COMSC 165

Spring 2020

Programming Assignment 10

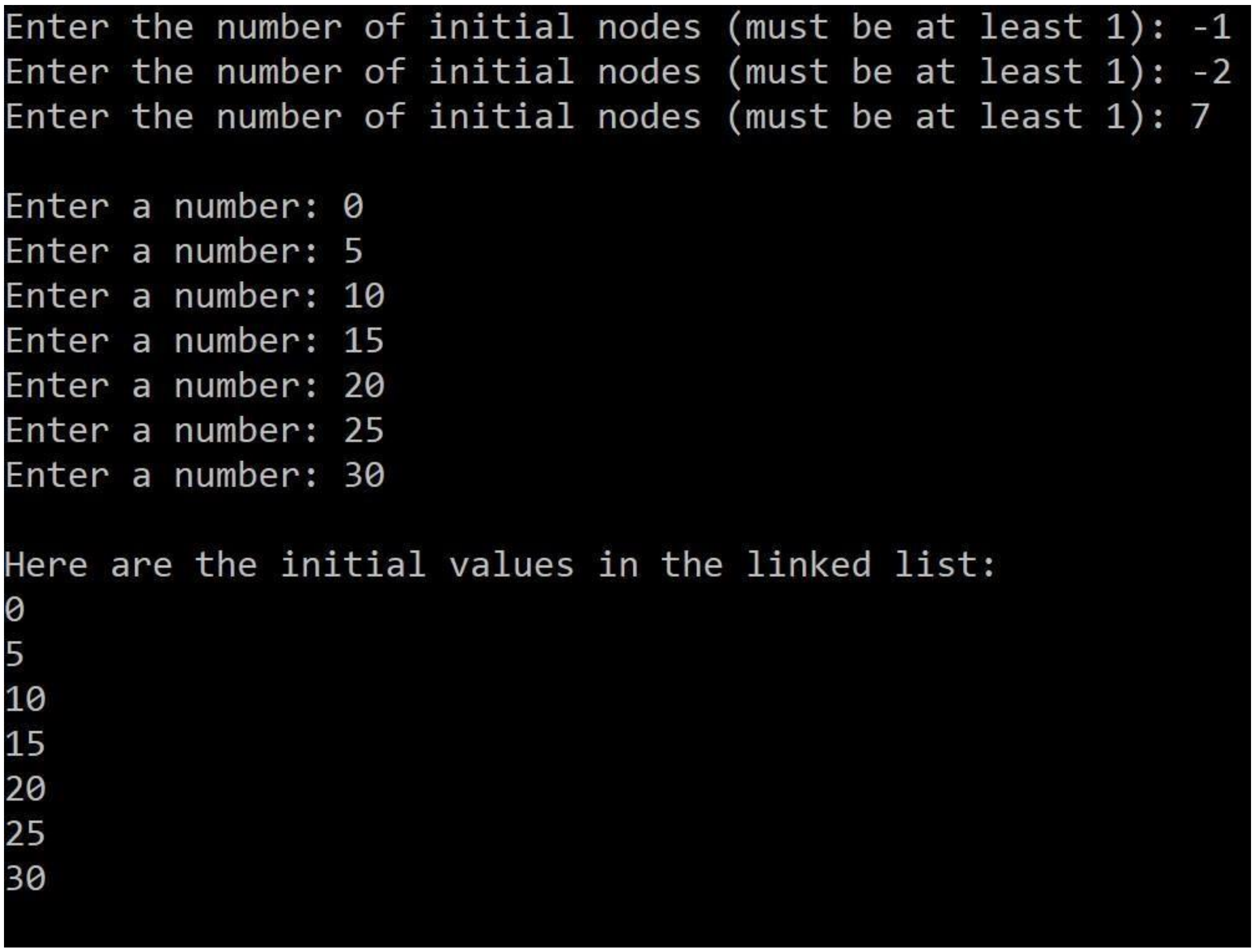
Worth 20 points (2% of your grade)

