Homework 2: Shapes

COSC150

Assigned: 9/13/2016
Due: 9/27/2016 at 11:59pm ET

This assignment is worth 150 points.

Description:

The goal of this mini-homework is to become more familiar with some basic Java concepts, including interfaces, polymorphism, generics, collections, and class hierarchies, as well as best programming practices such as good naming conventions.

You'll be writing a program, called FunWithShapes, that reads a file called *shapes.txt*. Briefly, your program will instantiate a "Shape" instance for each shape defined in *shapes.txt*, and print the sum of the areas and the sum of the perimeters over all of the defined shapes.

Format of shapes.txt

Each line of *shapes.txt* defines a separate geometric shape. The four possible shape definitions are:

- `circle` *`radius`*
  where *radius* specifies the radius of a circle.
  For example, the line may read:
  `circle 5.0`
  which defines a circle with a radius of 5.
- `square` *`side_size`*
  where *side_size* specifies the size of a side of a square.
  For example, the line may read:
  `square 15.0`
  which defines a square where each side has length 15.
- `rectangle` *`height width`*
  where *height* and *width* respectively specify the height and width of a rectangle.
  For example, the line may read:
  `rectangle 3.0 5.0`
  which defines a rectangle width height 3 and width 5.
- `triangle` *`side1 side2 side3`*
  where *side1, side2,* and *side3* respectively specify the lengths of the three sides of a triangle.
  For example, the line may read:
  `triangle 3.0 4.0 5.0`
  which defines a triangle with side lengths 3, 4, and 5.

An example shapes.txt file is included in the handout for this assignment. The example looks like:

```
square 5
rectangle 5 10
square 3
triangle 3 4 5
```

The above is just one example. Your program should be able to accept any list of square, rectangles, triangles, and circles.

Your FunWithShapes program should read *shapes.txt* (we give you code to help you with this). For each line in the file, your code should instantiate the appropriate shape object. In particular, you should write Square, Rectangle, Circle, and Triangle classes, all of which will be subclasses of the Shape class which we provide.

Provided Files

As part of this assignment, we provide you with the following files:

Filename	**Description**
Shape.java	Defines an abstract class Shapes. Your Square, Rectangle, Circle, and Triangle classes should be subclasses of Shape.
ShapeException.java	An exception that should be thrown when *shapes.txt* either is not found or contains one or more lines that do not conform to the specification above/
ShapeDescription.java	Each instance of ShapeDescription describes a line in *shapes.txt*. The getShapeType() method returns the type of shape (circle, square, etc.), and the getDoubles() method returns a vector of doubles that define the size of the shape (e.g., radius for circles, width and height for rectangles, etc.).
ShapeHandler.java	An abstract class whose constructor reads *shapes.txt* and converts each line in the file to a ShapeDescription object. **Your FunWithShapes class should be a subclass of this class.**

These files can be downloaded either as a jar file (hw2-handout.jar) or as a zip file (hw2-handout.zip). Both files are available as attachments to the Autolab assignment.

- The .jar file may be uncompressed with the command "`jar xvf hw2-handout.jar`"; or
- The .zip file may be uncompressed normally.

Once you download the files, you'll need to import them into your Eclipse project.

TL;DR

Well, you should really read all of the above. It's wicked important.

But to summarize, you should:

- write Square, Circle, Triangle, and Rectangle classes, all of which will extend Shape.java. The constructors of these four classes take in a ShapeDescription and should use that to construct the particular shape.

- Write the FunWithShapes class, which will be a subclass of ShapeHandler. You will need to override the convertDescriptionsToShapes() function which should iterate through the shapeDescriptions vector (see ShapeHandler) and instantiate the apropriate shape object (Circle, Square, etc.) for each ShapeDescription.

- In main() in FunWithShapes, compute and output (in this order) the sum of the areas of the defined shapes and the sum of the perimeters of the defined shapes. You'll need to override the sumOverAreas() and sumOverPerimeters() methods from ShapeHandler.

Other Requirements

- This is not a group assignment. You must solve this assignment individually, and you should not work collaboratively on the assignment.

- **Important: your program must output ONLY the following:**
  - The sum of the areas of all shapes, followed by a newline ("\n")
  - The sum of the perimeters of all shapes, followed by a newline ("\n")

  In other words, your program should output exactly two lines. Any other output will upset the autograder and will result in loss of points.

- Your program should accept any well-formed *shapes.txt* file.

- Your program should not crash. It may exit gracefully (e.g., with a polite error message) if it encounters an error -- e.g., an undefined shape type.

- You should use the compiler annotation `@Override` when you override methods.

- You must use getters and setters.

- To aid in autograding, all of your code must be in the default package namespace. In other words, do <u>not</u> call "package" anywhere in your code.

OK, I've just finished my amazing FunWithShapes implementation. How do I turn this thing in?

What to turn in:

You will turn in a single file archive (kinda like a .zip file) that contains your entire program. It should include everything necessary to compile on its own. To generate this file, do the following:

1. Right-click package name, select “Export…”
2. Select “Java -> Jar File”
3. In the Jar File Specification wizard:
    a. **MAKE SURE THAT YOU SELECT “Export Java source files and resources”.** We will not award any credit without the source Java files.
    b. Save the file as “hw2.jar”. You may want to use “Browse…” to easily figure out where Eclipse is saving the file.
4. Upload hw2.jar to Autolab at https://autolab.georgetown.edu/. In Autolab, look for the “Shapes” assignment.

Where to go for help:

For questions about this assignment, please post to Piazza at https://piazza.com/georgetown/fall2018/cosc150/home.

A Grading Rubric

Amazing Feat	**Points Awarded**
You submitted something. Anything.	10
It compiles!	30
Square objects are correctly instantiated	10
Rectangle objects are correctly instantiated	10
Triangle objects are correctly instantiated	10
Circle objects are correctly instantiated	10
Area correctly computed (per shape; multiply by 4 for all shapes)	5 (per shape)
Perimeter correctly computed (per shape; multiply by 4 for all shapes)	5 (per shape)
Total area and perimeter correctly computed and outputted	30

Please note that we may deduct points for program crashes, not using getters and setters, and other violations of good programming practices (as covered during lecture).