Implement the following class diagram:

```
Shape
     -color:String = "red"
     -filled:boolean = true
     +Shape()
     +Shape(color:String,filled:boolean)
     +getColor():String
     +setColor(color:String):void
     +isFilled():boolean
     +setFilled(filled:boolean):void
     +toString():String.
                                             "Shape[color=?,filled=?]"
                   extends
            Circle
                                                  Rectangle
-radius:double = 1.0
                                       -width:double = 1.0
                                      -length:double = 1.0
+Circle()
+Circle(radius:double)
                                      +Rectangle()
+Circle(radius:double,
                                      +Rectangle(width:double,
  color:String,filled:boolean)
                                         length:double)
+getRadius():double
                                      +Rectangle(width:double,
+setRadius(radius:double):void
                                         length:double, color:String,
+getArea():double
                                         filled:boolean)
+getPerimeter():double
                                      +getWidth():double
+toString():String。
                                      +setWidth(width:double):void
                                      +getLength():double
                                      +setLength(legnth:double):void
           "Circle[Shape[color=?,
                                      +getArea():double
          filled=?],radius=?]"
                                      +getPerimeter():double
                                      +toString():String
     "Rectangle[Shape[color=?,
                                                   Square
     filled=?],width=?,length=?]"
                                       +Square()
                                       +Square(side:double)
                                       +Square(side:double,
     The length and width shall be
                                          color:String,filled:boolean)
     set to the same value.
                                       +getSide():double
                                       +setSide(side:double):void
                                      +setWidth(side:double):void
"Square[Rectangle[Shape[color=?,
                                       +setLength(side:double):void
filled=?],width=?,length=?]]"
                                      +toString():String
```

## 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 is XXX() (instead of getXxx() for all the other types).
- A toString() method that returns "A Shape with color of xxx and filled/Not filled".
- 2) Write a test program to test all the methods defined in Shape.
- 3) 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.

## 4) Write a class called Square, as a subclass of Rectangle.

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