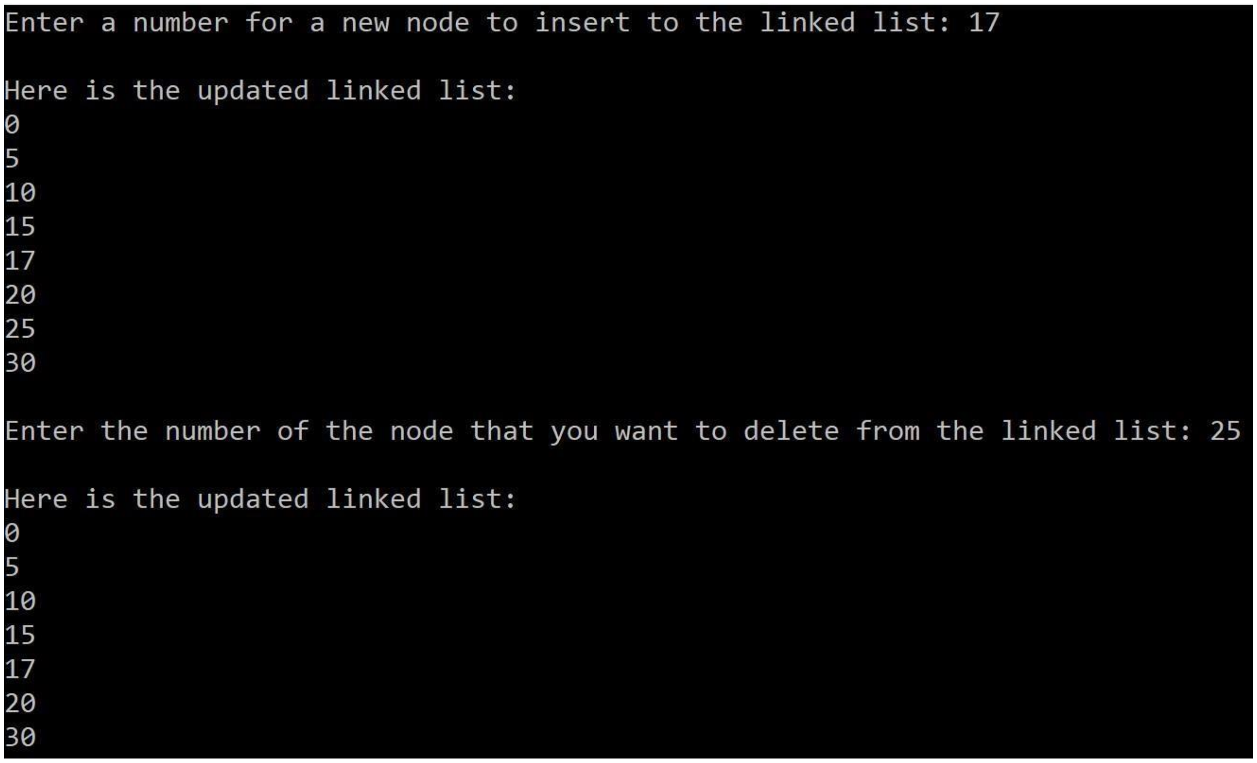
DUE: Wednesday, 7/22/20 by 11:59 P.M. on Canvas

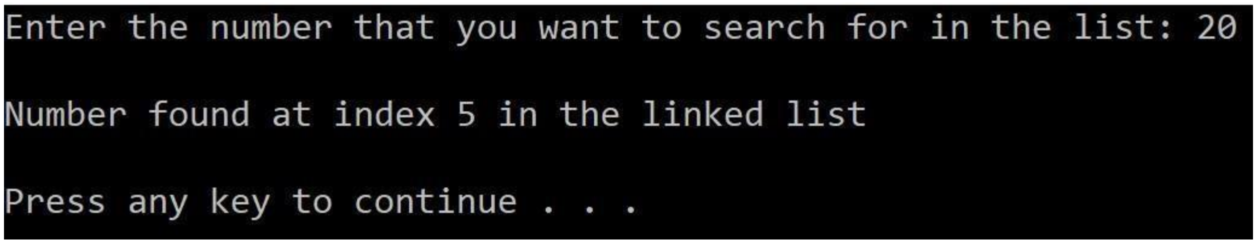
NOTE: Your submission for this assignment should be a single **.cpp** file and a single **.pdf** file. The following naming convention should be used for naming your files: **firstname_lastname_165_assign10.cpp and firstname_lastname_165_assign10.pdf**. The pdf file that you submit should contain the screenshots of your sample runs of the program (see below). For example, if your first name is "James" and your last name is "Smith", then your files should be named James_Smith_165_assign10.cpp and James_Smith_165_assign10.pdf.

