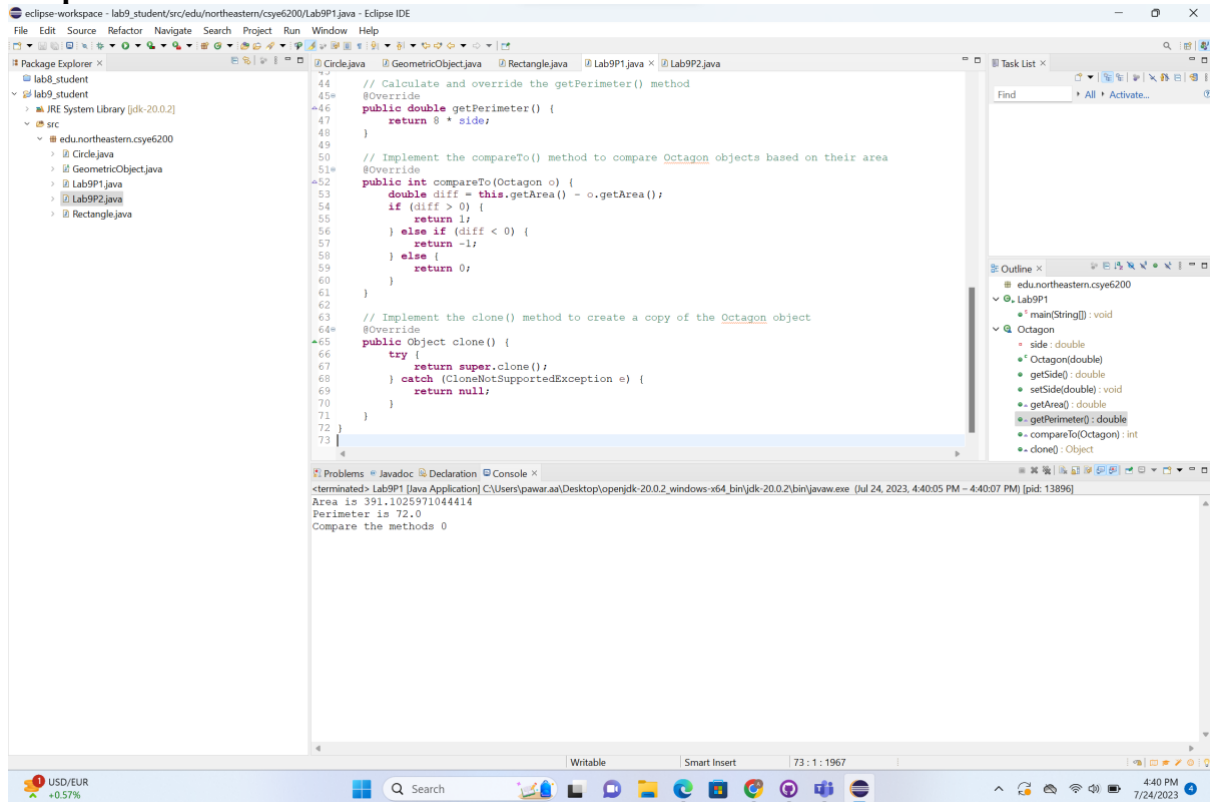
```
public double getArea() {
    return (2 + 4 / Math.sqrt(2)) * side * side;
}

// Calculate and override the getPerimeter() method
@Override
public double getPerimeter() {
    return 8 * side;
}

// Implement the compareTo() method to compare Octagon objects based on their area
@Override
public int compareTo(Octagon o) {
    double diff = this.getArea() - o.getArea();
    if (diff > 0) {
        return 1;
    } else if (diff < 0) {
        return -1;
    } else {
        return 0;
    }
}

// Implement the clone() method to create a copy of the Octagon object
@Override
public Object clone() {
    try {
        return super.clone();
    } catch (CloneNotSupportedException e) {
        return null;
    }
}
}
```

Output:



```
eclipse-workspace - lab9_student/src/edu/northeastern/csy6200/Lab9P1.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
lab9_student
  JRE System Library [jdk-20.0.2]
  src
    edu.northeastern.csy6200
      Circle.java
      GeometricObject.java
      Lab9P1.java
      Lab9P2.java
      Rectangle.java
Circle.java
// Calculate and override the getPerimeter() method
44
45 @Override
46 public double getPerimeter() {
47     return 8 * side;
48 }
49
50 // Implement the compareTo() method to compare Octagon objects based on their area
51 @Override
52 public int compareTo(Octagon o) {
53     double diff = this.getArea() - o.getArea();
54     if (diff > 0) {
55         return 1;
56     } else if (diff < 0) {
57         return -1;
58     } else {
59         return 0;
60     }
61 }
62
63 // Implement the clone() method to create a copy of the Octagon object
64 @Override
65 public Object clone() {
66     try {
67         return super.clone();
68     } catch (CloneNotSupportedException e) {
69         return null;
70     }
71 }
72 }
73
Problems Javadoc Declaration Console
<terminated> Lab9P1 [Java Application] C:\Users\pawar.aa\Desktop\openjdk-20.0.2_windows-x64_bin\jdk-20.0.2\bin\javaw.exe (Jul 24, 2023, 4:40:05 PM - 4:40:07 PM) [pid: 13896]
Area is 391.1025971044414
Perimeter is 72.0
Compare the methods 0
```

