Programming Assignment #2
CS 163 Data Structures

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

Remember to turn in a design writeup with your program

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: Have you noticed that the food carts in Portland are getting really popular. When I want to get lunch, I have to plan it out otherwise I might be standing in the rain for a good 10 to 15 minutes, waiting for my food. And, it isn't like I can look out my window (no, I do not have a window to the outside…) to see how the line is going.

In our second program, we have decided to make an "app" to allow for potential food cart customer's to determine how long the line is (represented as a queue ADT) and keep track of the time it takes to be serviced (represented as a stack)

Programming – ADTs: There are three ADTs in this program (Service, Queue and Stack ADT).

- The Queue ADT will keep a list of customers waiting in line for their food to be cooked. It should include the customer's name, their order and their arrival time. Customers are enqueued when they arrive (or when an order is phoned in). The information is dequeued when they pay and get their food. To find out if there is someone in line, we can peek into the queue.
- The Stack ADT will keep the statistics, such as the length of time, a customer has had to wait in the worst case, along with the average wait time total. If the current statistics at the top of the stack are worse than the most recent customer's wait time, then the stack only the average time on the top of stack is changed (popped, modified and pushed back). If the most recent customer that was dequeued waited more time, then new statistics are pushed onto the stack. At the end of the day the manager can find out what the worst situations were all day long.
- The Service ADT will have a Queue for the orders waiting, a Stack for the statistics, and the number of people served. The Service ADT will use the

enqueue, dequeue, peek functions from the Queue, and the push, pop, peek functions from the Stack to perform its job.

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. Make sure to thoroughly test each of the stack and queue functions!

Programming – Data Structures: The stack should be implemented using a linear linked list of arrays that was discussed in class (and Lab 3). Each element of the array will be the latest statistics (e.g., the worst case wait time statistics through the day). Have the constructor of the stack class receive the size of the array. It should not be hard-coded into the ADT. The array of statistics must be dynamically allocated. USE POINTER ARITHMETIC!!!!!

The queue should be implemented using a circular linked list, where the rear pointer points to the most recently added item, and rear->next points to the item at the front of the queue.

Things you should know...as part of your program:

1) Lab #3 is on stacks, Lab #4 is on queues; they use these data structures.
2) Do not use statically allocated arrays in your classes or structures.
3) All data members in a class must be private
4) Never perform input operations from your class in CS163
5) 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!**
7) Use the iostream library for all I/O; do not use stdio.h.
8) Make sure to define a constructor and destructor for your class. Your destructor must deallocate all dynamically allocated memory.
9) Remember that 20% of each program's grade is based on a written discussion of the design. *Take a look at the style sheet which gives instruction on the topics that your write-up needs to cover.*