COMMENTS (7.5% of programming assignment grade): Your program should have at least **ten (10)** different detailed comments explaining the different parts of your program. Each individual comment should be, at a minimum, a sentence explaining a particular part of your code. You should make each comment as detailed as necessary to fully explain your code. You should also number each of your comments (i.e., comment 1, comment 2, etc.).

SAMPLE RUNS (7.5% of programming assignment grade): You should submit screenshots of at least **five (5)** different sample runs of your program. Each sample run needs to use different user inputs, and your sample runs should **NOT** be the same as the sample runs that are used in this write-up for the assignment. You should also number each of your sample runs (i.e., sample run 1, sample run 2, etc.). Each of your sample runs should be similar to this format:


```
Enter the number of initial nodes (must be at least 1): -1
Enter the number of initial nodes (must be at least 1): -2
Enter the number of initial nodes (must be at least 1): 7

Enter a number: 0
Enter a number: 5
Enter a number: 10
Enter a number: 15
Enter a number: 20
Enter a number: 25
Enter a number: 30

Here are the initial values in the linked list:
0
5
10
15
20
25
30
```


```
Enter a number for a new node to insert to the linked list: 17

Here is the updated linked list:
0
5
10
15
17
20
25
30

Enter the number of the node that you want to delete from the linked list: 25

Here is the updated linked list:
0
5
10
15
17
20
30
```


```
Enter the number that you want to search for in the list: 20

Number found at index 5 in the linked list

Press any key to continue . . .
```

For your tenth programming assignment you will be implementing a program that uses a linked list. You will be implementing the following **structure** and functions:

```
struct LinkedList

{

   int value;

  LinkedList *next;

};
```

append_node: Appends a new node (created using the **new** statement) to the end of the linked list.

insert_node: Inserts a new node (created using the **new** statement) at the specified location in the list.

delete_node: Deletes a node from the specified location in the list.

search_node: Searches the list for a value.

destroy_list: At the end of the program, this function loops through the entire list and destroys the list by using the **delete** statement on each dynamically allocated linked list node (to avoid a **memory leak**).

Note that with a singly linked list, you need to maintain a **head** pointer (pointer to the beginning of the list). Typically, a tail pointer (pointer to the end of the list) is not maintained in a singly linked list (because you can only iterate from the beginning to the end), although it can be.

