CSE 110 - Lab 9

Lab Topics

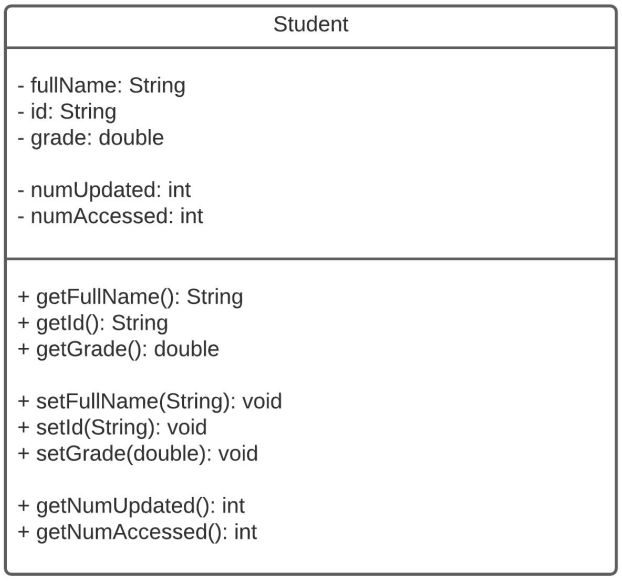
- Variable Scopes
- Object Getters and Setters
- Classes and Objects in Java
- Class Methods and Attributes

Lab Problem: Getters, Setters, and Scopes

In this lab, we are going to learn how to create an abstract data type, or a class, in Java to help store some data. At the same time, we will practice and get used to the idea of variable scopes and how a Java program calls methods in the runtime.

Part 1. Getters, Setters, and Variable Scopes

Getters and setters are two important elements in object-oriented design. They are usually referred to with the concept "*encapsulation*". Essentially, getters and setters are public methods serving as "accessors" and "mutators" respectively. As the names suggest, they are used to access or set information/states inside objects. As a result, the user would not accidentally access information of objects without a proper permission. Getters and setters are also helpful for program debugging tasks because they limit usage of information to some extent. In addition to getting/setting information, we also want to track how many times an object is updated and accessed by getters and setters. Specifically, the design of your Student class should follow this UML class diagram:



When you submit your program, you must submit both your **Student.java**, and **Lab9.java** from Canvas.

Grading Policy

- Code logic (up to 4pt):
  - -2 if the class Student is created in Lab9.java
  - -2 if the main method is duplicated in Student.java
  - -2 if there is any missing getters/setters
  - -2 if any permission modifiers (private and public) do not follow the spec
  - -2 if missing any class field (a.k.a. attribute, instance variable)
  - -2 if the types of class fields do not follow the spec
  - -2 if methods are declared as static accidentally

Sample Output

Below is an example of what your output should roughly look like when this lab is completed. The highlighted parts are the checkpoints and will be shown correctly when your Student.java is correct.

```
student1's name is Foo Bar
student1's ID is 10291029
student1's grade is 5.9

In student1, numUpdated = 3
In student1, numAccessed = 5

In main, numUpdated = 0
In main, numAccessed = 0
```