Lab 9: Inheritance in Java

Introduction

Inheritance is an effective class building tool in object-oriented programming. Through inheritance, a new class (or subclass) can obtain all of the data and methods declared in a previously created class (or superclass) that it "extends". If the data in the superclass is declared to be `protected` (or `public`), the subclass can also access or manipulate the data directly within its own methods. Before completing this lab, review the material on inheritance discussed in lecture and also provided in the course slides and it the textbook.

MySquare Class

In Lab 5, you completed the `MyRectangle` class, which is a simple representation of a rectangle. Review Lab 5 before completing this lab and retrieve the `MyRectangle` class that you completed for this lab, since you will need it again in this lab. Before starting this lab, edit your `MyRectangle` class in the following way:

- change the declaration of your instance variables from `private` to `protected`

This will enable access of these variables by your `MySquare` subclass.

What to Do?

In this lab, you will implement a new class with the following header:

```
public class MySquare extends MyRectangle
```

Your `MySquare` class should NOT have any new instance variables. If you think about it, you will see that the instance variables already present in `MyRectangle` are sufficient. **Note that since you just changed the declarations of your instance variables to `protected`, a subclass (`MySquare`) can have a direct access to the instance variables of the superclass (`MyRectangle`).** For this lab, you will need some new methods in your `MySquare` class as follows:

- `public MySquare(int x, int y, int side)`: This is a constructor that allows new objects to be created. `x` and `y` are the location coordinates and `side` is the side length.
- `public MySquare()`: This is a constructor that sets location `(x,y)` to `(0,0)` and `side` to 0.
- `public String toString()`: See output for effect of the `toString()` method.
- `public void setSize(int w, int h)`: Redefine the `setSize()` method. This must be done because the inherited version allows the width and height to difer but in a square they must be the same. In this version, if the width and height are not the same, the method should output a message and not change anything.
- `public void setSide(int side)`: This is a new method that updates the size of the square. Think about how this will be implemented using the existing inherited instance variables.

Your `MySquare` class must run with the `Lab9.java` program with no changes. To see how your output should look, see the file `lab9out.txt`. Note that there are some methods that are utilized in `Lab9.java` which are inherited and which you should not redefine in your `MySquare` class. See details in the comments in `Lab9.java`.

Grading

The grading criteria for this lab is as follows:

- Constructors (4 points)
- `toString()` (2 points)
- `setSize()` (2 points)
- `setSide()` (2 points)

Due Date and Submission

Once you completed the program, you must demonstrate your program for your Lab TA. Once your TA already checked you, **DO NOT FORGET** to zip (compress) your `MyRectangle.java` and `MySquare.java` files into the new file named `Lab9.zip` and submit it to the CourseWeb under this lab by the due date.

If you do not complete the lab this week, you may finish it and submit your code to the CourseWeb before the due date. However, you need to demonstrate it to your TA at the beginning of next week's lab.

No late submission will be accepted.