**COMP 152** **Object-Oriented Data Structures and Algorithms**

# Lab/Hwk 3

*Assigned: September 2* *Due: September 9*

**Abstract**

The goal for this lab and associated homework is to practice object-oriented design and implementing class hierarchies. This assignment is based on an assignment for a similar class at James Madison University.

## 1 The Task

You need to design and implement a class hierarchy of shape objects. You'll be using the `turtle` module just like in lab 1.

Your class hierarchy should include at least the following classes:

- Shape (abstract base class)
- Circle
- Square
- RegularPolygon
- Rectangle

If you'd like some bonus points, you can include additional shape classes, such as:

- FilledPolygon
- FilledCircle
- Dot

Each of these represents a broad category of shapes with some common features. Each concrete (non-abstract) shape should have a method `draw()` that uses the `turtle` module to draw itself on the screen. Every shape should also have `perimeter()` and `area()` methods[1].

Think carefully about the class hierarchy you want to use for this. You are free to design any other helper classes you think might be useful in implementing the required classes.

## 2 Lab

For the lab, draw a formal UML class hierarchy diagram of your design. You will need to determine the member functions and attributes that should be in each class, except you already know (see above) that you need a `draw()` method in your abstract `Shape` class. Each object should keep track of enough information to be able to draw itself without further input. You should look for opportunities to take advantage of inheritance and/or containment, minimizing duplicate code and member variables whenever possible.

Draw your design on a piece of paper and show it to me when finished, either by scanning and emailing or by holding it up to your webcam. Taking a picture is allowed, but please make sure it is readable; using an app like CamScanner is preferrable.

[1] The following webpage might be useful: `https://www.wikihow.com/Find-the-Area-of-Regular-Polygons`

# 3 Homework

Once you have finished your class hierarchy design you can begin implementing it in Python. You can code this in whatever environment you like – repl.it (use "Python with turtle" and see the note below), your own machine (requires installing Python – I suggest installing "Anaconda"), the lab machines, or the department server. If you need help logging on to the department server, let me know and I will walk you through it (you'll need to use VNC). If you'd like to use the lab machines, they have a program called "Anaconda" installed. Inside of Anaconda you can click on the "Spyder" IDE and develop there.

Notes:

- The `draw()` method should take no parameters and be implemented in each concrete class appropriately.
- To draw a circle, use the `circle` method in the `turtle` module.
- Include a `main()` method that tests all of your classes. Automatically testing graphical programs is quite difficult, so the style of tests we'll write require manual verification that they draw the right things.
- You can use `from turtle import *` to make writing the code a little more convenient. Usually I don't recommend this, but for small programs it is okay.
- If you implement your hierarchy correctly, you should be able to create a list of shapes in `main()` and then iterate over the list, calling `draw()` on every object regardless of what type of shape it is.
- Don't forget doc-strings for each class and method!
- If you use repl.it, you will not be able to use abstract classes and methods directly, as discussed in class. In the doc-string for an abstract class just note that is abstract. To declare an abstract method, declare it normally but put `raise NotImplementedError("error")` in the method body (replace "error" with something appropriate).

## 3.1 Submission

To submit the homework, send me an email with either (a) a single Python file attached, or (b) a link to a repl.it repl.

# 4 Resources

- Chapter 2 of the textbook
- the `turtle` documentation: `https://docs.python.org/3.7/library/turtle.html`