

Assignment 4-Report

Samar Ahmed

Computer Science and Engineering Department, The American University in Cairo

CSCE 1101

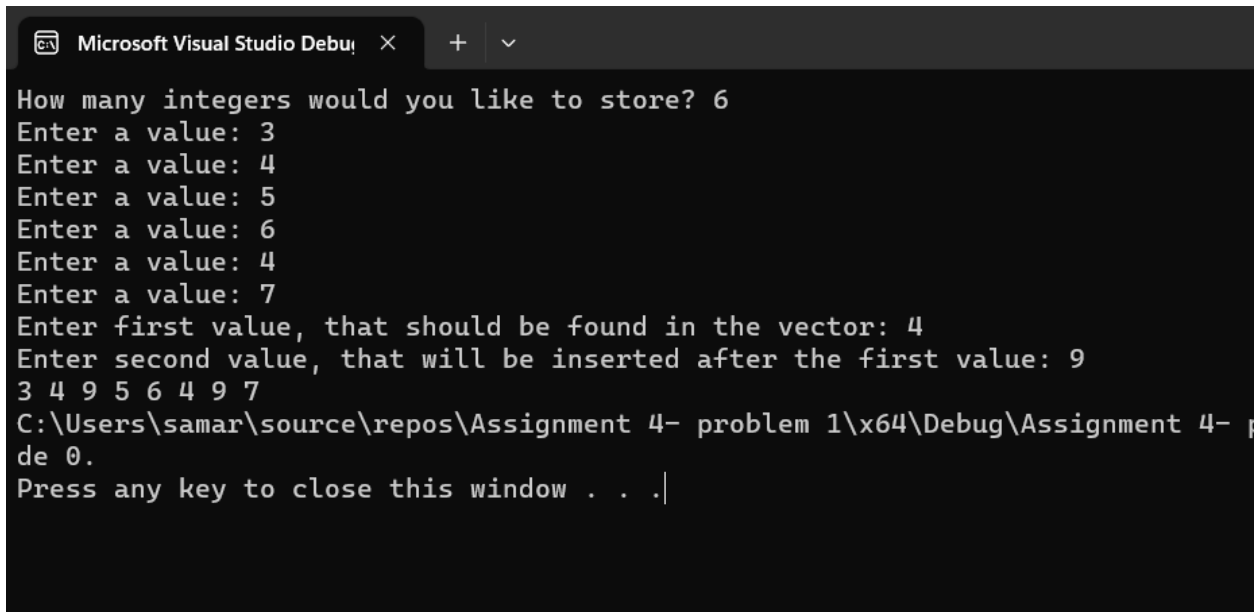
Dr. Howaida Ismaeel

4/19/2023

Problem 1:

For assignment 4, problem one, a C++ program that takes in the size n , stores integers in a vector, receives the first value from the user. It would use that value to check for occurrences using the program, and then asks for a second value from the user to be added after any first value, had to be created.

The output:



```
Microsoft Visual Studio Debug Console
How many integers would you like to store? 6
Enter a value: 3
Enter a value: 4
Enter a value: 5
Enter a value: 6
Enter a value: 4
Enter a value: 7
Enter first value, that should be found in the vector: 4
Enter second value, that will be inserted after the first value: 9
3 4 9 5 6 4 9 7
C:\Users\samar\source\repos\Assignment 4- problem 1\x64\Debug\Assignment 4- p
de 0.
Press any key to close this window . . .|
```

If the first value is located in the vector of all integers, it will be the first scenario for the output. The first value inputted by the user was the number 4 while the second value, 9, was entered after the first value since whenever 4 was located in the vector, 9 would be added after it. As shown in the output every time the initial value occurred, this proceeded to happen.

```
Microsoft Visual Studio Debug Console
How many integers would you like to store? 4
Enter a value: 4
Enter a value: 3
Enter a value: 2
Enter a value: 1
Enter first value, that should be found in the vector: 5
Enter second value, that will be inserted after the first value: 1
4 3 2 1
C:\Users\samar\source\repos\Assignment 4- problem 1\x64\Debug\Assignment
de 0.
Press any key to close this window . . .|
```

In the second scenario, the vector would be outputted as-is if the initial value did not occur. In this scenario, the second value would not be placed anywhere in the vector.

Problem 2:

3 files were created for this problem first the LinkedList header file, the LinkedList source file, and the main source file.

