Programming and Data Structures
Active Learning Activity 6: Polymorphism, abstract classes, and interfaces

Activity Objectives

At the end of this activity, students should be able to:

1. Create an abstract class that implements the interfaces Comparable and Cloneable
2. Derive concrete classes from the abstract class
3. Use polymorphism to manipulate objects of the concrete classes as objects of the abstract class type

Activity

Create the classes and the relationships shown in the UML diagram provided on the last page. Note that the interfaces **`Cloneable`** and **`Comparable`** already exist in the Java API. You do not need to create them. Class **`Shape`** is an abstract class with two abstract methods **`getArea()`** and **`getPerimeter()`**. The implementation of the function **`int compareTo(Shape)`** from the interface **`Comparable`** should be included in the class **`Shape`** and have the following behavior.

1. **`compareTo(Shape)`** returns 0 if the areas of the two **`Shape`** objects are equal

2. **`compareTo(Shape)`** returns -1 if the area of the first **`Shape`** object is less than the area of the second **`Shape`** object

3. **`compareTo(Shape)`** returns 1 if the area of the first **`Shape`** object is greater than the area of the second **`Shape`** object

The method **`compareTo()`** will call **`getArea()`** to compare the areas even if **`getArea()`** is not defined in class **`Shape`**. Remember that **dynamic binding** allows calling, at runtime, the right method from the concrete classes.

Write a test program **`TestShapes`** to do the following:

1. Create an array **`shapeObjects`** to hold 8 objects of type **`Shape`**.
2. Create one object of each of the concrete classes (**`Circle`**, **`Rectangle`**, **`Triangle`**, **`Octagon`**) and save the reference of each object in the array **`shapeObjects`**. The objects should have the following attributes.

    a. **`Circle`** object has **`color`** "**Black**" and **`radius 2.5`**

    b. **`Rectangle`** object has color **"Red"**, **`length 5`**, and **`width 3`**

    c. **`Triangle`** object has color **"Green"** and the three sides equal to **6**, **6**, and **8**. The area of the triangle is calculated using the formula: $\sqrt{p.(p-side1).(p-side2).(p-side3)}$ where p is half the perimeter ($p = \frac{side1+side2+side3}{2}$)

    d. **`Octagon`** object has color **"Yellow"** and **`side 7`**. The area of the octagon is calculated using the formula: $\left(2+4\sqrt{2}\right).side^2$

3. Clone each of the created objects (using the method **`Object clone()`**) and save the cloned objects in the array **`shapeObjects`**.
4. Modify the **`radius`** of the cloned **`Circle`** object to 1.5, the **`side`** of the cloned **`Octagon`** object to 2.5, and the **`length`** of the cloned **`Rectangle`** object to 20.
5. Display the information of the 8 objects stored in **`shapeObjects`**.
6. Call the method **`java.util.Arrays.sort()`** and pass the array **`shapeObjects`** as the argument. The method **`sort()`** will sort the objects based on their areas using the definition of the method **`compareTo()`** in class **`Shape`**.
7. Display the information of the 8 objects in **`shapeObjects`** after the sorting.
8. Create a static method **`double getAverageOfPerimeters(Shape[] list`**) that returns the average of the perimeters of the **`Shape`** objects in **`list`**. Call the method from

the **`main`** method with the array **`shapeObjects`** as the argument. Display the return value of **`getAverageOfPerimeters()`**.

