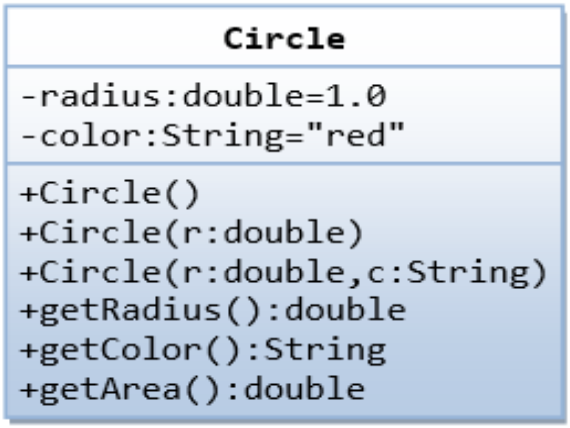
JAVA PROGRAMMING

Tutorial 03

Activity 1: Circle.java

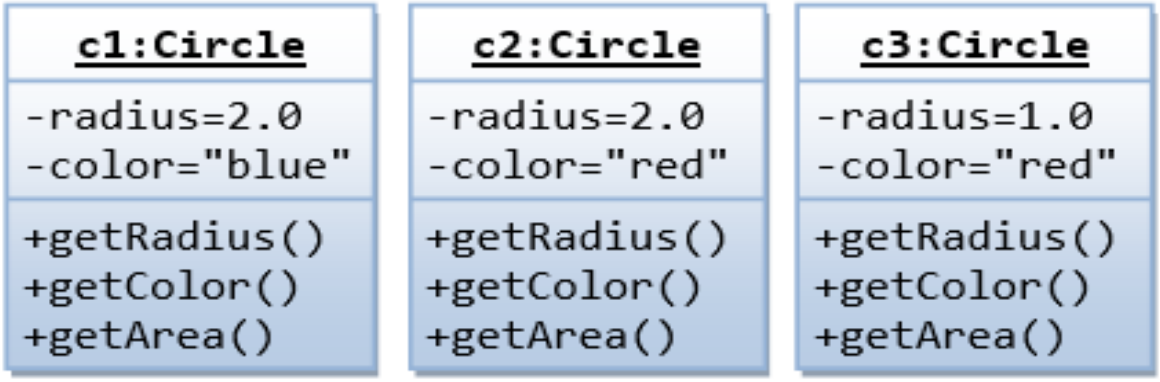
Implement a class called `Circle` defined as shown in the class diagram. It contains two private member variables: radius (double) and color (String); and three public member methods: **getRadius()**, **getColor()**, and **getArea()**.

Class Definition



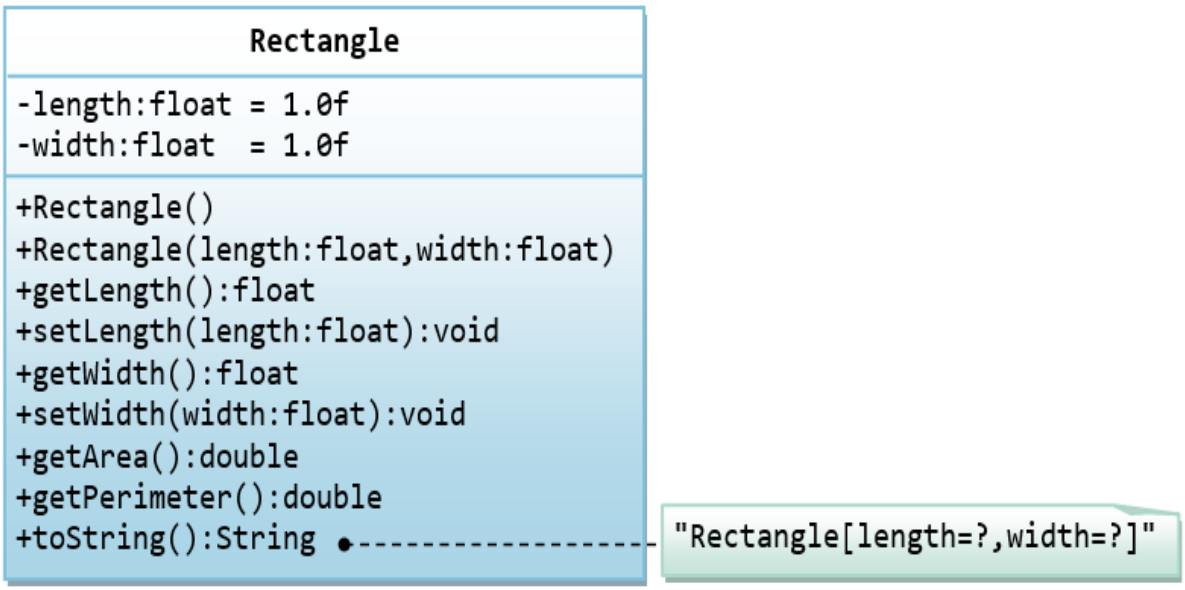
Implement three instances of `Circle`, called c1, c2, and c3, shall be constructed with their respective data members, as shown in the instance diagrams.

