Preparatory data Structure (CSCI 591)

Project - IV

Evaluating an Inventory using a linked list

Submitted By: Taddese Erba

February 20, 2020

St. Cloud state university

Department of Computer Science

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

Design Document

Introduction

A linked list is a linear structure of ordered objects that are stored at random memory locations and linked together by pointers. A linked list can be a singly linked list or a doubly-linked list. A doubly linked list stores data in between two fixed containers usually called the header and the trailer. The header is the first Node and the trailer is the last node in the linked list.

This project will implement an inventory list using the `MyInventory` class. The `MyInventory` class consists of the `inv.h` header file where all the private and public variables and functions declared. The `inv.cpp` implementation file contains all the implementation for the classes. The `main.cpp` file is the testing file for all the implemented class functions.

Data Structure

This program has three distinct files. The `inv.h` file contains all the declaration of the required functions and a few function decoration (implementation). It is the framework for `MyInventory` class implementation. It consists of two private objects, the `struct Node` object which is used to lold the two main components of a node and the `Node * getNode(long int num_0, string name_0, double p_0, int q_0, Node * list)` function used to declare and initialize the nodes. In addition to the two private objects, the header file contains twelve public functions which include two constructors, one destructor, eight operational functions, and one friend function.

Functions

As mentioned in the Data Structure section, there are twelve functions in this project. The first two functions, the `LinkedList()` and `LinkedList(const LinkedList& source)` are constructors.

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

The `MyInventory ()` constructor function is used to initialize the class. The constructor `MyInventory(const MyInventory& source)` is a copy constructor that is used to copy the elements of the list. The third function, `~ MyInventory ()`, is a destructor. It is used to delete all the nodes, deallocate the memory, and return it to the operating system. The `void re_Initialize()` function is used to re-initialize the linked lists to empty. The `void insert(long int new_num, string new_name, double new_p, int new_q)` function is used to insert items into the list. The `void remove(long int old_num) function` is used to remove a node from the list. This function uses the inventory number as a reference to search and remove the node. The `void operator = (MyInventory source)` is a function that overloads the assignment operator (=) to be used in the assignment operation involving copying the elements of the list. The `bool isEmpty()` function returns true if the list is empty or false otherwise. The `int listLength()` function will count each node in the list and return the size of the list (number of nodes). The `bool isPresent(long int t_num)` will check if an item is in the list and returns true. This function also uses the inventory number to search for the item. If the item is not found it returns false to `main()`. The `long int kthValue(int numval)` function will return the k^{th} node of the list. If the node is not found, or the list is empty, it returns nothing. The `friend ostream& operator << (ostream& out_s, const MyInventory& l)` is a friend function that is used to overload the outstream operator (<<) for the purpose of printing all the elements of the list. For convenience reasons, the friend function is implemented in the header file where it was declared but outside of the class `MyInventory`.

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

The Main Program

As a testing function, where the implementation is tested, there are many things going on in the `main()` function. To keep things simple, I will talk only about the main components of the main() function. The key frameworks in the main() function are the instantiation of the class `MyInventory` and representation of the key operations by a menu system. There are ten main menus from which the user can choose to perform an operation. The menus are represented by alphabets that are closely related to the operation followed by the name of the operation as in ***I -- Insert Item*** and ***R -- Remove Item***. The menus are continuously displayed after each operation until the user chooses to quit the program. A switch statement will track each choice of the user and perform the necessary operation accordingly. It may also worth mentioning the `bool searchArray(const char [], int, char)` function that is the part of the `main()` function that is used to search the array of constants that hold the alphabets designated to the menu. This enables that if the user enters a choice that is not available, the program can display the necessary message and exit the program.

