Data Structures (2028C) -- Spring 2025 – Lab 7

Topics covered: Linked Lists

Lab due: ***Sunday, Mar 23 at 11:55PM for Monday Section***
Tuesday, Mar 25 at 11:55 PM for Wednesday Section

Objective:

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

Task 1: Create an ordered double linked list class.

1. Create a new project. You can name this whatever you like.
2. Design and implement an ordered double linked list class as described in class. This class should be a template. The template is expected to have overloaded the >, < and == operators (meaning this class can only use >, <, and == when looking at the data stored in a node).
    a. The class should have the following methods fully implemented.
        i. Constructor
        ii. AddItem – adds an item from 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. IsInlist – returns a bool indicating if the given item is in the list.
        v. IsEmpty – returns a bool indicating if the list is empty.
        vi. Size – returns an int indicating the number of items in the list.
        vii. 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 or the list is modifed between the two calls).
        viii. SeePrev – Same as SeeNext except in the other direction.
        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 variable that the SeeNext function uses so the next call to SeeNext will return the first item in the list.
        xi. Destructor – make sure you remove all items to avoid memory leaks
    b. All items passed to or from the class should be done so via a pointer rather than by value.
    c. Make sure you don't have any memory leaks.

Complete this before moving on to task 2.

Task 2: Create a class to be used as the item stored in the list. This class will be "Part". It should have the following members at a minimum.

1. Private:
    a. SKU – stock keeping number
    b. Description
    c. Price
    d. UOM - Unit of measure (is it sold by the foot, pound, each, etc…)
    e. QuantityOnHand
    f. LeadTime (number of days it takes to order if there aren't any on hand to sell)
2. Public:
    a. Constructor – This accepts parameters for SKU, Description, Price, and UOM. It has an optional parameter for QuantityOnHand which if missing is set to 0.
    b. GetPartInfo – this returns a string containing both SKU concatenated with Description
    c. GetPrice
    d. InStock – returns a bool indicating if the Quantity On Hand is >0.
    e. Available – accepts a date. This returns true if QuantityOnHand is >0. If it isn't, it calculates if the desired date is > LeadTime (added to current date) and returns the results of that comparison
    f. Overloads of the >, < and == operators that compare the SKU member of two Part instances. This is required to allow this class to be used in the Linked List class from task 1.

Complete this before moving on to task 3.

Task 3: Create a test program that has a menu allowing you to test each of the functions in your linked last class (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. For example, the GetItem function only requires a SKU as that is what is being compared but the AddItem requires the user to enter a SKU, Description, Price, UOM and optionally a QuantityOnHand.
3. Test your program. Include a screen shot of some of this testing in your lab report.

Complete this before moving on to task 4.

Task 4: Create a visualization of your list using ASCII art.

1. Modify your Part class (task 2) to include a display method that will output to the screen the contents of the class
    a. This method should write directly to the screen.
    b. This method does not need to display all members of the item, just key members (explain your choice of key members in your lab report).
    c. This method should reuse members of the class as much as possible.
2. Modify your linked list class (task 1) to add a new public method to display the list.
    a. This method should write directly to the screen.

   b. This method should walk through the list and display every item in the list on the screen. The format of the output is up to you.
   c. To display the contents of the item in the list, it should call the display member of the item (created in step 1 of this task).
   d. This method should reuse members of the class as much as possible.
   e. This method should not change the results of SeeNext method meaning it should not have a different value for the internal variable used by SeeNext after this method completes.
3. Modify your test program from task 3 to include an option that calls this new method.
4. Test your modifications. Include in your lab report a screen shot showing the results of this new method with at least 4 items in your list.

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 CS and/or 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 Canvas.

Lab Grading:

1. 20% - Lab attendance
2. 25% - Task 1 has been correctly implemented and meets all requirements.
3. 10% - Task 2 has been correctly implemented and meets all requirements.
4. 10% - Task 3 has been correctly implemented and meets all requirements.
5. 15% - Task 4 has been correctly implemented and meets all requirements.
6. 20% - Lab report contains all required information and is well written.

If program fails to compile, 0% will be given for that Task.