Homework 2
(Due: March 7, 2022)

The purpose of this homework is to get you started with classes and objects in Java and Python. For each question hand in a solution (i.e., program source code + program output) in Java, and then a second solution (i.e., program source code + program output) coded in Python.

Question 1: 10 points

As shown in Figure 1 below, rectangles may be defined by the (x,y) coordinates of corner points that are diagonally opposite.

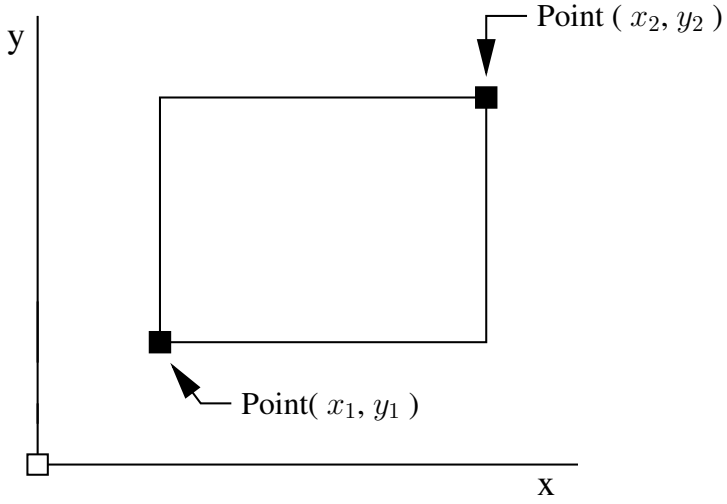

Figure 1: Definition of a rectangle via diagonally opposite corner points.

With this definition in place, the following script of code is a very basic implementation of a class for creating and working with rectangle objects.

```
/*
 *  ============================================================================
 *  Rectangle.java : A library of methods for creating and managing rectangles
 *
 *  double      area() -- returns the area of a rectangle
 *  double perimeter() -- returns the perimeter of a rectangle
```

```
 *
 *  Written By : Mark Austin                                    November 2019
 *  ============================================================================
 */

import java.lang.Math;

public class Rectangle {
   protected double dX1, dY1;  // Coordinate (x,y) for corner 1....
   protected double dX2, dY2;  // Coordinate (x,y) for corner 2....

   // Constructor methods ....

   public Rectangle() {}

   public Rectangle( double dX1, double dY1, double dX2, double dY2 ) {
      this.dX1     = dX1; this.dY1     = dY1;
      this.dX2     = dX2; this.dY2     = dY2;
   }

   // Convert rectangle details to a string ...

   public String toString() {
          return "Rectangle: Corner 1: (x,y) = " + "(" + dX1 + "," + dY1 + ")\n" +
                 "           Corner 2: (x,y) = " + "(" + dX2 + "," + dY2 + ")\n";
   }

   // ====================================
   // Compute rectangle area and perimeter
   // ====================================

   public double area() {
       return Math.abs( (dX2-dX1)*(dY2-dY1) );
   }

   public double perimeter() {
       return 2.0*Math.abs(dX2-dX1) + 2.0*Math.abs( dY2-dY1 );
   }

   // Exercise methods in the Rectangle class ....

   public static void main ( String args[] ) {

      System.out.println("Rectangle test program     ");
      System.out.println("==========================");

      // Setup and print details of a small rectangle....

      Rectangle rA = new Rectangle( 1.0, 1.0, 3.0, 4.0 );
      System.out.println( rA.toString() );

      // Print perimeter and area of the small rectangle....

      System.out.println( "Perimeter = " + rA.perimeter() );
      System.out.println( "Area      = " + rA.area() );
```

```
   }
}
```

The script of program output is as follows:

```
prompt >>
prompt >> java Rectangle
Rectangle test program
==========================
Rectangle: Corner 1: (x,y) = (1.0,1.0)
           Corner 2: (x,y) = (3.0,4.0)

Perimeter = 10.0
Area      = 6.0
prompt >> exit
```

The Rectangle class has methods to create objects (i.e, Rectangle), convert the details of a rectangle object into a string format (i.e., toString), and compute the rectangle area and perimeter (i.e., area() and permieter(), respectively). The implementation uses two pairs of doubles (dX1, dY1) and (dX2, dY2) to define the corner points.

Suppose that, instead, the corner points are defined via a Vertex class, where

```
public class Vertex {
    protected double dX, double dY

    ..... details of constructors and other methods removed ...
}
```

The appropriate modification for Rectangle is:

```
public class Rectangle {
   protected Vertex vertex1;  // First corner point....
   protected Vertex vertex2;  // Second corner point....

   ..... details rectangle removed ....
}
```

Fill in the missing details (i.e., constructors and toString() method) of class Vertex. Modify the code in Rectangle to use Vertex class. The resulting program should have essentially has the same functionality as the original version of Rectangle.

