Overview

Stacks are very useful for its backtracking features. Stacks can be used to implement recursive solutions iteratively because of the LIFO property. The purpose of this assignment is to give you some experience creating your own Linked List Stack object in C++.

Description

In this assignment, you will be implementing your own Linked List *Stack*. you are going to create a class called LinkedListStack. Here are the minimum requirements:

1. Your class has the following member variables:
    a. Items in the stack (student name, ID, phone #, and address)
    b. Number of items in the stack
    c. Reference to the top item
2. Your class has the following member methods in addition to its constructor(s):
    a. push to add new item to the stack.
    b. pop to remove from stack the item that was added to the stack most recently.
    c. size to get the number of Items in the stack.
    d. empty to check whether the stack is empty.
    e. full to check whether the stack is full.
    f. top to get a copy of the item that was added to the stack most recently.
3. Your program provides a menu-based user interface that allows the user to use/test all your class member methods.

Deliverables:

A compressed file of your project that must contain the whole project folder, which includes:

1. Class header file that contains the definition of your class including prototypes of your class member methods/functions.
2. Class C++ file that contains the definitions of your class member methods/functions.
3. Your user-interface program in C++.

Hints:

1. You must follow the Google guide when you implement your solution. Failing to follow this coding guidelines will cost you **10%** of your grade. Here is the link of the Google guide: https://google.github.io/styleguide/cppguide.html#Scoping.
2. Comments are essential to keeping our code readable. The file-level comment should broadly describe the contents of the file and must start with the author line (your name. student ID, and email) followed by the assignment #, title, description.

    You must have detailed comments/documentation within each function or part of your program and not at the file level. Be consistent with how you comment and what style you use where. Not enough comments/documentation will cost you another **5%** of your grade.

Late Policy:

Assignments received after due date will be marked down 10%, and 2% more per day that assignment is late. No credit will be given for work turned in more than one week late.