Project 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.

Project 2 will build on Project 1 with the following changes/additions.

* When the program opens, it will LOAD the *text file* that you created in Project 1 (into a sorted array), containing the sorted list of the current roster (just like opening the roster to start work).
* You will not need the “Save Changes” option, but you can just comment it out, since it will return in the next part of the program.
* You will not need to use the linked list methods for the roster, but you may adapt them to the waiting list (queue) methods. ALL ROSTER CHANGES WILL BE MADE IN THE SORTED ARRAY ONLY.
* The size of the array (roster) will remain 10 for this part of the project.
* If the user chooses “Add Student” from the menu, the program will check if the roster is full. If it is full, the new student will be added to a *waiting list* (queue). You may use the Java Queue, Linked List, or write your own queue methods.
* If a user chooses “Drop Student” from the menu, the program will check the waiting list. If there is a student on the list, he/she will be added to the class roster.
* When the user “Saves and Closes” the program , save the roster into a text file, and the waiting list into a separate text file.

Note: Since the original text file is already sorted, you can just load the students directly into the array (it will be sorted!)

Test your program by adding students until it is full (if it isn’t already), and then add 2 or 3 more students that will be “enQ”d into the waiting list. Then drop one or two students, adding from the waiting list to keep the roster full.

Project 2 is due Wednesday, July 23 at 11:59 PM.