Instances



Activity 2: Rectangle.java

A class called **Rectangle**, which models a rectangle with a length and a width (in float), is designed as shown in the following class diagram.

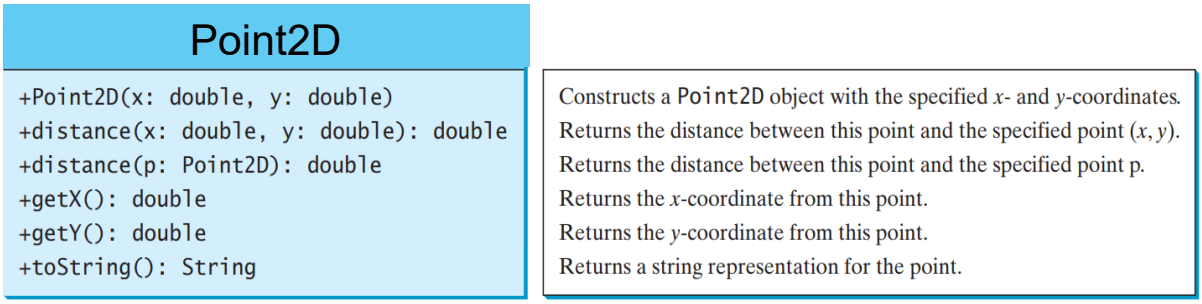


Implement the class **Rectangle**. An example using **RectangleTest.java** (provided in LMS):

```
Rectangle[width=1.2,height=3.4]
Rectangle[width=1.0,height=1.0]
Rectangle[width=7.8,height=5.6]
Length is: 5.60
Width is: 7.80
Area is: 43.68
Perimeter is: 26.80
```

Activity 3: Point2D.java

A class called `Point2D` represents a point in a two-dimensional plane. The UML diagram for the class is shown in the following figure.



Create a `Point2D` object for a point with the specified x- and y-coordinates, use the `distance` method to compute the distance from this point to another point, and use the `toString()` method to return a string representation of the point.

Sample run:

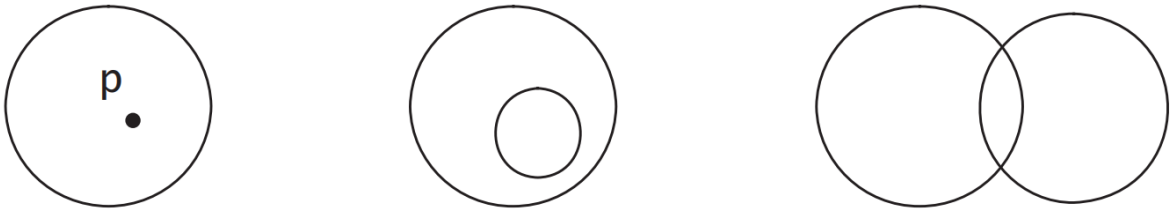
```
Enter point1's x-, y-coordinates: 1.5 5.5 Enter
Enter point2's x-, y-coordinates: -5.3 -4.4 Enter
p1 is Point2D [x = 1.5, y = 5.5]
p2 is Point2D [x = -5.3, y = -4.4]
The distance between p1 and p2 is 12.010412149464313
```

Activity 4: Circle2D (Upgraded)

Define the `Circle2D` class that contains:

- A `Point2D(x, y)` that specify the center of the circle with getter methods.
- A data field `radius` with a getter method.
- A no-arg constructor that creates a default circle with `Point2D`(`0`, `0`) and `1` for radius.

