Wesley Yu
Data Structures
Prof. Anasse Bari
2/22/19

Method overloading:

I used method overloading a couple of times – most prevalently was in the Course class

- addStudent() and removeStudent() are both overloaded
    - e.g.: addStudent(fullname, username) AND addStudent(fullname);
- When a student is created and added into the studentList through the program, there are two variables required (String fullname, String username) for both methods.
- However, if the studentList is empty, the method calls itself and adds "NULL" to the List, which is just one variable (String fullname).
- When studentList gets to a size of more than 1, the method removes "NULL", also one variable (String fullname).

Method overriding:

I used method overriding a couple of times too. The first example was not implemented (due to a shortage of time, and it not being necessary for the program to run). Admin and Student both inherit from User, and both inherit a boolean method called permission();

For User, that method is set to False
Therefore Student inherits permission() to be False too.
However, the Admin class overrides the method and sets it to be True.

This could be used to provide Admin with special capabilities that the Student does not have access too—however in the current state of the program the Admin & Student Main Menus are separate.

Abstract Class:

User is an abstract class as an User object is never instantiated in the program, but its subclasses (Admin and Student) inherit its methods.

Inheritance:

Inheritance is shown in this program through the two subclasses of User—Admin and Student both inherit the majority of their methods from User. All the getters and setters, permission(), etc.

(next page →)

Polymorphism:

Overloaded and overridden methods are polymorphic, as you can reference the method in many different ways. Thus methods I had like addStudent(); removeStudent(); permission(); are all polymorphic.

Encapsulation:

Encapsulation – I achieved this through setting various variables to private, meaning that the data implemented can change without affecting other variables which depends on it… username, passwords, course info, etc.

Concept of ADT

Abstract datatype is a collection of data, with operations that indicate what the ADT does, without implement them.

User is an Abstract Data type – Collection of variables (username, password, etc.) which indicates what User does, but the implementation of the data is only achieved when the subclasses (Admin and Student) are instantiated.