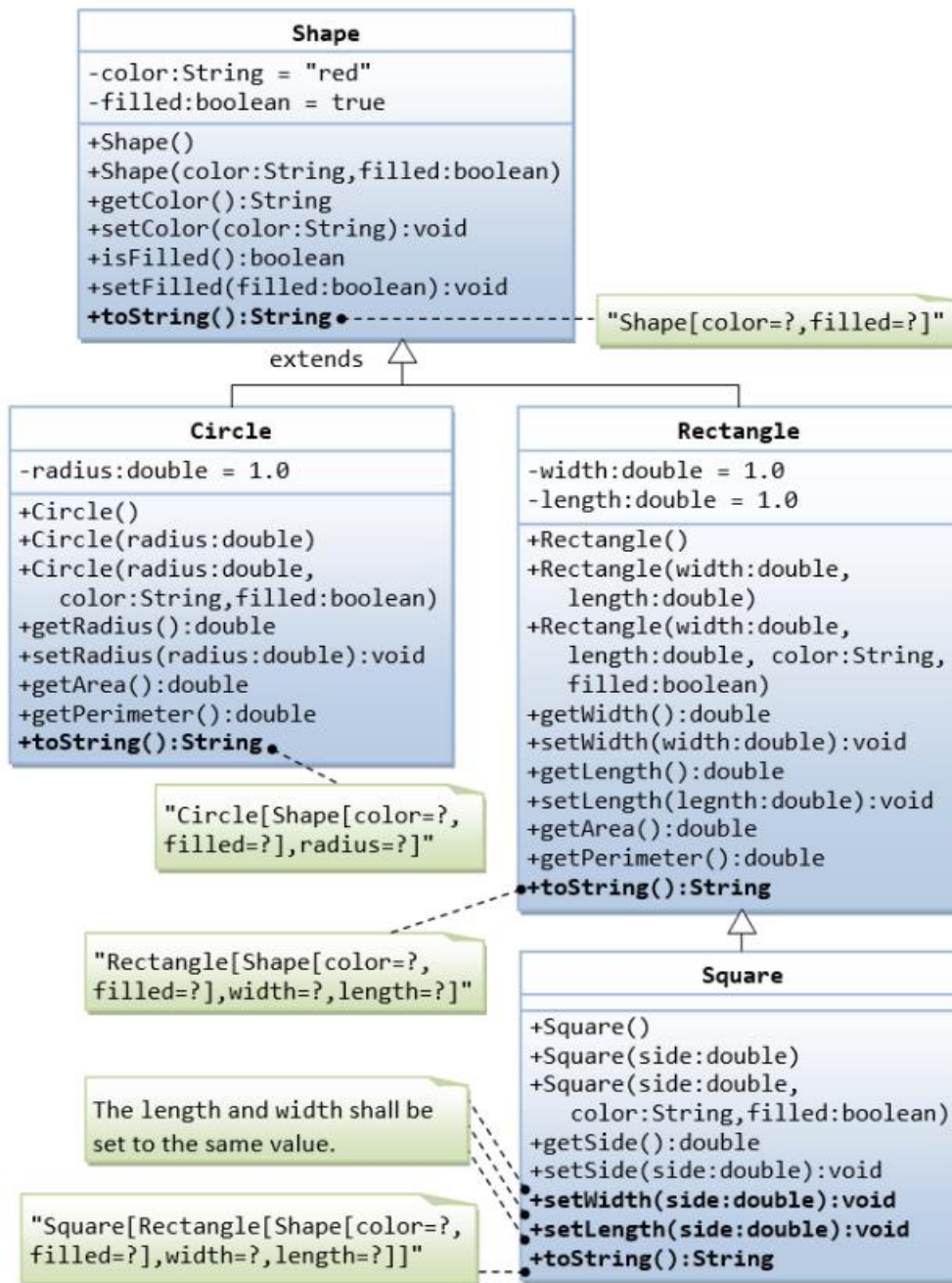


Ex 01: Write the classes as shown in the following class diagram . Mark all the overridden methods with annotation `@Override`.



Ex 02: Write the classes as shown in the following class diagram . Mark all the overridden methods with annotation `@Override`.



1.1 Write a superclass called Shape (as shown in the class diagram), which contains:

- Two instance variables `color` (String) and `filled` (boolean).
- Two constructors: a no-arg (no-argument) constructor that initializes the `color` to "green" and `filled` to true, and a constructor that initializes the `color` and `filled` to the given values.

- Getter and setter for all the instance variables. By convention, the getter for a boolean variable xxx is called isXXX() (instead of getXxx() for all the other types).
- A toString() method that returns "A Shape with color of xxx and filled/Not filled".

1.2 Write two subclasses of Shape called Circle and Rectangle, as shown in the class diagram.

The Circle class contains:

- An instance variable radius (double).
- Three constructors as shown. The no-arg constructor initializes the radius to 1.0.
- Getter and setter for the instance variable radius.
- Methods getArea() and getPerimeter().
- Override the toString() method inherited, to return "A Circle with radius=xxx, which is a subclass of yyy", where yyy is the output of the toString() method from the superclass.

The Rectangle class contains:

- Two instance variables width (double) and length (double).
- Three constructors as shown. The no-arg constructor initializes the width and length to 1.0.
- Getter and setter for all the instance variables.
- Methods getArea() and getPerimeter().
- Override the toString() method inherited, to return "A Rectangle with width=xxx and length=zzz, which is a subclass of yyy", where yyy is the output of the toString() method from the superclass.

1.3 Write a class called Square, as a subclass of Rectangle. The Square can be modeled as a subclass of Rectangle. Square has no instance variable, but inherits the instance variables width and length from its superclass Rectangle.

- Provide the appropriate constructors (as shown in the class diagram). Hint:

```
public Square(double side) {
    super(side, side); // Call superclass Rectangle(double, double)
}
```

- Override the toString() method to return "A Square with side=xxx, which is a subclass of yyy", where yyy is the output of the toString() method from the superclass.
- Override the setLength() and setWidth() to change both the width and length, so as to maintain the square geometry.