Doubly Linked List Template

1. Authors of the program:

Fnu Manju Muralidharan Priya for DLL template
Dat Truong for added function.

2. File Structure of the project submitted:

Main.cpp and readme.pdf

3. Software that can be used to run the files / Installation Guide

Clion or any C++ compiler that is version 11 and above.

4. How to get started?

a. Describe the input required from the user

The program required the user to input a character that appears on the menu. If the user inputs an integer, then it will ask for a character option. If user presses any character that is not on the menu option, then it will close the program.

b. Describe what function is called by each input and briefly describe each function.

There are 14 options ranging from a to n:

- Option A: will call the constructor of the linked list then, asking user to input the string value then integer value of the node to add to the new list.
  - CAUTION: if the input for integer is not integer the program will enter an infinite loop.
  - Function is O(1) since it is assigning the value.

- Option B: will call destroyList() to delete the list.
  - CAUTION: if the input for integer is not integer the program will enter an infinite loop.
  - Function destroyList() is O(1) since it assign nullptr to head and tail.

- Option C: it will call the prepend() function. It will ask the user to enter the string value then integer value for the node to add to the front of the list, then it will display the list before and after adding the node.
  - CAUTION: if the input for integer is not integer the program will enter an infinite loop. Or if the list is not previously added, the program will crash.
  - Function prepend() is O(1) since it is assigning node to the head of the list

- Option D: it will call the append() function and ask the user to enter the string value then integer value for the node to add to the end of the list, then it will display the list before and after adding the node.
  - CAUTION: if the input for integer is not integer the program will enter an infinite loop. Or if the list is not previously added, the program will crash.
  - Function Append() is O(1) since it is assigning node to the tail of the list

- Option E: it will call the insert() function and ask the user to enter the string value then integer value for the node, then it will ask for the index that user wants to add to the list. There will be a validation for the index input. At the end, it will display the list before and after adding the node.
  - CAUTION: if the input for integer is not integer the program will enter an infinite loop. Or if the list is not previously added, the program will crash.
  - Function insert() is O(n) since it loop through the whole list while getting the index of previous node.

- Option F: it will call the deleteAtHead() function to delete the head node of the list. At the end, it will display the list before and after deleting the node. If the list is fully delete, it will destroy the list by calling destroyList() function. Then quit the program.
  - CAUTION: if the list is not previously added, the program will crash
  - Function deleteAtHead() is O(1) since it delete right at the head of the list

- Option G: it will call the deleteAtTail() function to delete the head node of the list. At the end, it will display the list before and after deleting the node. If the list is fully delete, it will destroy the list by calling destroyList() function.
  - CAUTION: if the list is not previously added, the program will crash
  - Function deleteAtTail() is O(1) since it delete right at the tail of the list (because it is double linked list)

- Option H: It will call deleteAtIndex() function. It will ask the user to enter an index integer to delete. It will also validate the index value if it is positive and is within the length of the current list. At the end, it will display the list before and after deleting the node. If the list is fully delete, it will destroy the list by calling destroyList() function.
  - CAUTION: if the list is not previously added, the program will crash
  - Function deleteAtIndex() is O(n) because it need to loop through the list to find the index of prevNode.

- Option J: it will call reverseList() function, it will reverse the list then display the before and after the call.
  - CAUTION: if the list is not previously added, the program will crash
  - Function reverseList() is O(n) because it have to loop through the whole list to reverse the list.

- Option K: it will call sortList() function, it will sort the list in ascending order by the integer value of the node using bubble sort algorithm. It then displays the before and after sort.
    - CAUTION: if the list is not previously added, the program will crash
    - Function sortList() is $O(n^2)$ since it use bubble sorting with nested while loop

- Option L: it will call countMultiples() function and ask user to enter the string and integer value respectively and then display the Node that user want to ask and display the appearance time.
    - CAUTION: if the list is not previously added, the program will crash or if user enters the wrong data type for the integer value, then it will enter infinite Loop.
    - Function countMultiples() is O(n) since it goes through every element to count the appearance.

- Option M: it will call the removeMultiples() function. This will loop through the list to remove any node that has more than 1 appearance and delete the one after keeping the original node.
    - CAUTION: if the list is not previously added, the program will crash.
    - Function removeMultiples() is O(n2) since it loop through the list twice to check then remove.

- Option N: it will call the evenOddSplit() function that will split the list into two lists, one with even index nodes and the other one with odd index nodes. It then displays the two lists while deleting the original list and then it will quit.
    - CAUTION: if the list is not previously added, the program will crash.
    - Function evenOddSplit() is O(n) since it loop through the list once and adding node to the new list along the way.