Semester Project  
Summer 2021

Over the course of the next six weeks, we will develop a “Registration” Program that will be able to maintain records of students, using various Data Structures. It is very important that you not fall behind on this project, since each part will depend on previous sections to make the project complete.

As each part is submitted, you will receive a grade on that individual part, based on 5 points, with the final project worth 30 points (30% of your grade). You will technically have one week for each component, but the components will be posted at least two (2) weeks before they are due. You may want to see how the parts work together:

“Warm-Up”: Create a *Student* class containing Student Name, (String) ID, and an “extended” class *StudentRecord* which includes an ArrayList of Courses (String) and write a tester to make sure the class works correctly. (Make sure that *equals* and *compareTo* work correctly!) We will review Inheritance and Polymorphism as needed!

Part 1: Write a program that can maintain a Roster of *StudentRecords*, using a LinkedList structure to enter students originally, and then storing the students into a sorted array. with a maximum of 10 students in the roster. The program should be able to search for a given student, add a student from the main roster, and drop a student from the main roster; students should also be able to add or drop a Course. If a student drops all courses, the student should be dropped from the main roster. The roster should be saved as a text file.

Part 2: Use a *Queue* structure to maintain a **waiting list** to add new students if a student is deleted. The waiting list will be saved as a separate text file.

Part 3: You will increase the class size to 20, using the waiting list to fill the class, and implement the initial linked structure as a BST. Once the registration is completed (“end of withdraw period”) the BST will be saved again as a sorted array, and the final roster will be saved as a text file.

Part 4: You will now create an alternate way to access student records, using a *Hash Table* and the Student ID to store, search and retrieve information.

Part 5: Document the project, including overall runtimes for each of the parts. You can create the document as you work on the project, and just submit the finished document separately, or just “Javadoc” to create the documentation as you write the project.

Each part of the project will be posted on BlackBoard in greater detail in the Assignments section with the submission information.