Data Structures (EECS 2080C) – Summer 2020 – Lab 4

Topics covered: Linked Lists

Lab due: ***Monday, June 8 at 8:00 AM***

Objective:

The objective of this Lab is to explore creating and using an unordered linked list.

Task 1: Create an unordered linked list class.

1. Create a new project. You can name this whatever you like.
2. Design and implement an unordered linked list of strings as described in class.
    a. The following methods fully implemented.
        i. AddItemFront – adds an item to the front of the list
        ii. AddItemEnd – adds an item to the end of the list
        iii. GetItem – searches the list for the given item. If found, it removes it from the list and returns it. If not found, it returns a null pointer.
        iv. GetItemFront – removes and returns the item in the front of the list.
        v. IsInlist – returns a bool indicating if the given item is in the list.
        vi. IsEmpty – returns a bool indicating if the list is empty.
        vii. Size – returns an int indicating the number of items in the list.
        viii. SeeNext – returns the item without removing it from the list at a given location in the list. The class will maintain the next location and will start at the first item in the list. When it gets to the last item in the list, it will return a null pointer after it gets past the last item. If the list is empty, this will throw an error. 2 calls to SeeNext will return the 2 items next to each other in the list unless SeeAt or Reset is called in between the 2 calls (or the first call returns the last item in the list).
        ix. SeeAt – Finds an item at a location in the list (int passed in from user), and returns the item without removing it. If the location passed by the user is past the end of the list, this will throw an error. This will set the location used by SeeNext to point at the item after the item returned.
        x. Reset – resets the location that the SeeNext function uses to point at the first item in the list.
        xi. Print – This will display the contents of the list using cout.

    b. All items passed to or from the methods should be done so via a pointer rather than by value.

    c. Make sure you don't have any memory leaks.

Task 2: Create a test program that has a menu allowing you to test each of the functions in your linked last (Task 1).

1. This should present the user with a choice of public member functions of your linked list class and ask which the user would like to try.
2. When the user selects a member function, the program will prompt the user for any required information such as AddItemFront requiring a string or SeeAt requiring an int.

3. Test your program. Include a screen shot of some of this testing in your lab report.

Complete this before moving on to task 3.

Task 3: Create implementations of a Stack and Queue using a linked list

1. Modify the stack and queue classes from Lab 3 to inherit from a Linked List. This should be able to use most, if not all of the built-in Linked List functions to support the Stack and Queue functionality. You should have a total of 3 classes at the end of this task.
2. Modify your test program from task 2 to include an option that calls these new classes.
3. Test your modifications. Include in your lab report a screen shot showing the results of these new classes.

Lab Submission:

1. Write a lab report including the following information:
   a. A description of the objectives/concepts explored in this assignment including why you think they are important to this course and a career in Engineering.
   b. The sections from each task indicated to be included in the lab report.
2. Include all source code from all tasks, input and output files (if any), and any special instructions to compile and run those programs.
3. Package all files in a single zip folder and submit the file to blackboard.

Lab Grading:

1. 20% - Lab attendance
2. 25% - Task 1 has been correctly implemented and meets all requirements.
3. 15% - Task 2 has been correctly implemented and meets all requirements.
4. 20% - Task 3 has been correctly implemented and meets all requirements.
5. 20% - Lab report contains all required information and is well written.

If program fails to compile, 0% will be given for that Task. Any memory leaks will result in a 5% grade deduction.