COSC 140 Intro to Data Structures

4/1/21

In-class objectives

Learn more about data hiding and using STL sequence containers by discussing a simple container adapter and solving a programming problem using it.

Part 1

Review the notes from lass class. Discuss in your groups how you would convert the queue code from the last lecture into a circular, doubly linked list container.

Discuss and briefly implement the following using either Stack/Queue (lecture notes, more likely) or CardDeck (your prior lab) as a starting point:

* void insertFromFront (int &); // inserts an element of type int in the front

* void insertFromBack (int &); // inserts an element of type int at the back

* void deleteFront ();
*void deleteBack ();

* bool isEmpty();

* ~cdl_list (); // question: is this even needed?

Part II: Moving around your new data structure

To be doubly, you should be able to move forward and backward. To be circular, the first element should consider the last element its preceding neighbor and the last element should point to the first element as its "next" neighbor.

Discuss adding the ability to move around your cdl_list described above by adding the following "iterator"-like functionality. You should create an internal, private data member called "position" (or similar) that indicates the node currently visited by the "iterator".

What are the special cases you need to consider and why?

HINT: This exercise is much easier if you use a container that supports random access (vector, deque). Discuss why and/or how? Since we are encouraging data hiding in this exercise, any viable approach will receive full credit.

Discuss the implementations of following functionality (or their equivalent):

- void reset(); // returns to the front of the list
- void next(); // moves to the next element of list
- T currentValue(); // returns current value at top of the list
- void delete(); // removes an element from the list

Part III: Putting it all together

In ancient times, a fair princess Buttercup had many suitors. Thinking her true love dead, she decides to use the following procedure to choose the knight she will marry. First, all suitors are lined up and given a number from 1 ... *n*. Then, because of a strange fascination with the number three, she would count to three and remove the third suitor she considers that round from contention. This process would repeat, returning to the front of the list once all suitors were considered. For example here is the results of the process with $n = 6$:

123456
12456
1245
125

15 1

(3, you're not the one for me) (6 ... sorry)(4, no more)(2, I don't choose you)(5, at least you're still alive)

Luckily, Buttercup's true love Wesley was not killed but rather became a protégé of the Dread Coder SJE. As part of the Dread Coder's crew, you have been asked to help Wesley determine which position in line he should stand to win Buttercup's hand.

1.) How could you use the circular doubly linked list, and specifically Part II, to perform the simulation and display what spot Wesley should stand in for any arbitrary *n* number of suitors. Add pseudocode if you have time

2.) Write brief text for your submission on what data hiding is (in your own words) and how it was beneficial. Are there any special advantages?