Problem 2

Design an interface named `Colorable` with a void method named `howToColor()`. Every class of a colorable object must implement the `Colorable` interface. Design a class named `Square` that extends `GeometricObject` and implements `Colorable`. Implement `howToColor()` to display the message `Color all four sides`. Write a test program that creates an array of five `GeometricObjects`. (`new Square(2)`, `new Circle(5)`, `new Square(5)`, `new Rectangle(3, 4)`, `new Square(4.5)`) For each object in the array, display its area and invoke its `howToColor()` method if it is colorable.

Problem Description: The problem involves designing an interface named `Colorable` with a method `howToColor()`. Every class of a colorable object must implement the `Colorable` interface. Additionally, a class named `Square` needs to be designed, which extends `GeometricObject` and implements the `Colorable` interface. The `howToColor()` method in the `Square` class should display the message `"Color all four sides"`.

A test program is required to create an array of five `GeometricObjects` (instances of `Square`, `Circle`, `Rectangle`). The program will then display the area of each object and invoke the `howToColor()` method for objects that are colorable (implement the `Colorable` interface).

Analysis:

1. Design the Colorable interface with a method howToColor().
2. Create the Square class that extends GeometricObject and implements the Colorable interface.
3. Implement the howToColor() method in the Square class to display the message "Color all four sides".
4. Write a test program (Lab9P2) that creates an array of five GeometricObjects.
5. Display the area of each object in the array.
6. If an object in the array is colorable (implements Colorable), invoke its howToColor() method.

Code:

```
// Author: Aashay Pawar
// NUID: 002134382
// Date: 2023-07-31
// Description: Problem 2

package edu.northeastern.csye6200;

public class Lab9P2 {
    public static void main(String[] args) {
        GeometricObject[] objects = {
            new Square(2),
            new Circle(5),
            new Square(5),
            new Rectangle(3, 4),
            new Square(4.5)
        };

        for (GeometricObject obj : objects) {
            System.out.println("Area is " + obj.getArea());
            if (obj instanceof Colorable) {
                ((Colorable) obj).howToColor();
            }
        }
    }
}

interface Colorable {
    void howToColor();
}

class Square extends GeometricObject implements Colorable {
    private double side;

    // Constructor
    public Square(double side) {
        this.side = side;
    }
}
```

```
// Getter and setter for side
public double getSide() {
    return side;
}

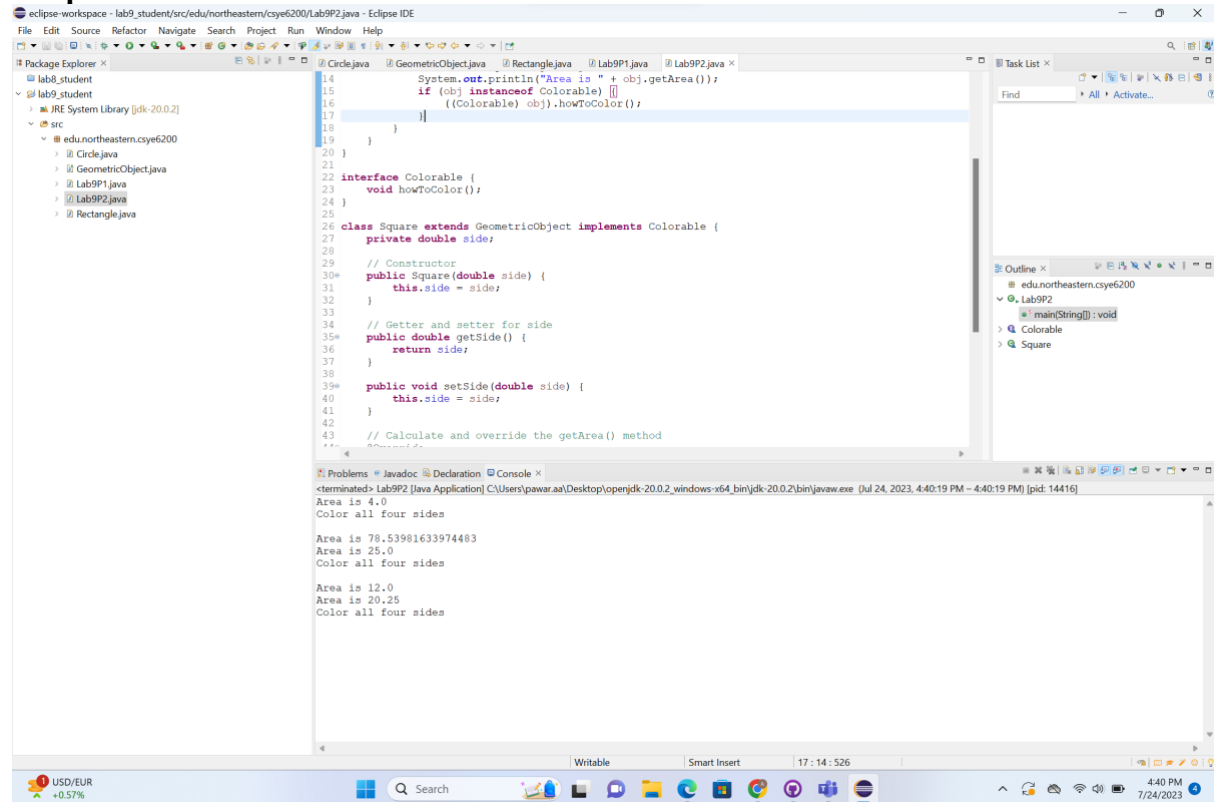
public void setSide(double side) {
    this.side = side;
}

// Calculate and override the getArea() method
@Override
public double getArea() {
    return side * side;
}

// Calculate and override the getPerimeter() method
@Override
public double getPerimeter() {
    return 4 * side;
}

// Implement the howToColor() method from the Colorable interface
@Override
public void howToColor() {
    System.out.println("Color all four sides\n");
}
}
```

