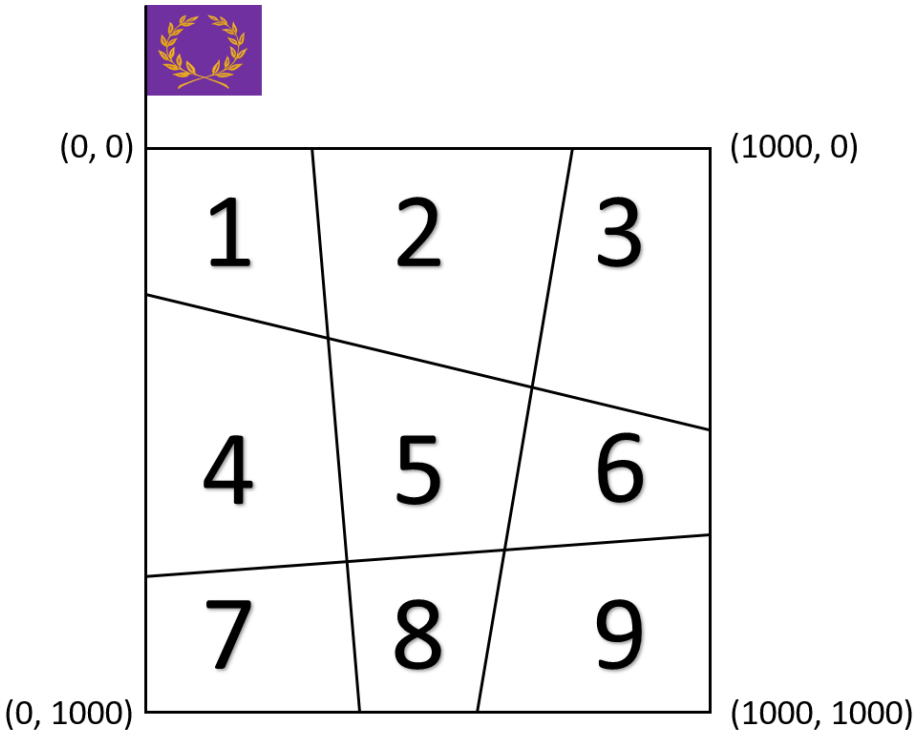
THE RUMMY GARDEN

Your father always wanted to have hobby garden, well as a hobby to look after it and finally bought one after he retired from his job. But with all that excitement of making his dreams come true, he did a very bad job at dividing his perfectly square garden into nine regions. He did this division with two vertical and two horizontal lines which happen to be tilted because of his poor measurements (If only, he knew an engineering student who could have helped him!) and numbered every region according to this plan:



Now, realizing you are an engineering student, instead of changing the regions, he wants help from you to find a method to calculate in which region a point is. Fortunately your father's plan involves the measurement system that he is using in his garden. This system uses a flag in the garden as an origin and the four corner's coordinates are written as an example. After days of thinking, you decide that the best method will be to write an object-oriented program.

When you finish your program, it will take two vertical lines (by taking coordinates of two points for each line) from left to right respectively and two horizontal lines from top to bottom again respectively from the user. Then it will continue to print out in which region the entered points are until the user enters a negative number.

An example runtime is given in the next page. (The black texts are outputs and the red texts are inputs.)

Enter the left most vertical line:
x1: 200
y1: 0
x2: 200
y2: 1000

Enter the right most vertical line:
x1: 800
y1: 0
x2: 800
y2: 1000

Enter the top most horizontal line:
x1: 0
y1: 200
x2: 1000
y2: 200

Enter the bottom most horizontal line:
x1: 0
y1: 800
x2: 1000
y2: 800

x: 500
y: 500
This point is in the region number 5.

x: 0
y: 0
This point is in the region number 1.

x: -1

POINT CLASS

You start by implementing a point class which will just hold two double variables as coordinates.

1) You add a default constructor which creates the point (0, 0).
2) You add a constructor which takes two parameter for x and y coordinate.
3) You add a copy constructor, just in case you might want to use it.
4) Finally you add get methods for each of the coordinates.

LINE CLASS

Second class that you implement is the line class. You have two options to define a line, you can either use a slope value and a point or just two points. You think carefully about which would be more helpful for the program you want to write and decide that the _____ is the best way.

1) You add a constructor which takes two points and converts into to your way of storage.
2) You add two function which work very similarly: onTop and onLeft. These functions take a point as a parameter and returns a Boolean result after some calculations.

For example:

```
Point p1 = new Point();
Point p2 = new Point(1, 1);
Line firstBisector = new Line(p1, p2);

Point p = new Point(0, 1);
firstBisector.onTop(p); //returns true
firstBisector.onLeft(p); //returns true

p = new Point(2, 2);
firstBisector.onTop(p); //returns false
firstBisector.onLeft(p); //returns false
```

Since you are good at math as much as you are good at programming, you realize that these two functions will always return the same value except for two special cases. When the line is perfectly horizontal or vertical. When you encounter a meaningless question you decide to return null. (If the line is vertical but the onTop function was called, for an instance.)

After some extensive testing of your two classes, it is time for you to implement the main method. You just hope that your father will not do wrong measurement ever again and enter invalid inputs to your program.