Lab #4
Linked Lists, Stacks, Queues

Purpose:

The purpose of this lab is to write some code in C that uses aspects of the language that has been covered in class thus far.

Source code files for the lab are on Canvas. You will save all your source files for the lab to the GitHub account you created in Lab 0.

Preparation:

Review class presentations and chapters from the text according to the syllabus.

Procedure:

1) Start up CodeLite and open the workspace ECEGR2020 you created for Lab 0

2) In the workspace, create a new project: File -> New -> New Project. Name the project Lab_4 and make the Type a "Simple executable (gcc)"

3) For each function below, program, run and debug each. In your lab report, be sure to place the output of each function to show that your program worked.

Program the Following:

A) Update this structure from Lab 3:

```
struct student
{
   int  ID;
   char firstName[30];
   char lastName[30];
   float GPA;
} Student;
```

   To make it usable as a linked list. Update the program you wrote in Lab 3 that read in the file of students, but build a linked list of the students.

   Make sure to clean up the memory that was allocated before exiting the program

B) Update the program above to add a menu to the user after all the students are read in

1) List all Students
2) Add Student
3) Remove Student
4) Update Student
5) Quit

Add all the functions needed to list the students from the linked list, add a new student to the list (sorted by highest GPA), update a student's info, and remove a student. Note that the linked list will always need to be sorted by GPA, so think about what steps are needed to be done for each of the operations of the program. You could sort the whole list after every operation, but that is not very efficient especially if you have a list with thousands of students. Indicate in the code comments how you are avoiding a full sort of the whole list after each operation. When the program ends via the Quit menu option, the file of students needs to be updated with the latest list.

C) Update the StackArray.cpp program to be implemented using a linked list. Remove the capacity and maxsize variables as the stack can now resize without those limits, but do maintain what the current size of the stack is. What should be the implementation of isFull() and isEmpty() now?

D) Update the QueueArray.cpp program to be implemented using a linked list. Remove the capacity and maxsize variables as the stack can now resize without those limits, but do maintain what the current size of the queue is. What should be the implementation of isFull() and isEmpty() now?