- A constructor that creates a circle with the specified `Point2D(x, y)`, and radius.
- A method **getArea()** that returns the area of the circle.
- A method **getPerimeter()** that returns the perimeter of the circle.
- A method **contains(Point2D point)** that returns true if the specified point is inside this circle (see Figure a).
- A method **contains(Circle2D circle)** that returns true if the specified circle is inside this circle (see Figure b).
- A method **overlaps(Circle2D circle)** that returns true if the specified circle overlaps with this circle (see Figure c).



(a) (b) (c)

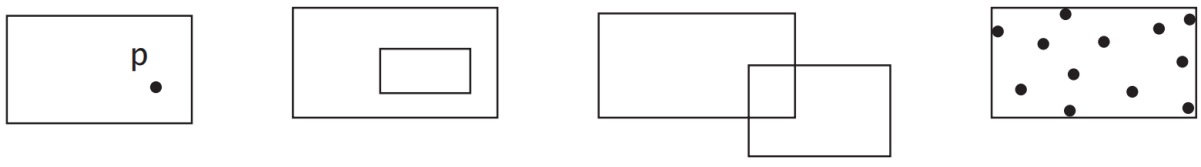
Draw the UML diagram for the class and then implement the class. Write a test program that creates a `Circle2D` object **c1 (new Circle2D(new Point2D(2, 2), 5.5))**, displays its area and perimeter, and displays the result of:

- `c1.contains(new Point2D(3,3))`
- `c1.contains(new Circle2D(4, 5, 10.5))`
- `c1.overlaps(new Circle2D(3, 5, 2.3))`

Activity 5: Rectangle2D (Upgraded)

Define the **MyRectangle2D** class that contains:

- A **Point2D(x, y)** that specify the center of the rectangle with getter and setter methods. (Assume that the rectangle sides are parallel to **x-** or **y-** axes.)
- The data fields **width** and **height** with getter and setter methods.
- A no-arg constructor that creates a default rectangle with **Point2D(0, 0)** and **1** for both **width** and **height**.
- A constructor that creates a rectangle with the specified **Point2D(x, y)**, **width**, and **height**.
- A method **getArea()** that returns the area of the rectangle.
- A method **getPerimeter()** that returns the perimeter of the rectangle.
- A method **contains(Point2D p)** that returns **true** if the specified point is inside this rectangle (see Figure a).
- A method **contains(MyRectangle2D r)** that returns **true** if the specified rectangle is inside this rectangle (see Figure b).
- A method **overlaps(MyRectangle2D r)** that returns **true** if the specified rectangle overlaps with this rectangle (see Figure c).



(a) (b) (c) (d)

Draw the UML diagram for the class and then implement the class. Write a test program that creates a **MyRectangle2D** object **r1** (**new MyRectangle2D(2, 2, 5.5, 4.9)**), displays its area and perimeter, and displays the result of:

- **r1.contains(new Point2D(3,3))**
- **r1.contains(new MyRectangle2D(4, 5, 10.5, 3.2))**
- **r1.overlaps(new MyRectangle2D(3, 5, 2.3, 5.4))**

Activity 6: LargeNumber.java (Upgraded)

Define the **LargeNumber** class that contains:

- A **String** data field **number** with a getter and a setter method.
- A no-arg constructor that creates a default large number with **null** for **number.**
- A constructor that creates a large number with the specified **number**. Note to validate the data before
- A method **isLargeNumber(String number)** that returns **true** if the specified **number** contains only numbers and has length bigger or equal to 15.
- A method **add(LargeNumber n)** that return the **sum** of two large numbers.
- A method **sub(LargeNumber n)** that return the **difference** of two large numbers.
- A method **times(LargeNumber n)** that return the **product** of two large numbers.
- A method **div(LargeNumber n)** that return the **division** of two large numbers.

Draw the UML diagram for the class and then implement the class. Write a test program that creates at least two instances.

Submission

Submit a zip file containing all Java programs to this tutorial's submission box in the course website on FIT Portal.