Output:



The screenshot shows the Eclipse IDE with the following components:

- Package Explorer:** Shows the project structure with 'edu.northeastern.csye6200' containing 'Circle.java', 'GeometricObject.java', 'Lab9P1.java', 'Lab9P2.java', and 'Rectangle.java'.
- Editor:** Displays the code for 'Circle.java'. The code includes a 'main' method that creates a 'Circle' object and prints its area and color. The 'Circle' class is defined as a subclass of 'GeometricObject' with a 'radius' attribute and methods for getting and setting the radius, and a 'getArea()' method.
- Console:** Shows the output of the program, which includes the area and color of the circle for different inputs.

```
14 System.out.println("Area is " + obj.getArea());
15 if (obj instanceof Colorable) {
16     ((Colorable) obj).howToColor();
17 }
18 }
19 }
20 }
21
22 interface Colorable {
23     void howToColor();
24 }
25
26 class Square extends GeometricObject implements Colorable {
27     private double side;
28
29     // Constructor
30     public Square(double side) {
31         this.side = side;
32     }
33
34     // Getter and setter for side
35     public double getSide() {
36         return side;
37     }
38
39     public void setSide(double side) {
40         this.side = side;
41     }
42
43     // Calculate and override the getArea() method
44 }
```

```
<terminated> Lab9P2 [Java Application] C:\Users\pawar.aa\Desktop\openjdk-20.0.2_windows-x64_bin\jdk-20.0.2\bin\javaw.exe (Jul 24, 2023, 4:40:19 PM) [pid: 14416]
Area is 4.0
Color all four sides
Area is 78.53981633974483
Area is 25.0
Color all four sides
Area is 12.0
Area is 20.25
Color all four sides
```

Reference Codes:

Circle.java

```
package edu.northeastern.csye6200;

public class Circle extends GeometricObject {
    private double radius;

    /**Default constructor*/
    public Circle() {
        this(1.0);
    }

    /**Construct circle with a specified radius*/
    public Circle(double radius) {
        this(radius, "white", false);
    }

    /**Construct a circle with specified radius, filled, and color*/
    public Circle(double radius, String color, boolean filled) {
        super(color, filled);
        this.radius = radius;
    }
}
```

```

/**Return radius*/
public double getRadius() {
    return radius;
}

/**Set a new radius*/
public void setRadius(double radius) {
    this.radius = radius;
}

/**Implement the getArea method defined in GeometricObject*/
public double getArea() {
    return radius*radius*Math.PI;
}

/**Implement the getPerimeter method defined in GeometricObject*/
public double getPerimeter() {
    return 2*radius*Math.PI;
}

/**Override the equals() method defined in the Object class*/
public boolean equals(Circle circle) {
    return this.radius == circle.getRadius();
}

@Override
public String toString() {
    return "[Circle] radius = " + radius;
}
}