LinkedList.h:

```
1  #include <iostream>
2  #include <vector>
3  #pragma once
4  using namespace std;
5
6  struct Node {
7      int info; // information found inside the link
8      int counter; // to count the number of occurrences of the information
9      Node* next;
10 };
11
12 class LinkedList {
13 private:
14     Node* head;
15
16 public:
17     LinkedList();
18     void addNode(int n); //1.to add the node
19     void removeNode(int n); //2.to remove a node
20     void display(); //3.to print all the elements of the linked list
21     int sumNodes(); //5.sum of all nodes
22
23 };
```

The attributes and functions required for creating a node are contained in the header file. A structure named Node was initially created to hold the value of the numbers and the number of occurrences in the vector, followed by a pointer. The pointer head and every function needed to add, remove, sum, and show a node were then created to the class linked list, along with a function a constructor at the beginning.

LinkedList.cpp:

```
1  #include "LinkedList.h"
2  LinkedList::LinkedList() { head = NULL; } //create the constructor and initialise the head
3  void LinkedList::addNode(int n) { //to search for n and check if it already exists
4      Node* curr = head;
5      while (curr != NULL) {
6          if (curr->info == n) {
7              curr->counter++; //if data is found increase the occurrence by one
8              return;
9          }
10         curr = curr->next;
11     }
12     Node* N = new Node; //if n does not exist a new node will be created
13     N->info = n;
14     N->counter = 1;
15     N->next = head;
16     head = N;
17 }
18 void LinkedList::removeNode(int n){
19     if (head == NULL) { //to check if node is initially empty
20         return;
21     }
22     else if (head->info == n) { // if not then check if info of node is the same as user input
23         head = head->next;
24     }
25     else { // if not remove node
26         Node* curr = head;
27         while (curr->next != NULL) {
28             if (curr->next->info == n){
29                 curr->next = curr->next->next;
30                 return;
31             }
32         }
33     }
34 }
35 void LinkedList::display() { // to display on console
36     Node* curr = head;
37     while (curr != NULL) { // if linked list is not empty
38         cout << curr->info << " has occurred this number of times: " << curr->counter << endl;
39         curr = curr->next;
40     }
41 }
42
43 int LinkedList::sumNodes() { //to sum the list
44     Node* curr = head;
45     int sum = 0;
46     //pass through all the nodes in a loop and sum all of them
47     while (curr != NULL) {
48         sum = sum + curr->info;
49         curr = curr->next;
50     }
51     return sum;
52 }
```

The functions were defined in the LinkedList.cpp after the constructor was defined. addNode was defined to receive a value from the main, then create a new pointer to point at the head, if the head is not empty, then it will enter a loop and check if any of the input that has been added is equal to n, this way it will be able to check the number of occurrences of that node, however, if n does not exist then a new node will be created, the info will be added, the counter

will change and the header will move next. To remove the node first the program will check if the head is NULL if it is it means the list is empty. If not, the head will continue to advance and be removed when n is discovered. The purpose of the display function is to display the values from the vector and the number of its occurrences onto the console. To display the number of occurrences, it will make use of the counter that was created and used in the addNodes function. The information will be displayed one by one according to the order of the integers in the linked list. SumNodes will total all the nodes and return the result as the last step.

main.cpp:

```
1  #include "LinkedList.h"
2  LinkedList createLL(vector<int> v) { // 4.function to create linked list
3      LinkedList x; //create object for class LinkedList
4      for (int i = 0; i < v.size(); i++) { //create a for loop depending on the size of vector
5          x.addNode(v[i]); //add vector to a node to form the linked list
6      }
7      return x; // return linked list
8  }
9
10 int main() {
11     vector<int> v = { 5, 6, 1, 3, 5, 8, 9, 1, 5 }; //vector containing the values of type int
12     LinkedList x = createLL(v); //calls function create Linked List
13     x.display(); // to display linked list onto screen
14     cout << "The sum is: " << x.sumNodes() << endl; //to show the sum of values
15 }
```

The function to generate a linked list can be found in Main.cpp. First, a vector with a specific range of numbers is included in the main. The linked list produced by the function createLL will be added to an object of type LinkedList named x. The function createLL is used to convert the vector into a linked list. Using the addNode function developed in the LinkedList.cpp, it will receive the linked list v from the main and enter a for loop based on the size of the vector. A list called x will be created by adding a vector and then it will return to the main. Finally, the sum will then be determined and displayed using the sumNodes function, and this will be presented on the console using the display function.

The output:

```
9 has occurred this number of times: 1
8 has occurred this number of times: 1
3 has occurred this number of times: 1
1 has occurred this number of times: 2
6 has occurred this number of times: 1
5 has occurred this number of times: 3
The sum is: 32

C:\Users\samar\source\repos\Assignment 4-Problem 2\x64\
0.
Press any key to close this window . . .|
```

The output will show the integer at the beginning, followed by the number of times it appeared in the linked list. For example, the integer 9 appeared just once whereas the integer 5 appeared three times. Finally, the total number of times the linked list appeared is summed together at the end and presented.