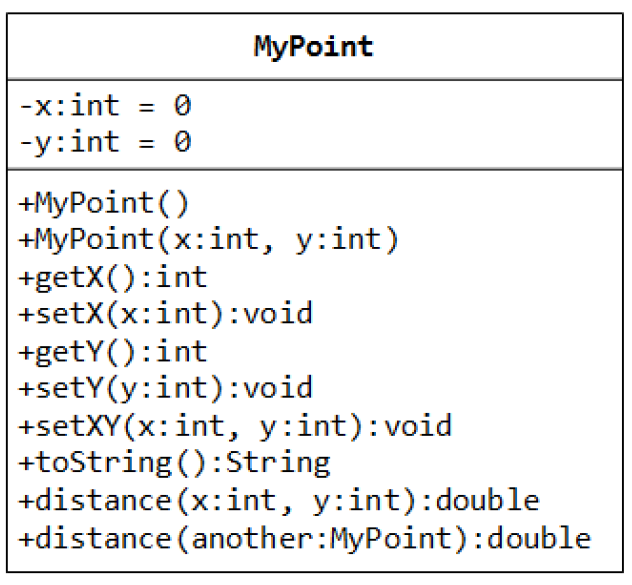
1.3 Exercise: The MyPoint Class

A class called MyPoint, which models a 2D point with x and y coordinates, is designed as shown in the class

diagram. It contains:

Two instance variables x ﴾int﴿ and y ﴾int﴿.

A "no‐argument" ﴾or "no‐arg"﴿ constructor that construct a point at (0, 0).

A constructor that constructs a point with the given

x and y coordinates.

Getter and setter for the instance variables x and y.

A method setXY() to set both x and y.

A toString() method that returns a string

description of the instance in the format "(*x*, *y*)".

A method called distance(int x, int y) that

returns the distance from *this* point to another point

at the given (x, y) coordinates.

An overloaded distance(MyPoint another) that

returns the distance from *this* point to the given

MyPoint instance another.

You are required to:

1. Write the code for the class MyPoint. Also write a

test program ﴾called TestMyPoint﴿ to test all the methods defined in the class.

Hints:

// Overloading method distance()

public double distance(int x, int y) { // this version takes two ints as arguments

int xDiff = this.x – x;

int yDiff = ......

return Math.sqrt(xDiff\*xDiff + yDiff\*yDiff);

}

public double distance(MyPoint another) { // this version takes a MyPoint instance as argument

int xDiff = this.x – another.x;

.......

}

// Test program

MyPoint p1 = new MyPoint(3, 0);

MyPoint p2 = new MyPoint(0, 4);

......

// Testing the overloaded method distance()

System.out.println(p1.distance(p2)); // which version?

System.out.println(p1.distance(5, 6)); // which version?

.....

2. Write a program that allocates 10 points in an array of MyPoint, and initializes to (1, 1), (2, 2), ... (10,

10).

Hints: You need to allocate the array, as well as each of the ten MyPoint instances.

