

## CS108 Assignment 3: Control Structures and Arrays

**Due Date: 17 March 2005 @ Midnight**

### Learning Objectives

- Use an array of object references to create a data structure.
- Use control structures to develop simple algorithms to implement store, search & retrieve, list, and delete functions.
- Use control structures and String comparison methods to create a simple text-based menu.

### Problem Statement

A data structure is a way of organizing and keeping track of objects. In assignment 1, you created a StudentInfo object to track student information. In this case, you will create a data structure that can add (store), find, list, and delete entries, where each entry is a StudentInfo object.<sup>1</sup> You will also create a menu-based user interface for your application.

### Requirements

The **InfoManager** object will be responsible for managing an array of StudentInfo objects. The array will be a fixed size, initialized by a constructor argument. The following methods must be implemented on this object:

Method name, return type, parameters	Implementation notes
public void add(StudentInfo si)	Store the StudentInfo object into the array. Check for duplicates, overwrite an existing entry with the same name.
public StudentInfo delete(String name)	Search the array for a StudentInfo object with a matching name, and return the reference to that object. If not found, return null.
public StudentInfo find(String name)	Build a String which will comprise the entire listing of all entries, one per line.
public String listAllStudents()	Search the array for a StudentInfo object with a matching name. If found, return a copy of the reference to it, and set the corresponding array item to null. If not found, return null.

### Menu-driven User Interface

In order to test your add, delete, find, and list methods, you will create a menu-driven user-interface. The menu could be implemented in the main method in the InfoManager class, or in another method of your choosing. The behavior of the menu will be to display a list of choices, collect input from the user, and then execute the user's choice. After executing the operation, the application will return to the menu.

The following commands are required:

- ADD will prompt the user to input the name, email, etc. for a new StudentInfo object.
- FIND will prompt the user for the name of a student, call the find method on the InfoManager, and then print out the StudentInfo object or else a message indicating a record for that name was not found.

---

<sup>1</sup> You may use your own StudentInfo object, or else make use of the one I have written, which I located in the cs108.jar file you used in lab 5 <http://people.bu.edu/azs/academics/cs108/lab5.html>.

- LIST will call the InfoManager's list method, and will
- DELETE will prompt the user for the name of a student, call the delete method on the InfoManager, and then print out the StudentInfo object or else a message indicating a record for that name was not deleted.
- QUIT will call the System.exit() method to end the application.

### Hints/getting started

- You might want to **start by writing just the user-interface and menu** – using the work you did in lab 5 and connecting to my implementation of the InfoManager and StudentInfo classes. By doing this, you will be able to ensure you can test your work when you get into writing the add, delete, find, and listAllStudents.
- Next, you will probably want to **write just one method at a time** – add will be the easiest, followed by list, find, and delete will probably be the hardest. Write one method, test it, and do not write the next one until you get the previous one to work.
- It might be helpful for you to **write out pseudocode** for how to go about searching an array and comparing StudentInfo objects to see if you've found the object you want.
- Finally, you will probably want to **practice using the debugger** to step through your code line by line and watch the variable values at run time. It is a good idea to become comfortable with the debugger before you get stuck.

### Deliverables

- The java code file(s) (\*.java).
- A short executive summary, which will serve to introduce the application.

### Grading Criteria

- Correct syntax, no compile errors, good code formatting, follow naming conventions, appropriate comments in the code
- Proper implementation of methods: separate behavior into logical chunks, correct use of parameters, return values, etc.
- Proper use of arrays – declaration, initialization, accessing elements, storing and retrieving data
- Proper implementation of the text-based menu (all commands work, menu prints to the screen, etc.)
- Executive summary document: briefly explain the motivation for creating such an application, what the application does, how it works, why you used the programming elements you did for this type of project, as well as any lessons learned or recommendations for a revised implementation of this application.

### Questions?

**Bring them to class, or email me, [azs@bu.edu](mailto:azs@bu.edu).**

**NOTE: I will take all questions up until 6 hours before the assignment deadline. After that time, you're on your own. Plan your time wisely and start early.**