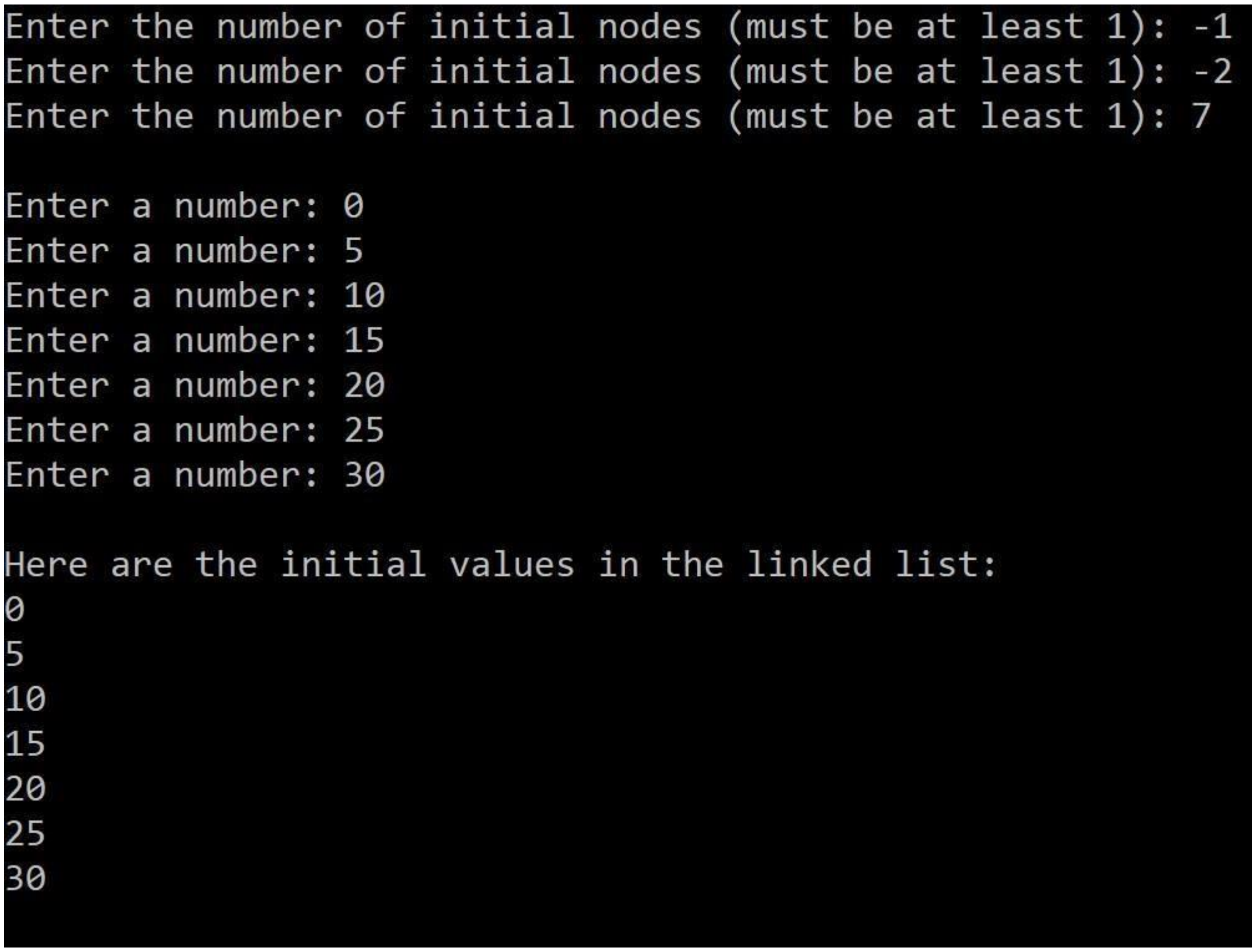
NOTE: Sometimes a function will need to change the head pointer (i.e., change what the head pointer is pointing at), and this change needs to hold back in main. We've seen that by default, which you pass an address to a pointer parameter, it is passed by **VALUE**, meaning that we make a **COPY** of the address, and we have two pointers that are both pointing at the address. We can also pass an address to a pointer parameter by **REFERENCE**.

To pass a pointer by reference in this assignment use the type LinkedList *&. This allows a change made to the head pointer inside of the function to hold back in main.

