**Assignment 4 – Report**

Student: Omar Badawy

ID: 900212151

**Linked List & Vectors Assignment**

**Outline:**

1. Explaining the code
   1. .h file
   2. .cpp file (in the order organized in the code)
      1. insertAfter function.
      2. addNode function.
      3. removeNode function.
      4. printList and sumNodes functions.
      5. createList function.
2. Output of the code
   1. Screenshots of the output and use of each function
      1. createList function.
      2. addNode function.
      3. insertAfter function.
      4. printList function.
      5. sumNodes function.
      6. removeNode function.
      7. Terminal

**Explanation of the code**

1. **.h file**

The header file oof this assignment includes private data member, head, which is a pointer to the first node of the list. The class has public member functions, addNode(), removeNode(), printList(), createList(), and sumNodes(). The struct Node defines the structure of each node in the list, including an integer value, count, and a pointer to the next node in the list. The class has a constructor LinkedList and a destructor ~LinkedList() which is called when the object is destroyed to deallocate memory used by the linked list.

1. **.cpp file**
   1. ***InsertAfter function:***

The insertAfter function takes in a vector of integers values, num, and inserted. It finds the first occurrence of the first value (num) in values and inserts the second value (inserted) after it. The function uses a for loop to iterate over each element of values, checks if the current element is equal to num, and if it is, inserts second value (inserted) immediately after it using the insert() method of the vector class. The function is then ready for use as the modified values vector is available for use.

* 1. ***addNode function:***

The addNode function simply takes an integer value as an argument (num). It checks if the head of the linked list is null. If the head is null, it creates a new node with the number given (num) and sets it as the head of the list. If the head is not null, the code iterates through the list until it finds a node with the given number (num) or reaches the end of the list. If it finds a node with the number, it increments its count by one.

* 1. **removeNode function:**

The removeNode function is a function that simply removes a specific value (num) from the linked list given. The code traverses the linked list and checks each node's value until it finds the node with the specific value given (num) or reaches the end of the list. We use various if statements to check for certain things. The first if statement checks if the node is not found, if it wasn’t found, the method returns false. If the node is found and its count is greater than one, the count is then decremented by one. If the count is already one, then the node is removed from the linked list and its memory is deallocated using "delete". If the removed node was the head, then the pointer is updated to point to the next node. If the removed node was not the head, then the previous node's "next" pointer is updated to point to the next node. In the end, the method returns true.

* 1. ***printList and sumNodes functions:***

The function printList is used to print the numbers and the counter (the counter checks how many times the value was repeated in the list) of all the nodes in the linked list. It starts from the head, then traverses the list, and then prints each number and count in a bracket format “(1)” until it reaches the end of the list. The function sumNodes is used to calculate the sum of all the nodes multiplied by their counts in a linked list. Like the printList function, It starts from the head, traverses the list. The extra feature it has is that it calculates the sum by multiplying each node's value by its count and adding it to the total sum until it reaches the end of the list. Then, it returns the total sum.

* 1. ***createList function:***

The function createList takes the vector of integers as input. It goes over each value in the vector using a for loop and then calls the addNode function to add each value to the end of the linked list. That creates a new linked list with the variables that are in the vector.

**Output of the code**

* Creating the list using the createList function which calls the addNode function:

Text

Description automatically generated

* Text

  Description automatically generatedAdding a new value to the list using insertAfter function:
* Printing the linked list using printList function:

Text

Description automatically generated

* Text

  Description automatically generatedPrinting the sum of the list using sumNodes function:
* Text

  Description automatically generatedRemoving a node using removeNode function and displaying the modified sum by calling sumNodes:
* Terminal screenshot: 