Programming Assignment #2
CS 260 Data Structures

ALWAYS make a backup of your programs before using tar
Double check that the arguments specified with for tar are correct!
***Late penalties** will apply to all submissions incorrectly archived/emailed*

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

Problem Statement:
Have you been to a popular restaurant recently? Many are using cell phones to notify groups when their table is ready. But, why not an App? It could let groups know their place in line and be a sales tool for the restaurant to forward coupons and other promotions to increase business.

Let's say you had a friend opening up a new restaurant business and they are looking for help with the software support. They have asked you to be part of the project. You have decided that a queue would be a great ADT for keeping track of who is in line, and how long it will take before they get a table. You have also decided that a stack would be great for reaching out to the customers who most recently frequented the restaurant to provide promotional materials so that they will return before forgetting about this great new restaurant.

Programming: There are two ADTs in this program (Queue and Stack ADT).

1. The queue will represent the people in line waiting for a table.
   a. The queue is ordered based on the order in which the groups arrive. The data should include
      (a) name of the group,
      (b) number of people in the group
      (c) if there is anyone who needs special seating
         If so, include the information about any requirements (e.g., wheel chair or high chair).
      (e) if the group would like to receive coupons and other promotional materials

If so, include the contact name (full name) and email address

2. The stack then will represent people interested in receiving promotional material, which should include their full name and email address

- 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. You should support complete implementations of the Stack and Queue ADT. Make sure to thoroughly test each of the stack and queue functions!

Programming – Data Structures:

- The queue should be implemented using a circular linked list of people, where the rear pointer points to the most recent group to arrive at the restaurant, and rear->next points to the first group to arrive that hasn't yet been seated. You must implement enqueue, dequeue, peek, and display. Enqueue would be used when people arrive. Dequeue would be used when groups get seated. Display should include the number in line, so people will know how long of a wait they have.

- The stack should be implemented using an array, where each element in the array is a person interested in promotional materials. The array must be dynamically allocated. Have the constructor receive the size of the array from the client program. Stack ADT functions should include push, pop, peek, and display. When a group gets seated at a table, it is time to push their information onto the stack if they are interested in promotional materials.

  Then, when the manager will contact those people using pop when there are a few new promotional events (e.g., a pumpkin carving party). As you know, using a stack will allow the manager to contact the most recent people who have frequented the restaurant. The manager doesn't want to entirely lose this information, so they would like it saved in an external data file after being removed from the stack.

<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 data structure class in CS260
4) Global variables are not allowed in CS260
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!**