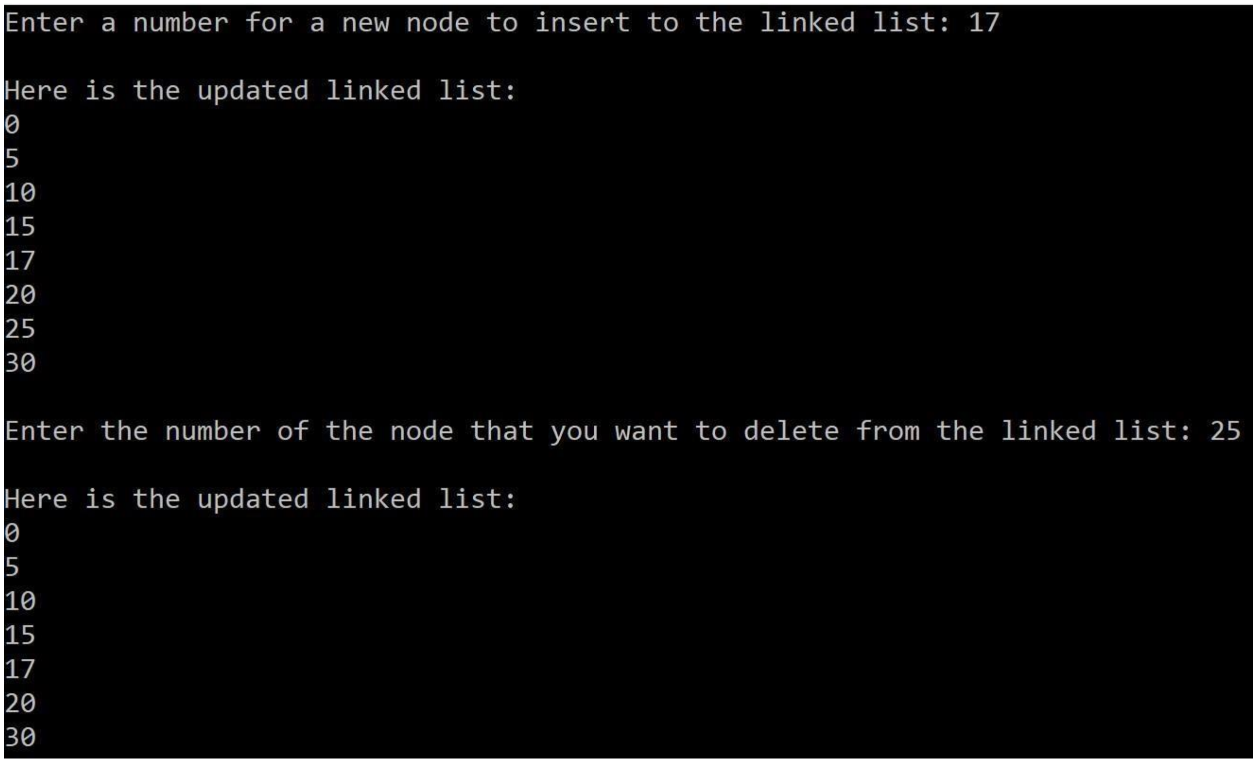
LinkedList * will pass the pointer by value (don't do this if the head pointer needs to change though!!!)

const LinkedList *& will pass the pointer by constant reference (don't do this if the head pointer needs to change though!!!)

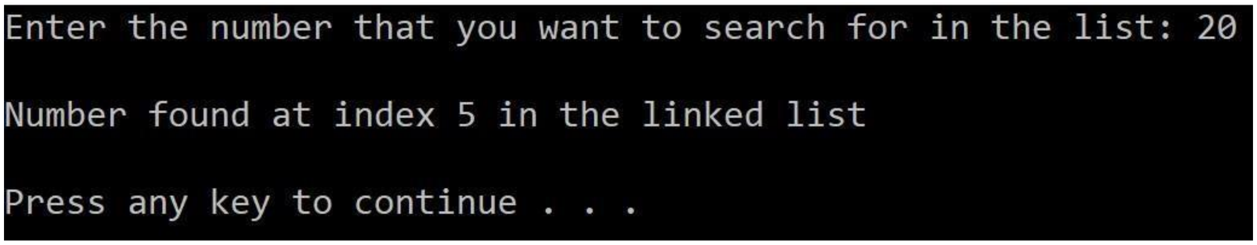
Sample run:



```
Enter the number of initial nodes (must be at least 1): -1
Enter the number of initial nodes (must be at least 1): -2
Enter the number of initial nodes (must be at least 1): 7

Enter a number: 0
Enter a number: 5
Enter a number: 10
Enter a number: 15
Enter a number: 20
Enter a number: 25
Enter a number: 30

Here are the initial values in the linked list:
0
5
10
15
20
25
30
```



```
Enter a number for a new node to insert to the linked list: 17

Here is the updated linked list:
0
5
10
15
17
20
25
30

Enter the number of the node that you want to delete from the linked list: 25

Here is the updated linked list:
0
5
10
15
17
20
30
```



```
Enter the number that you want to search for in the list: 20

Number found at index 5 in the linked list

Press any key to continue . . .
```