INFSCI 0017 – Fundamentals of Object-Oriented Programming (Fall 2018)

Lab 6

Note: If you do not get to demonstrate your exercise during this lab session, you may finish it between sessions and demonstrate it at the beginning of your next lab session.

Topics Reviewed

1. Classes
2. Methods
3. Accessors / Mutators (Getters & Setters)
4. Instance variables (Properties)
5. Class constructors

Introduction and Background

In the text and in lecture we saw some examples of using and writing simple classes in Java. We saw the basics of how instance variables, constructors, accessor methods and mutator methods are declared and used. We also saw how to accomplish encapsulation with data hiding by declaring our instance variables to be private and by declaring our instance methods to be public. Although this will not always be the case, it is a good initial rule-of-thumb to use for writing new classes. In this lab you will complete a new, simple class and see it work by running it with a simple driver program.

MyRectangle Class

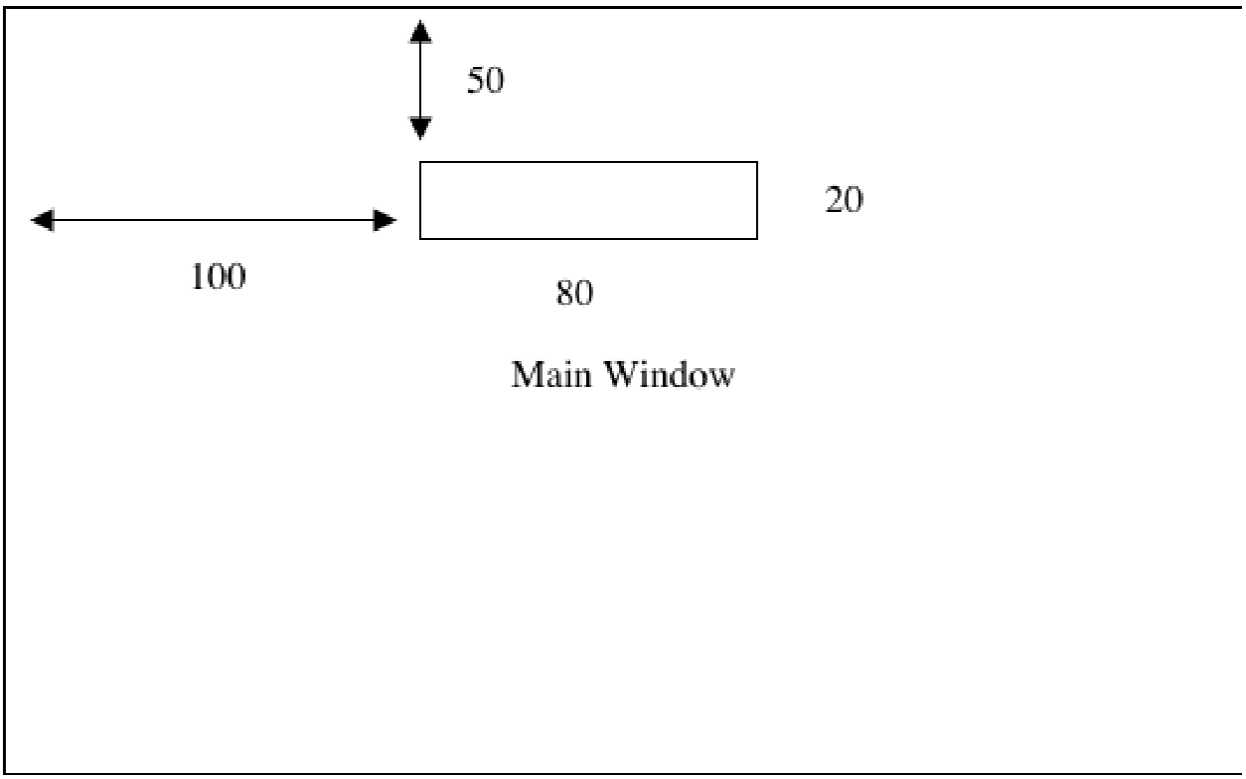
You will write a simple class that will represent a rectangle. Later on we will see how to do this with graphics, but for now we will just have to use our imaginations for the visual representations of the rectangles.