9. The expected output of the program should be similar to the listing below.
10. Submit the files:

    `Shape.java`

    `Circle.java`

    `Rectangle.java`

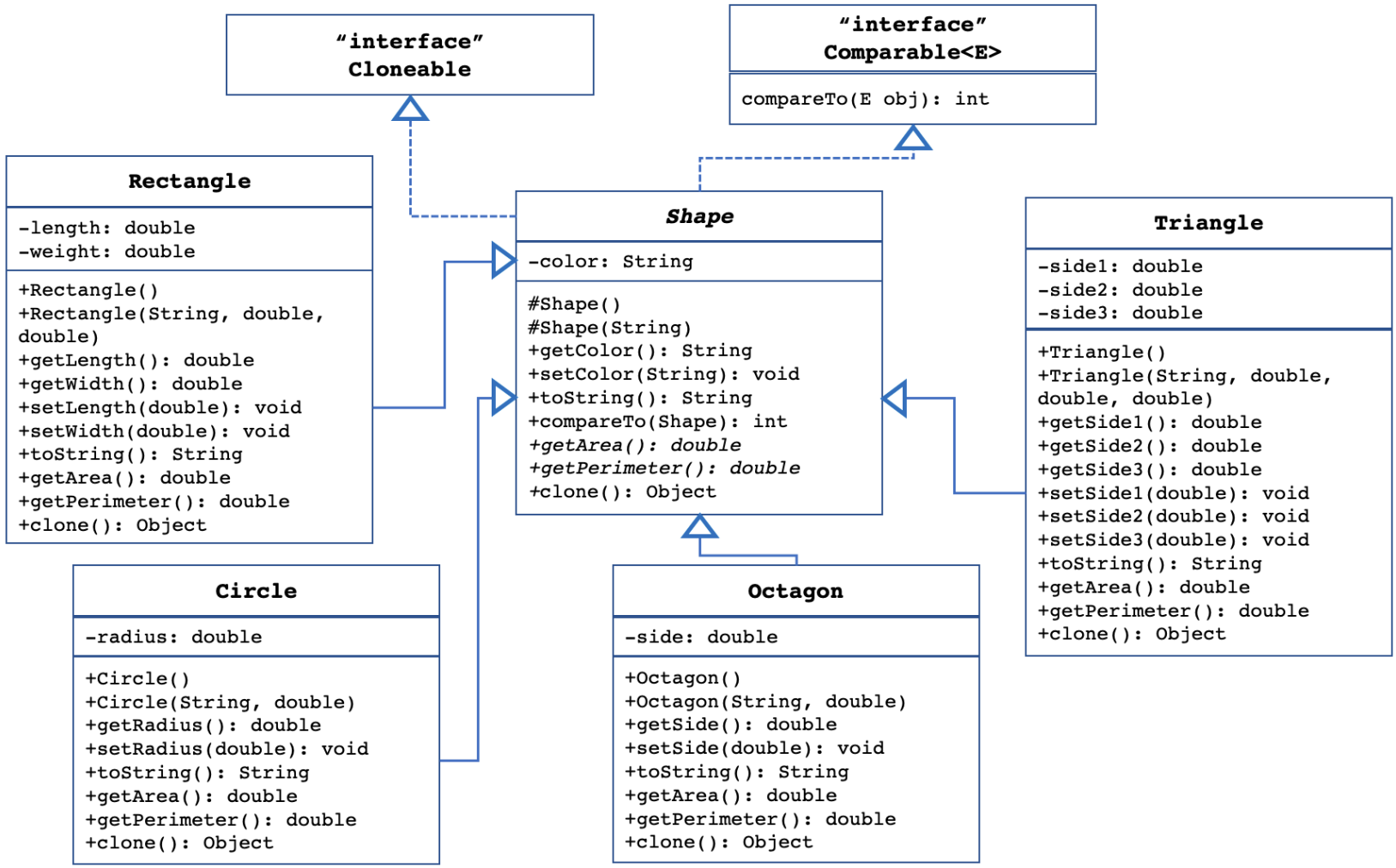
    `Triangle.java`

    `Octagon.java`, and

    `TestShapes.java` on courseSite and get your program checked off by a TA or me.

```
---------------------- SAMPLE OUTPUT -------------------------

BEFORE SORT
-----------

Object 1: Circle
 Color: Black, Radius = 2.5
 Area = 19.63, Perimeter = 15.71

Object 2: Rectangle
 Color: Red, Length = 5.0, Width = 3.0
 Area = 15.00, Perimeter = 16.00

Object 3: Triangle
 Color: Green, Side 1 = 6.0, Side 2 = 6.0, Side 3 = 8.0
 Area = 17.89, Perimeter = 20.00

Object 4: Octagon
 Color: Yellow, Side = 7.0
 Area = 236.59, Perimeter = 56.00

Object 5: Circle
 Color: Black, Radius = 1.5
 Area =  7.07, Perimeter =  9.42

Object 6: Rectangle
 Color: Red, Length = 20.0, Width = 3.0
 Area = 60.00, Perimeter = 46.00

Object 7: Triangle
 Color: Green, Side 1 = 6.0, Side 2 = 6.0, Side 3 = 8.0
```

```
 Area = 17.89, Perimeter = 20.00

Object 8: Octagon
 Color: Yellow, Side = 2.5
 Area = 30.18, Perimeter = 20.00

AFTER SORT
-----------

Object 1: Circle
 Color: Black, Radius = 1.5
 Area =  7.07, Perimeter =  9.42

Object 2: Rectangle
 Color: Red, Length = 5.0, Width = 3.0
 Area = 15.00, Perimeter = 16.00

Object 3: Triangle
 Color: Green, Side 1 = 6.0, Side 2 = 6.0, Side 3 = 8.0
 Area = 17.89, Perimeter = 20.00

Object 4: Triangle
 Color: Green, Side 1 = 6.0, Side 2 = 6.0, Side 3 = 8.0
 Area = 17.89, Perimeter = 20.00

Object 5: Circle
 Color: Black, Radius = 2.5
 Area = 19.63, Perimeter = 15.71

Object 6: Octagon
 Color: Yellow, Side = 2.5
 Area = 30.18, Perimeter = 20.00

Object 7: Rectangle
 Color: Red, Length = 20.0, Width = 3.0
 Area = 60.00, Perimeter = 46.00

Object 8: Octagon
 Color: Yellow, Side = 7.0
 Area = 236.59, Perimeter = 56.00


The average perimeter of all the objects is 25.39
```


"interface"
Cloneable
"interface"
Comparable<E>
compareTo(E obj): int
Rectangle
-length: double
-weight: double
+Rectangle()
+Rectangle(String, double, double)
+getLength(): double
+getWidth(): double
+setLength(double): void
+setWidth(double): void
+toString(): String
+getArea(): double
+getPerimeter(): double
+clone(): Object
Shape
-color: String
#Shape()
#Shape(String)
+getColor(): String
+setColor(String): void
+toString(): String
+compareTo(Shape): int
+getArea(): double
+getPerimeter(): double
+clone(): Object
Triangle
-side1: double
-side2: double
-side3: double
+Triangle()
+Triangle(String, double, double, double)
+getSide1(): double
+getSide2(): double
+getSide3(): double
+setSide1(double): void
+setSide2(double): void
+setSide3(double): void
+toString(): String
+getArea(): double
+getPerimeter(): double
+clone(): Object
Circle
-radius: double
+Circle()
+Circle(String, double)
+getRadius(): double
+setRadius(double): void
+toString(): String
+getArea(): double
+getPerimeter(): double
+clone(): Object
Octagon
-side: double
+Octagon()
+Octagon(String, double)
+getSide(): double
+setSide(double): void
+toString(): String
+getArea(): double
+getPerimeter(): double
+clone(): Object