CS151 – Lab 6

Make a Linked List

The last lecture discussed a great deal about linked lists, but leaves out a great many of the implementation details. In this lab, you will actually go about implementing your own version of the grocery list. This means that you will have to implement the functions given from the lecture with the slight modification of using a string in addition to a number.

You can choose to implement your linked list using the template in chapter 17 of the Gaddis Book or Chapter 13 of the Savitch Book. Gaddis uses an object oriented structure whereas Savitch uses a procedural structure. The object oriented structure requires a constructor and a destructor, but is more reusable since you can take advantage of separate compilation.

**The choice is up to you**, but your implementation needs to have the following constructs:

# Node Structure

Implement a node structure to reflect the grocery list. Make sure that you include this before the main part of your program.

**Hint:** Be sure to include cstddef so that you can be certain that the NULL pointer works.

# Head of the List

Create a pointer to head and a first item in your linked list. You can do this with either a constructor or procedurally. You may want to use cout to test whether or not you have done this correctly.

# Insert a Node

Now that you have a linked list with a single item, create a function to insert a node at the head of the list or the tail of the list. Put some driver code in your main program to test this function[[1]](#footnote-1).

# Print the List

Using all these cout statements in main can be a bit trying. Implement a function to print the contents of your linked list.

**Hint:** This function will be similar to the sequential search routine, but won’t use a search value. Think about how you will step through the list.

# Locate a Value in the List

Implement a function to locate a value in your linked list. You may use sequential search here. You may wish to have it search by the item name as opposed to the quantity, since quantity is more likely to be duplicated.

# Insert a Node in the Middle

Implement a function to insert a node in the middle or at the end of your list and test it.

**Note:** Here, you will want to use results of your sequential search function to locate the node where you want to insert the item.

# Delete a Node

Implement a function to delete a node from your linked list. Use the results from your locate value function to do this.

# A Driver Program

Now that you have all the functions necessary to have a linked list of your very own, you will want to create a driver program that tests all of your functions. Once you are finished with this driver program, submit your program via Laulima.

1. Gaddis shows how to put an item at the end of the list. [↑](#footnote-ref-1)