Exercise: Geometry

Assignment

Two classes, Point and Rectangle, were defined for you and you want to use those classes to solve some simple geometric questions.

Description

One of the basic things in object-orientation is what is called 'encapsulation'. In this case, some functionality was created inside the classes Point and Rectangle. In principle, all you need to know is to know how to use these two classes for solving your problem. You don't really need to know what's going on inside the classes Point and Rectangle. However, the module is defined as a Python module and there is no law that prevents you from looking inside and trying to understand what is going on there.

User Manual

Here's a little user manual for the classes. In order to be able to use the two classes Point and Rectangle you need to include the module called 'geometry.py' in your code. Copy the geometry.py file to your working directory, where you will create your own script, create a new empty file called 'geotest.py' (this is going to be your Python script), right-click on the file and select "Edit with IDLE" (by doing so the working directory is set to the current path), and include the geometry module as follows:

```
from geometry import Point, Rectangle
```

The Point and Rectangle classes are now ready for use.

Class Point represents a point in 2D space. Points are created like this:

```
p1 = Point(1, 8)
```

Rectangles are 2D objects and can be created like this:

```
r1 = Rectangle(p1, 3, 5)
```

Or directly, without creating a separate corner point:

```
r1 = Rectangle(Point(1, 8), 3, 5)
```

The Data

The data we use is defined in the following two tables. Your 6 points p_n are:

point	x	y
p1	1	8
p2	4	4
p3	6	6
p4	6	1
p5	8	8
p6	8	3

Your 4 rectangles r_n are, represented as (ll = lower-left corner):

rectangle	llx	lly	width	height
r1	2	1	3	8
r2	3	2	6	3
r3	3	3	4	4
r4	8	7	2	2

As you might have guessed, point objects have two data attributes called 'x' and 'y'. Rectangles have attributes 'corner', 'width' and 'height', in which 'corner' contains a reference to a point object. This corner point represents the lower-left corner of the rectangle.

Points have a method 'distance()' for calculating distances between points. The result is a floating point number.

```
>>> print p1.distance(p2)
3.0
```

Inside the class Rectangle there are two functions for determining whether two objects have some point or area in common. The method 'covers()' returns a Boolean if there is overlap between a rectangle and a point, or between a rectangle and another rectangle. Examples:

```
>>> print r1.covers(p1)
True
>>> print r1.covers(r2)
False
```

Another method of Rectangle actually calculates the intersection of a rectangle and a point, or a rectangle and another rectangle. The intersection of a point and a rectangle results in a Point object. The intersection of two rectangles results in another rectangle object. If the intersection is empty, the result is 'None'. Example:

```
>>> r1 = Rectangle(Point(1,1), 2, 2)
>>> r2 = Rectangle(Point(2,2), 2, 2)
>>> r3 = r1.intersect(r2)
>>> print r3
Rectangle(Point(2, 2), 1, 1)
```

As you can see from the last statement, rectangle objects can be printed. Also point objects can be printed:

```
>>> print p1
Point(1, 1)
```

This makes it easy to test your code.

The Questions

Use the two classes Point and Rectangle to solve the following questions:

1. "Is point p1 covered by rectangle r1?"
2. "Is point p1 covered by rectangle r2?"
3. "Is point p1 covered by rectangle r3?"
4. "Is point p1 covered by rectangle r4?"
5. "Does rectangle r4 intersect with any of the other rectangles?"
6. "What is the intersection of rectangle r1 and r4?"
7. "Does rectangle r1 cover r2?"
8. "What is the intersection of rectangle r1 and r2?"
9. "What is the intersection of rectangle r1, r2 and r3?"
10. "Which of the other points is closest to p1?"

Bonus questions

1. "What is the minimum distance between ALL the different points?"
2. Point and Rectangle objects both have a method called 'plot'. What do you think do these functions do? What does `p1.plot('p1')` do? Experiment with these functions and visually check your answers to the ten questions above!