MyPoint[] points = new MyPoint[10]; // Declare and allocate an array of MyPoint

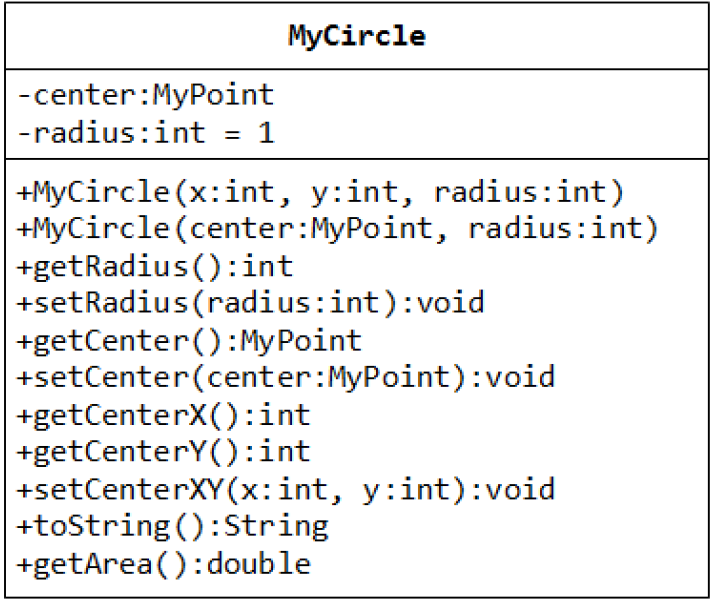
for (......) {

points[i] = new MyPoint(...); // Allocate each of MyPoint instances

}

Notes: Point is such a common entity that JDK certainly provided for in all flavors.

1.4 Exercise: The MyCircle Class

A class called MyCircle, which models a circle with a center ﴾x, y﴿ and a radius, is designed as shown in the class diagram. The MyCircle class uses an instance of MyPoint class ﴾created in the previous exercise﴿

as its center.

The class contains:

Two private instance variables: center ﴾an

instance of MyPoint﴿ and radius ﴾int﴿.

A constructor that constructs a circle with the

given center's ﴾x, y﴿ and radius.

An overloaded constructor that constructs a

MyCircle given a MyPoint instance as

center, and radius.

Various getters and setters.

A toString() method that returns a string

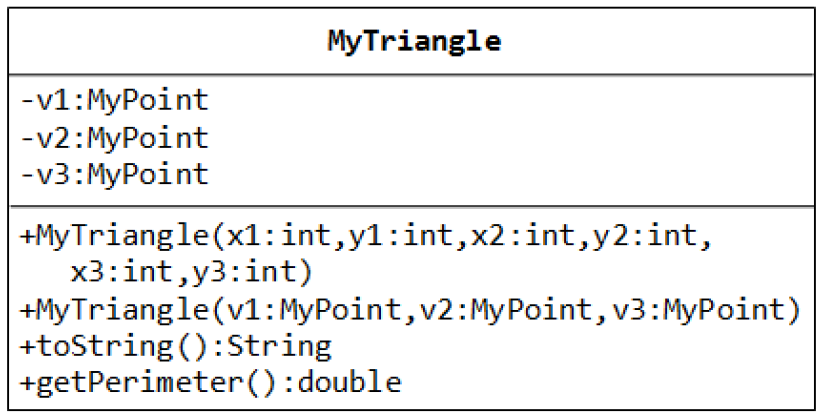
description of this instance in the format

"Circle @ ﴾*x*, *y*﴿ radius=*r*".

A getArea() method that returns the area of the circle in double.

Write the MyCircle class. Also write a test program ﴾called TestMyCircle﴿ to test all the methods defined in the

class.

1.5 Exercise: The MyTriangle Class

A class called MyTriangle, which

models a triangle with 3 vertices, is

designed as follows. The MyTriangle

class uses three MyPoint instances

﴾created in the earlier exercise﴿ as the

three vertices.

The class contains:

Three private instance variables v1,

v2, v3 ﴾instances of MyPoint﴿, for

the three vertices.

A constructor that constructs a

MyTriangle with three points v1=(x1, y1), v2=(x2, y2), v3=(x3, y3).

An overloaded constructor that constructs a MyTriangle given three instances of MyPoint.

A toString() method that returns a string description of the instance in the format "Triangle @ ﴾x1, y1﴿, ﴾x2, y2﴿, ﴾x3, y3﴿".

A getPerimeter() method that returns the length of the perimeter in double. You should use the

distance() method of MyPoint to compute the perimeter.

A method printType(), which prints "equilateral" if all the three sides are equal, "isosceles" if any two of the three sides are equal, or "scalene" if the three sides are different.

Write the MyTriangle class. Also write a test program ﴾called TestMyTriangle﴿ to test all the methods defined in the class.