CECS 277 HOMEWORK IPI

OBJECTIVE: Do design work on a small set of classes that relate to each other as subtype/supertype, experience dynamic method lookup, and use an interface.

INTRODUCTION: Geometric objects have a number of features in common about them, but also a number of things about them that are distinct. For instance, each object has a location, but a square might be located by the coordinates of its upper left-hand corner, while a circle's location is its center. All geometric objects have an area and a perimeter, but a circle and a square will have different way to calculate such values. You need to develop and test a set of classes that implement the following business rules. Your test program will need to demonstrate all these methods (including the constructors) work properly in practice.

- All geometric objects share:
  - Area()
  - Perimeter()
  - toString()
- Circles are defined by:
  - A center point
  - A radius (must be > 0)
- Rectangles are defined by:
  - Location of the upper left corner
  - Width (must be > 0)
  - Length (must be > 0)
- Triangles are defined by three points.
  - Do not worry about checking whether the points are on the same line or not. That gets a bit gnarly.
- The formula for the area within a triangle is: $A = \sqrt{s(s-a)(s-b)(s-c)}$ where $s = \frac{a+b+c}{2}$ where a is the length of the first side of the triangle, b is the length of the second side, and c is the length of the third side.

PROCEDURE:

- Part 1
  - Based on the business rules given above, decide which properties and which methods go into the classes. You will need one generic class for what is common to all three of them.
    - Decide whether the generic class must be abstract or concrete.
  - Build a Point class that has:
    - X – the x coordinate of the Point.
    - Y – the y coordinate of the Point.
    - A method called distance:
      - Accepts one explicit parameter, another Point
      - Returns the distance between the implicit parameter Point and the explicit parameter Point.

        - toString()
- Part 2
    - Modify the `GeometricObject` class to implement the `Comparable` interface.
    - Implement a **static** method on `GeometricObject` that accepts two arguments: both of type `GeometricObject`. Max will return the **object** that is the larger of the two.
        - Use the area of the `GeometricObject` for your comparison.
        - Use the results from `Comparable` in your max routine.
    - Demonstrate that you can sort an array of `GeometricObjects`
        - Create an array of `GeometricObjects`. Make sure that you have at least one Circle, Rectangle, and Triangle in the array.
        - Sort your array of `GeometricObjects` using Arrays `parallelSort` method. Please note that `Arrays` is a class in the Java API, and the `parallelSort` method is part of that class. You do **not** have to write that sort.
        - Display the sorted array.
- Part 3
    - Update your `GeometricObjectRunner` class to include a function called `FindMax`:
        - Takes an Array of `GeometricObjects` as its only parameter
        - Returns the largest `GeometricObject` that it found in that array.
    - Demonstrate your `FindMax` routine by using it to find the largest `GeometricObject` in your array of `GeometricObjects` before you sort them and print that `GeometricObject` out to the console.

WHAT TO TURN IN:

- Circle.java
- GeometricObject.java
- GeometricObjectRunner.java
- Point.java
- Rectangle.java
- Triangle.java
- Your console output named console.txt

SAMPL OUTPUT: Note – your output does not have to look exactly like this:

```
Before the sort
Circle: Center at: Point: X: 1.0 Y: 1.0 Radius: 3.0 Area: 28.274333882308138 Perimeter: 18.84955592153876
```

```
Triangle: Point: X: 0.0 Y: 0.0, Point: X: 1.0 Y: 1.0, Point: X: 2.0 Y: 0.0 Area: 0.9999999999999996
Perimeter: 4.82842712474619
Rectangle: Upper left corner at: Point: X: -2.0 Y: 2.0 Width: 2.0 Length: 3.0 Area: 6.0 Perimeter: 10.0
Circle: Center at: Point: X: 5.0 Y: 5.0 Radius: 2.5 Area: 19.634954084936208 Perimeter: 15.707963267948966
Largest found was: Circle: Center at: Point: X: 1.0 Y: 1.0 Radius: 3.0 Area: 28.274333882308138
After the sort
Triangle: Point: X: 0.0 Y: 0.0, Point: X: 1.0 Y: 1.0, Point: X: 2.0 Y: 0.0 Area: 0.9999999999999996
Perimeter: 4.82842712474619
Rectangle: Upper left corner at: Point: X: -2.0 Y: 2.0 Width: 2.0 Length: 3.0 Area: 6.0 Perimeter: 10.0
Circle: Center at: Point: X: 5.0 Y: 5.0 Radius: 2.5 Area: 19.634954084936208 Perimeter: 15.707963267948966
Circle: Center at: Point: X: 1.0 Y: 1.0 Radius: 3.0 Area: 28.274333882308138 Perimeter: 18.84955592153876
Largest found was: Circle: Center at: Point: X: 1.0 Y: 1.0 Radius: 3.0 Area: 28.274333882308138
```