CS 2101 Lab 2: Object-Oriented Design

Due: 21 September 2016

The goal of this lab is to give you experience with some of the design principles of object-oriented programming, particularly the mechanisms in Java that facilitate the use of these principles. You do not need to write a working program, although writing such a program in which you test your classes would help satisfy you that your classes are working correctly. If you do this, please turn in your entire program, including the testing code. But, you do not need to write a working program. I will ignore minor syntax errors, such as forgetting a semicolon, since the focus in this lab is on object-oriented design. In any case, I want you to turn in a typed copy of your classes.

As always, the first thing you should do is skim the entire lab once through to get an overview of what is involved. Then you should read it carefully a second time, focusing on the details. Then you should outline the structure of the required classes. You should not start writing details of your classes until you have thought about the overall structure.

In this lab, you need to write a **Point interface**. The functionality of a **Point** is completely specified below, so all you need to do is put it inside a Java **interface**. Then, you will write a **Point2D** class that implements this **interface**. Next, you will write a **Point3D** class that **extends** the **Point2D** class. You will use the **Point2D** class to improve the **Circle** class I wrote in class so that it includes a **Point2D** object that contains the coordinates of the center of the **Circle**. Finally, you will write a **Sphere** class that **extends** the **Circle** class. You are free to use any of the code from my class examples without citing me as the source.

The functionality that you will put into your **Point interface** is the following:

```
public Point getPoint();
public Point getCopy();
public double getX();
public double getY();

public void setPoint(double x, double y);
public void setX(double x);
public void setY(double y);

public String toString();
```

Note that **getCopy** must make a *copy* of the object and return that, not simply return a reference to the object. Implement this interface in a **Point2D** class.

Now, **extend** your **Point2D** class to create a **Point3D** class. The additional piece of data in this class will be a **z** coordinate. Override **Point2D** methods and add new methods as necessary so that this class has the same functionality as the **Point2D** class, but in three dimensions. Use **super** wherever possible.

Add a **Point2D** instance variable to my **Circle** class that represents the center of the circle. You will need to change one of the constructors and the **toString** method, and write a getter and setter for the **Point2D** object that specifies the center. Note that the parameter of the **setCenter** method should be a **Point2D** object. Note: You must assign a *copy* of the **Point2D** parameter to the instance variable representing the center.

Finally, create a **Sphere** class by extending the **Circle** class. Override **Circle** methods and add new methods as necessary so that this class has the same functionality as the **Circle** class, but in three dimensions. Again, use **super** wherever possible. The formulas for the surface area and volume of a sphere are readily available online and you do not need to cite your source for these.

NOTE: All of your classes should have complete documentation, including a summary at the top that describes what the class does, a method summary above each method, and comments in your code, as necessary.

Submitting Your Program

Please turn in a hardcopy of 1) all your **.java** files. Also, submit them online by putting them in a single folder named with your login ID and the lab number, using this format: **<login-ID>-Lab 2**, so my folder would be **smajerci-Lab 2.** Compress that folder and upload it to Blackboard in the **Lab Submissions** section. If you do not know how to compress a folder, ask me or one of the TAs.