Hint. Your implementation should make use of the toString() method in Vertex.

Question 2: 10 points

There are lots of problems in engineering where the position of point needs to be evaluated with respect to a shape. Evaluation procedures can be phrased in terms of questions. For example, is the point inside (or outside) the shape?; Does the point lie on the boundary of the shape?; Does the point lie above/below the shape? Does the point lie to the left or right of the shape?; How far is the point from the perimeter?

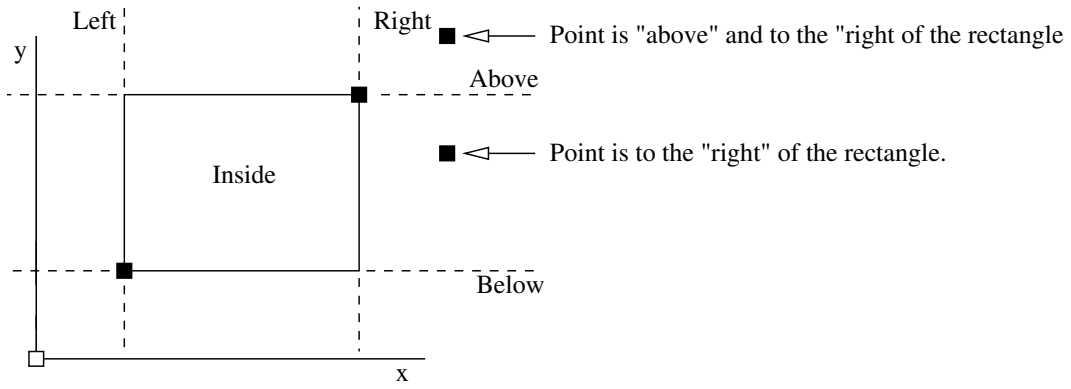

Figure 2: Classification of an (x,y) cooordinate relative to a rectangle

Figure 2 illustrates these ideas for one of the simplest cases possible, one point and a rectangle. Extend the functionality of the Rectangle class so that the possition of a point can be evaluated with respect to a specific rectangle object.

The appropriate method declarations are as follows:

```
public boolean isInside ( Vertex v ) { ... }
public boolean isOutside ( Vertex v ) { ... }
public boolean isOnPerimeter ( Vertex v ) { ... }
public boolean isAbove ( Vertex v ) { ... }
public boolean isBelow  ( Vertex v ) { ... }
public boolean isLeft ( Vertex v ) { ... }
public boolean isRight ( Vertex v ) { ... }
```

If the (x,y) coordinates of a vertex are inside a particular rectangle, then `isInside ()` should return `true`. Otherwise, it should return `false`. From Figure 2 is should evident that some points will result in multiple methods returning true. For example, points in the top right-hand side of the coordinate system will be outside, to the right, and above the rectangle.

Fill in the details of each method, and then develop a test program to exercise the procedures. Perhaps the most straight forward way of doing this is to write an extensive set of tests in the main() method for class Rectangle.

Question 3: 10 points

Figure 3 shows three types of spatial relationship (touching, overlap, and enclosure) between two rectangle.

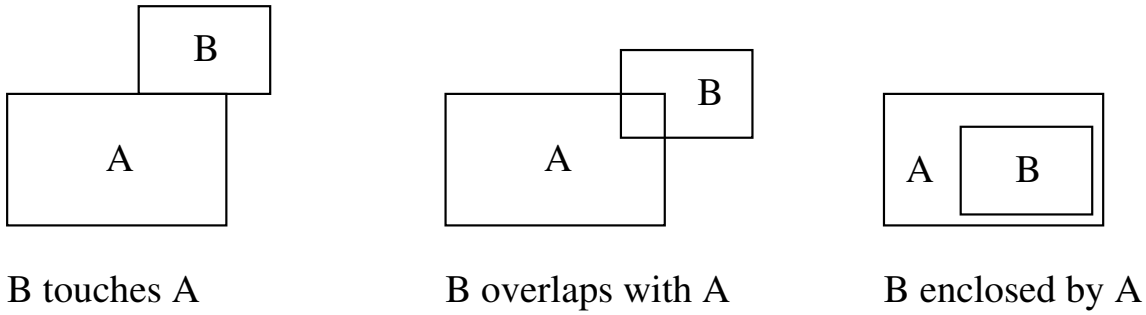

Figure 3: Spatial relationships between two rectangles

Things to do:

1. Extend the functionality of the Rectangle class to support the evaluation of these three types of relationships.
2. Develop a test program to exercise and validate computational procedures for each of these spatial relationships.