```

GeometricObject.java

```

package edu.northeastern.csye6200;

// GeometricObject.java: The abstract GeometricObject class

public abstract class GeometricObject {

    private String color = "white";
    private boolean filled;

    /** Default construct */
    protected GeometricObject() {
    }

    /** Construct a geometric object */
    protected GeometricObject(String color, boolean filled) {
        this.color = color;
        this.filled = filled;
    }
}

```



```

    /** Getter method for color */
    public String getColor() {
        return color;
    }

    /** Setter method for color */
    public void setColor(String color) {
        this.color = color;
    }

    /**
     * Getter method for filled. Since filled is boolean, so, the get method
     * name is isFilled
     */
    public boolean isFilled() {
        return filled;
    }

    /** Setter method for filled */
    public void setFilled(boolean filled) {
        this.filled = filled;
    }

    /** Abstract method findArea */
    public abstract double getArea();

    /** Abstract method getPerimeter */
    public abstract double getPerimeter();
}

```

Rectangle.java

```

package edu.northeastern.csye6200;

public class Rectangle extends GeometricObject {

    private double width;
    private double height;

    // Constructor
    public Rectangle(double width, double height) {
        this.width = width;
        this.height = height;
    }

    // Getter and setter for width and height
    public double getWidth() {
        return width;
    }
}

```

```

public void setWidth(double width) {
    this.width = width;
}

public double getHeight() {
    return height;
}

public void setHeight(double height) {
    this.height = height;
}

// Calculate and override the getArea() method
@Override
public double getArea() {
    return width * height;
}

// Calculate and override the getPerimeter() method
@Override
public double getPerimeter() {
    return 2 * (width + height);
}
}

```

Lab9P1.java

```

package edu.northeastern.csye6200;

public class Lab9P1 {
    public static void main(String[] args) {
        // Create an Octagon object with side value 9
        Octagon octagon = new Octagon(9);

        // Display the area and perimeter of the Octagon
        System.out.println("Area is " + octagon.getArea());
        System.out.println("Perimeter is " + octagon.getPerimeter());

        // Create a new object using the clone() method
        Octagon clonedOctagon = (Octagon) octagon.clone();

        // Compare the two objects using the compareTo() method
        int comparisonResult = octagon.compareTo(clonedOctagon);
        System.out.println("Compare the methods " + comparisonResult);
    }
}

class Octagon extends GeometricObject implements Comparable<Octagon>, Cloneable {
    private double side;

    // Constructor

```

```

public Octagon(double side) {
    this.side = side;
}

// Getter and setter for side
public double getSide() {
    return side;
}

public void setSide(double side) {
    this.side = side;
}

// Calculate and override the getArea() method
@Override
public double getArea() {
    return (2 + 4 / Math.sqrt(2)) * side * side;
}

// Calculate and override the getPerimeter() method
@Override
public double getPerimeter() {
    return 8 * side;
}

// Implement the compareTo() method to compare Octagon objects based on their area
@Override
public int compareTo(Octagon o) {
    double diff = this.getArea() - o.getArea();
    if (diff > 0) {
        return 1;
    } else if (diff < 0) {
        return -1;
    } else {
        return 0;
    }
}

// Implement the clone() method to create a copy of the Octagon object
@Override
public Object clone() {
    try {
        return super.clone();
    } catch (CloneNotSupportedException e) {
        return null;
    }
}
}

```

Lab9P2.java

```

package edu.northeastern.csye6200;

public class Lab9P2 {
    public static void main(String[] args) {
        GeometricObject[] objects = {
            new Square(2),
            new Circle(5),
            new Square(5),
            new Rectangle(3, 4),
            new Square(4.5)
        };

        for (GeometricObject obj : objects) {
            System.out.println("Area is " + obj.getArea());
            if (obj instanceof Colorable) {
                ((Colorable) obj).howToColor();
            }
        }
    }
}

interface Colorable {
    void howToColor();
}

class Square extends GeometricObject implements Colorable {
    private double side;

    // Constructor
    public Square(double side) {
        this.side = side;
    }

    // Getter and setter for side
    public double getSide() {
        return side;
    }

    public void setSide(double side) {
        this.side = side;
    }

    // Calculate and override the getArea() method
    @Override
    public double getArea() {
        return side * side;
    }

    // Calculate and override the getPerimeter() method
    @Override

```

```
public double getPerimeter() {  
    return 4 * side;  
}  
  
// Implement the howToColor() method from the Colorable interface  
@Override  
public void howToColor() {  
    System.out.println("Color all four sides\n");  
}  
}
```