ENSC 251 Lab Assignment 4

In the ENSC 251 course, you will work on a multi-lab project: you will design and implement a simplified graduate student admission system using the skills learned throughout this course (Object-Oriented Programming with C++). This project will be developed incrementally throughout a total of four labs, each weighing 10% of your final grade marks. All lab assignments will be carried out and evaluated in pairs. Some general grading logistics have been posted on our course website: https://coursys.sfu.ca/2019fa-ensc-251-d1/pages/labs. Please note that the detailed grading scheme for each lab will be released after your lab grading though: think about you are in a real interview, nobody will tell you what the detailed grading schemes are.

Through lab assignment 1 to 3, you have built a simplified graduate admission system that works with an *ideal user* and gained confidence about programming. In the final assignment, we will deal with some dirty details of the system that you have built: you need to make your system robust enough when you encounter an *evil user*.

As usual, for those student groups who didn't do well in assignment 3, the TAs have posted the code of assignment 3 from those top student groups who got bonus points, and you are free to use their assignment 3 code as basis for your assignment 4. **Note there will be no bonus marks for lab assignment 4 since it's the final one and no student will need to reuse the assignment 4 code.**

Lab Assignment 4:

Lab assignment 4 is based on assignment 3, and you will have to achieve the following tasks.

Part 1: Error Checking.

Some basic error checking has been added through lab assignment 1 to 3. Now let's add more to make your code more robust. You have to catch and handle the following errors. You are free to use regular if/else, switch/case statements, or try-throw-catch syntax to implement the error checking.

1. If any input field of a Domestic (and International) student in the input file is missing, catch the error, print out a message to hint the user that one filed is missing, and exit the program.
    a. E.g., if one line of input in the domestic-stu.txt looks like this "Mary,White,4.00,85", you should catch the error, and report that one field is missing, you don't need to report that the specific province field is missing though.
2. Make sure all string matching (e.g., search using firstName) in your program is case insensitive: e.g., "Mary", "mary", "maRY" should be treated the same. Note this applies to all cases including all the new error checking added in this assignment.
3. Make sure every filed of a Domestic (and International) student in the input file has a valid range. Specifically, please add the following checks
    a. Province must be one of the following: NL, PE, NS, NB, QC, ON, MB, SK, AB, BC, YT, NT, NU. Otherwise, print out an error message that it's not a valid province and exit the program

b. Country must be one of the following (note this is specific to our input test case, in real life, you should include all valid country names): Canada, China, India, Iran, Korea. Note there is a specific typo in your input file, which your program should detect and automatically fix it: your program should detect if the country name is "Idian", print out a warning message telling the user it's a typo and your program automatically fixes it to "India". For all other mismatches, print out an error message that it's not a valid country and exit the program

4. Make sure when a user types in a selection option which is not available (e.g., some option other than insert, search, delete, and merge in lab assignment 3), print out a hint to user: you typed in an option which is not available, here are the available options with detailed explanation… Then continue asking the input from the user. Do NOT exit the program.
5. Create an error input for each of the above case, and demo to your TA that your code can handle all the above errors.

Part 2: Unit Test.

Please review the concepts of unit test in Lecture 6, and write unit tests to validate the following major functions that you wrote in assignment 3 work correctly, including

1. Insert a DomesticStudent (and InternationalStudent) object into the DomesticStudent (and InternationalStudent) singly linked list in order.
2. Search existing DomesticStudent (and InternationalStudent) objects in the DomesticStudent (and InternationalStudent) linked list based on the user input information "application id", or "cgpa", or "researchScore".
3. Search existing DomesticStudent (and InternationalStudent) objects in the DomesticStudent (and InternationalStudent) linked list based on the user input information "firstName and lastName".
4. Delete existing DomesticStudent (and InternationalStudent) objects in the DomesticStudent (and InternationalStudent) linked list based on the user input information "firstName and lastName".
5. Delete both the head node and tail node from the DomesticStudent (and InternationalStudent) linked list in a single delete function.
6. Merge the two sorted DomesticStudent and InternationalStudent linked lists into a single Student linked list.
7. Search existing Student objects in the merged linked list based on the user input information "cgpa_threshold and researchScore_threshold".

Please make sure that your unit test covers: the "normal" cases, the boundary/corner cases, and the illegal cases.

Part 3: Algorithm Complexity Analysis

During your demo, explain to your TA what's the complexity of your insert, search, deletion, and merge functions in assignment 3 (i.e., all those functions mentioned in Part 2) using the big-O notation.

In addition to the actual coding implementation, you need to provide good commenting, naming, and other good coding styles (including the header file style to avoid the multiple includes problem); all these count in your lab assignment marking.

Note: For grading logistics and remote machine access, please refer to the course website. If you have any questions, please post them on Piazza.

Assignment Submission:

Your lab assignment 4 will be submitted electronically through CourSys. You will need to submit a single lab4.zip file. Failure to comply with this format could result in zero score on this assignment.

Submission Deadline:

Your lab assignment 4 is due at **11:59:59pm on Friday, Nov 22nd, 2019**. You need to meet the deadline: every 10 minutes late for submission, you lose 10% of this lab mark; that is, 100 minutes late, you will get zero for this lab.

Lab Demonstration:

You will have to demo your lab assignment 4 to your TA in the following lab sessions that you enrolled in: Tuesday (**Nov 26th, 2019**), Thursday (**Nov 28th, 2019**), and Friday (**Nov 29th, 2019**). Only code from your CourSys submission is allowed in the lab demo. Each student group has around 6-7 minutes to explain your code to the TA. If you fail to do the demo (without a medical note), or if it is determined that you do not understand the code being evaluated, you will be awarded zero on this lab assignment. Also please show up in the demo day on time (beginning time of each lab session), otherwise you will lose 0.5 mark.