CS20 – Section 093 – Fall 2019
Programming Assignment 2 – Due Wednesday September 18 at 11:59PM
Version 2 – changed 2019-09-10

Linked list with better programming techniques (60 points)

In this assignment, you'll make significant changes to your linked list coding from Assignment 1.

Step 1

Convert your LLnode struct from Assignment 1 into a header file. Any code module that refers to LLnode will need to contain a #include for this header file.

Create a linked list class named LL. Put into the same header file as your struct. It will contain private data members:

- an LLnode struct variable
- a linked list header, which is just a pointer variable of type LLnode for LLnode variables

You've already coded six linked list processing functions – push_front, push_back, list_length, retrieve_front, retrieve_back, and display_list. Convert these functions into public member functions of the class. In assignment 1, each function was passed a linked list header to operate on. Each function knew which linked list to operate on because the list's header was explicitly passed as a parameter. Now that you're using a class, it's the class object used in the call that makes this distinction, so passing the header is no longer necessary.

The constructor for the LL class should set the linked list header to nullptr. There is no destructor.

Remember that a good programmer always plans his or her debugging. If you've coded everything mentioned so far, you've reached a good point to debug. Use the main # 1 test program (see the assignment in Canvas) to make sure everything works before you continue.

CHANGE: Normally, the class member functions would be placed into their own .cpp file, meaning that you would have a total of three source files: the header file, the .cpp for the class member functions, and a .cpp for main. However, in step 2, you're going to templatize the class, and a requirement for templatized class functions is that they be coded in-line in the class definition. So, for this assignment, code your member functions in-line within the header file. You'll have only two course files: the header file, and the .cpp for main.

I'm looking for evidence of debugging cout's in your program. Leave them in but comment them out so I can see your approach to debugging.

Step 2

In assignment 1, the data in each node was of type string. Now, using a class template, change your struct and class to use any variable type. Main # 2 has been set up for debugging this step.

Step 3

Next, add member functions that perform the following functions:

void destroy_list ()
deletes each node in the list, and resets the header to nullptr

bool search_list (key value)
searches the list for a node with the given key. Returns true if found, false if not.

bool delete_node (key value)
deletes the node which contains the given key. If there is more than one node with the same key, delete_node deletes the first occurrence. Returns true if delete successful, false if the node was not found.

Main # 3 contains a test program for testing step 3.