Instance Variables

Clearly a rectangle requires the dimensions of the sides. We will call these dimensions width (for the X-dimension) and height (for the Y-dimension). The types of these dimensions could be either floating point (double) or integer (int). Since pixels on the computer display are discrete, we will make these dimensions int. In addition to its size dimensions, a MyRectangle must have a position on the "screen". We will base this position on the upper left corner of the MyRectangle. The graphics coordinate space for computers starts at (0,0) and proceeds to the right for positive X and down for positive Y. Call the X position startX and call the Y position startY, and, like the dimension variables, make these both ints. For example, consider the following instance variable values for a MyRectangle:

- startX: 100
- startY: 50
- width: 80
- height: 20

This would define the MyRectangle shown (not exactly to scale) in the figure below.



We do not want users of MyRectangle to have direct accessto the instance variables, so we will declare them as **private**.

Constructors

Recall that constructor methods are used to create new objects of a given class. They are special methods in that they have no return type (not even void). For the MyRectangle class you will have two constructors. One is a ***default constructor*** -- this is used to create objects when no arguments are used. For MyRectangle the default constructor will initialize all 4 of its instance variables to 0. The second constructor will have four parameters (in order): X position, Y position, new width and new height, and it will simply initialize the instance variables from the parameters.

For example, the MyRectangle above could be created by the following:

- MyRectangle R = new MyRectangle(100, 50, 80, 20);

and a "default" MyRectangle could be created by the following:

- MyRectangle R2 = new MyRectangle();

Accessors

Accessor methods allow us to get information from objects or have them perform tasks that do not alter the objects themselves. There is no special designation for accessor methods; rather we label them as accessors based on what we use them for. For MyRectangle we will have the following accessors:

- ***public int area()***: return the area of the given MyRectangle
- ***public String toString()***: return the MyRectangle's information in the form of a String. See sample output for details.
- **public boolean isInside(int X, int Y)**: Return true if point X, Y is inside the MyRectangle, and false otherwise

Mutators

Mutators allow us to change the data within an object. As with accessors, there is no special designator for mutators; we label them based on what we use them for. For MyRectangle, we will have the following mutators:

- ***public void setSize(int newWidth, int newHeight)***: change width and height of this MyRectangle to the values passed in
- public void setPosition(int newX, int newY) // change X and Y position of this MyRectangle to the values passed in

Class Skeleton

A skeleton of the class above is online in the following file:
MyRectangle.java
Download this file onto your account or computer and fill in the details so that the class will work as intended.

Main Program

Main programs in object-oriented programming are typically simple and used primarily to "set up" the objects, which in turn do the rest of the work of the execution. However, in this case our main program will just demonstrate the functionality of our new class. It is provided online:
Lab5.java
Download this file and compile once you have completed the details of MyRectangle.java. Then run it to make sure it works as intended. Also read the comments carefully -- some additional important / useful information is provided there.

Grading

Once you have your MyRectangle.java class working properly with the Lab6.java main program, compile it and run it for your TA

There will be 4 raw points for this lab, broken down in the following way:

1. The constructors work correctly (1 point)
2. The area method works correctly (1 point)
3. The isInside method works correctly (1 point)
4. The "set" methods work correctly (1 point)

Notes and Hints

When compiling a Java main program, sometimes classes are needed that are not written in the main program file. Some of these classes are within packages and can be accessed using import statements. For example, Scanner and ArrayList are both part of the java.util package. If a class needed by a main program is in a file in the same directory as the main program, the Java compiler will find it automatically and compile it, because the name of the file matches the name of the class. For example, in this lab your main program is called Lab5.java. When it is being compiled, the compiler sees that the class MyRectangle is used, so it automatically looks to compile file MyRectangle.java in the same directory. If MyRectangle.java is not present, the compiler will report an error that the class is not found (test this by renaming MyRectangle.java to something else and then trying to compile Lab6.java).