CS 11 Data Structures and Algorithms

Assignment 10: Linked Lists 2

Skip to Main Content

Assignment 10.1 [75 points]

No documentation (commenting) is required for this assignment.

This assignment is based on an assignment from "Data Structures and Other Objects Using C++" by Michael Main and Walter Savitch.

Add attach(), remove_current(), and the big-3 to the sequence class from the last assignment. All requirements from the previous assignment are still in force. Your score on this assignment will take into consideration your work on both the previous assignment and this assignment.

Specification and Implementation

Please refer to the specification and implementation information from the last assignment.

To simplify your code, you should write two private helper functions named "copy()" and "clear()". Then your copy constructor will simply call "copy()", your destructor will simply call "clear()", and your assignment operator will include calls to both.

Make sure that when you copy your list the four pointer data members point to the nodes in the new list that correspond to the nodes that they pointed to in the original list.

When writing the attach() and remove_current() functions, I prefer to start with the most restrictive cases, and then the last "else" is the general case. You can see that my insert() function (from last assignment) follows this pattern: (1) inserting into an empty list, (2) inserting at the front of a list with at least one item, (3) inserting anywhere else. For attach() I also have 3 cases: (1) attaching to an empty list, (2) attaching at the end of a list with at least one item, (3) attaching anywhere else. For remove_current() I start with an assert statement to make sure there is a current item (since that is a precondition), then the cases are (1) remove from a list that has exactly one item, (2) remove the first item in the list, (3) remove any other item. (This third case sort of has another case embedded in it: if the item being removed is the last item I have to reset tailptr and set precursor to NULL.)

It is very hard to get this working for all conceivable cases, and also very hard to test it exhaustively by thinking of every conceivable case. Here is a **client program** that tests the class pretty exhaustively. (Even this program does not test every possible case.)

This is the client program that we will be testing your class with. If you can't get every case to work but most of them work (as evidenced by the point total that client program reports) you can still get a pretty good score on the assignment.

Submit Your Work

Name your source code files sequence.cpp and sequence.h. No need to paste your output. Use the Assignment Submission link to submit the two files. When you submit your assignment there will be a text field in which you can add a note to me (called a "comment", but don't confuse it with a C++ comment). In this "comments" section of the submission page let me know whether the programs and classes work as required.