BSM 261: Object Oriented Programming
LAB 10: Building a Class Hierarchy

December 15/16, 2022

Date Performed:	December 15/16, 2022
Instructor:	Orhan özgüner
Teaching Assistant:	Yunus Emre Avci

1 Class Material, Lab&Quiz Scores

You can access to the class material from the following link:
https://drive.google.com/drive/folders/1yo4pvaTngQ1L2TkHuWPCzbfapLDyvwUt?usp=sharing

You can check your lab attendance and quiz scores from the following link:
https://docs.google.com/spreadsheets/d/1xKQEqMSGlj3RBo9kLh97UzbTO1fyCvaT/edit?usp=sharing&ouid=104851561367296854968&rtpof=true&sd=true

2 Getting Started

Before you perform the lab please check the following information:
• Make sure that the lab computer you are using has DrJava downloaded into the desktop. Here is the jar file to check:

drjava-beta-20190813-220051.jar

• If you can not locate DrJava, please download it from the following link and place it to your desktop.

https://sourceforge.net/projects/drjava/files/latest/download

• If you have the DrJava jar file in your desktop, you are ready to perform your lab. In order to use jar files in linux, you need to run it from terminal. Run the following commend from the terminal to start DrJava

java -jar /Desktop/drjava-beta-20190813-220051.jar

3 Lab Rules

The purpose of the lab is to get you used to using DrJava, to start experimenting with Java, and to digest the topics you learn over the week.
Here are the lab rules:

- The teaching assistant will make a short recap of the week. Then you will perform the lab in the recitation hour.
- Your grade will be determined by your participation and finishing the assigned work during the recitation. When you get stuck on a question, please ask for help.
- Once you finish all of the tasks, show your work to the TA to receive full credit.
- **If you did not join the lab, or joined more than 10 minutes late or did not show your work to the Teaching Assistant, or did not finish all of the questions in class time you will receive 0.**

4 Lab Instructions and Questions

In this lab, you will be completing code for a Shape Abstract Class and 3 subclasses, Triangle, Rectangle, and Circle.

4.1 Shape Abstract Class

Shape class will be an abstract class with no constructors since the shape is not specific enough to describe. Shape defines 2 abstract methods calculateArea() and calculateCircumference(). Both return doubles representing the area and circumference of the shapes respectively.

4.2 Circle class

For this part, you will be writing the Circle class. This class will have a constructor, getter/setters, and the abstract methods defined for this class.
The Circle class constructor will take in a double that represents a radius of a circle. This value should be stored in an instance variable that is private, since we will use getter/setters to manipulate and obtain the value of it.
Since this class extends the Shape class, this means that the Circle class must implement the abstract methods in the Shape class. In this case, those methods are calculateArea() and calculateCircumference(). For each method, return the area and circumference respectively of a circle with the radius specified by the object.

4.3 Rectangle class

For this part, you will be writing the Rectangle class. This class, like the previous, will have a constructor, getter/setters, and abstract methods defined for this class.
The Rectangle class constructor will take in two doubles, one for height and the other for width. These values should be stored in instance variables that is private, since we will use getter/setters to manipulate and obtain the value of it.
Like in the Circle class, the Rectangle class also extends the Shape class meaning the abstract methods in the Shape class must be defined in the Rectangle class. Again, these methods are calculateArea() and calculateCircumference(). It may be obvious to you that rectangles do not have circumferences but perimeters. For the calculateCircumference() method, return the perimeter of a given rectangle object. Also, return the area of a given rectangle in the calculateArea() method.

4.4 Triangle class

For this part you will be writing the Triangle class. As in the previous class, the Triangle Class has a constructor, getter/setters and abstract methods defined.
The constructor for the Triangle Class will take three doubles one for each side. These values should be stored in instance variables that is private, since we will use getter/setters to manipulate and obtain the value of it.
The methods that must be implemented are calculateArea() and calculateCircumference(). For calculateArea(), return the area of a triangle. As before with the Rectangle Class, there is not a circumference for a triangle. Therefore, for this method return the perimeter of a triangle. As you have the base of the triangle, and because this class is limited to just equilateral triangles, you should be able to easily calculate the perimeter of the triangle.

Demostrate your code to your lab instructor to receive full credit.

LAB is over at this point but I wanted to add a challenge question for you to work on yourself:

4.5 Challenge: Add a Square Class

For this part you will be writing the Square class. As in the previous class, the Square Class has a constructor, getter/setters and abstract methods defined.
The constructor for the Square Class will take a double for its side. These value should be stored in instance variable that is private, since we will use getter/setters to manipulate and obtain the value of it.
The methods that must be implemented are calculateArea() and calculateCircumference().

4.6 Challenge: Add a Ellipse Class

For this class, there is no definition. You should decide how the constructor should be and what values needs to be stores. Only restriction is that Ellips should be in the hierarchy and implement calculateArea() and calculateCircumference() methods.