Programming Assignment #2
CS 163 Data Structures

Submit your assignment to the D2L Dropbox (sign on via d2l.pdx.edu)

Goal: The purpose of the second programming assignment is to experience stacks, queues, and new data structures. The data structures for this assignment are a linear linked list of arrays and a circular linked list.

Problem Statement:
As I sit here at my grid of computers, I realize that if I were to venture out I would need to be socially distant to others (shouldn't we be saying "physically" distant?!). Instead of standing in a line with x's indicating where I should stand, I would prefer an App to alert me when it is time for "my turn". Such an application could be used by many different kinds of systems – as an alternative to making appointments.

For our assignment, a queue would be great to keep track of everyone in line (First In, First Out). Then, a stack would hold the alerts as they are sent out from the store (or restaurant, doctor's office, etc.) to let people know when it is their turn. The stack is used to hold the alerts because if a person doesn't respond within a given amount of time, then the next person should be alerted and they will get priority.

Programming: There are two ADTs in this program (Queue and Stack ADT). This means each needs to be implemented as a separate class. You can put the class interfaces in either one .h file or two .h files.

1. The information kept in the queue of people "in line"
   a. Person's full name
   b. Contact information (e.g., cell phone number)
2. The information kept in the stack of people alerted as to their "turn"
   a. Person's full name
   b. Information about when the alert took place

- Please keep in mind that because we are creating ADTs, the **user** of the program must not be aware that stacks and queues are being used. However, the client program knows that they are using these ADTs.

Programming – Data Structures:

- The queue should be implemented using a circular linked list (lab 3) where the rear pointer points to the last person in line, and rear->next points to the first. You must implement **enqueue, dequeue, peek, and display.**
- The stack should be implemented using a linear linked list of arrays (lab 3), where each element in the array is a person being alerted. The array must be dynamically allocated and each array must be of the same size. I recommend that the arrays be relatively small (e.g., 5) so that you can properly test the code. Stack ADT functions should include **push, pop, peek, and display.**
- You should support complete implementations of the Stack and Queue ADT. Make sure to thoroughly test each of the stack and queue functions from the client program!
- Remember the idea of your client program is to test out your ADT functionality so that a real application can be developed off site by another team of Software Engineers based on the ADT public functions. You want to make sure that your stack and queue ADTs perform well!

<u>Things you should know...as part of your program:</u>

1) **Do not use statically allocated arrays in your classes or structures used by the ADT.**
2) All data members in a class must be private
3) Never perform input operations from your class in CS163
4) Global variables are not allowed in CS163
5) **Do not use the String class! (use arrays of characters instead!) You may use the cstring library.**
6) Use modular design, separating the .h files from the .cpp files. Remember, .h files should contain the class header and any necessary prototypes. The .cpp files should contain function definitions. **Never "#include" .cpp files!**