Code listing

a. The header file (`List.h`)

```
/*
  *** This is the "inv.h" header file. ****
    It contain the following three main parts.
        1. The class MyInventory
            => This class hosts:
                --> The private data type struct
                --> The private function get_node()
                    to declare and initialize
                    the struct object.
                --> The public constructors
                    - LinkedList() declaration and implimentation
                    - LinkedList(const LinkedList& source) declaration.
                --> The declaration of eight public functions & one friend function.
        2. The implementation of the friend function outside the class.
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
15      Precondition for this program to run properly:
16          => The program can read in file from the local machine only.
17      Postcondition for the program
18          => It can handle any number of items.
19          => It lists the items from smallest to the largest inventory number.
20          => It performs the following functions:
21              -> retaining the copy of the original data
22              -> inserting new item
23              -> deleting an item
24              -> re-initializing the list to empty
25              -> searching and returning the kth value of the list
26              -> looking up for an item in the list
27              -> checking if the list content for emptiness
28              -> print the list on the screen
29              -> counting the number of items in the list.
30  */
```

```
31  #include <iostream>
32  #include <iomanip>
33  #ifndef _INV
34  #define _INV
35  #include <ostream>
36  #include <string>
37  #include <fstream>
38  using namespace std;
39  class MyInventory{
40      private:
41          struct Node{
42              long int num;
43              string name;
44              double price;
45              int quantity;
46              Node * next;
47              Node * last;
48      };
49      Node * head;
50      Node * getNode(long int num_0, string name_0,
51                     double p_0, int q_0, Node * list);
52      public:
53          MyInventory(){      //default constructor
54              head = NULL;
55          }
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
 56          MyInventory(const MyInventory& source); // copy constructor
 57          ~MyInventory(); // destructor
 58          // to re-initialize the list to empty. The make_empty() function
 59          // is absent becouse it will have the same function as re_initialize()
 60          //void re_Initialize();
 61          void insert(long int new_num, string new_name,
 62                      double new_p, int new_q);          //to insert items to the list
 63          void remove(long int old_num);      //to remove items from the list
 64          void operator = (MyInventory source); //"="Operator overloading
 65          bool isEmpty();             //to check if the list is empty or not
 66          int listLength();       //to get the number of nodes in the list
 67          bool isPresent(long int t_num); //to check if an item is in the list
 68          void make_empty();
 69          //to access the kth item of the list
 70          long int kthValue(long int numval);
 71          //the friend function is used for the purpose of
 72          //"<<" operator overloading.
 73          friend ostream& operator << (ostream& out_s, const MyInventory& l);
 74          void readFile(ifstream& in);
 75  };
 76  // Implimentation of the friend function.
 77  ostream& operator << (ostream& out_s, const MyInventory& l){
 78      double total = 0;
 79      MyInventory::Node * ptr;
 80      ptr = l.head;
```

```
 81      cout << left << setw(8) << "Number"
 82                           << "    " << "Name"
 83                           << "      " << "UnitPrice"
 84                           << "    " <<"Quantity"
 85                           << "       " << "Value" << endl;
 86      cout << "-------------------------------------------------------\n";
 87      while(ptr -> next != NULL){
 88          cout << left << setw(10) << fixed;
 89          out_s << ptr -> num
 90                << left << fixed << setw(10)<<ptr -> name << setprecision(2)
 91                << "$ " << left << fixed << setw(10) << ptr -> price
 92                << left << fixed << setw(14)<<ptr -> quantity << setprecision(2)
 93                << "$ " <<left << fixed << setw(8)<< (ptr ->price) * (ptr ->quantity)
 94                                << endl;
 95          ptr = ptr -> next;
 96          total = total + (ptr ->price) * (ptr ->quantity);
 97      }
 98      cout << "-------------------------------------------------------\n";
 99      cout << "Total value of inventory =                $ " << total <<endl;
100      return out_s;
101  }
102  #endif  //end of header file definition.
103  /* ================================================================================*/
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

b. The implementation file (`List.cpp`)

```cpp
/*
    **** This is the "List.cpp" implementation file  ****
    This file implements all the functions defined in the
    "List.h" header file. Hence it is the file for class
    implimentation.
*/
#include <iostream>
#include "inv.h"
#include <cstddef>
#include <fstream>
using namespace std;
MyInventory::Node * MyInventory::getNode(long int num_0,
        string name_0, double p_0, int q_0, Node * list){
    Node * temp;
    temp = new Node;
    temp->num = num_0;
    temp->name = name_0;
    temp->price = p_0;
    temp->quantity = q_0;
    temp->next = list;
    return temp;
}
MyInventory::MyInventory(const MyInventory& source){
    Node * ptr;
    Node * last;
```

```cpp
    if(source.head == NULL)
        head = NULL;
    else{
        head = getNode(source.head -> num,
                       source.head -> name,
                       source.head -> price,
                       source.head -> quantity,
                   NULL);
        last = head;
        ptr = source.head -> next;
        while(ptr != NULL){
            last->next = getNode(ptr->num, ptr->name,
                                 ptr->price, ptr->quantity, NULL);
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
39                last = last->next;
40                ptr  = ptr->next;
41            }
42        }
43    }
44    MyInventory::~MyInventory(){
45        Node * temp;
46        while(head != NULL){
47            temp = head;
48            head = head->next;
49            delete temp;
50        }
51    }
52    void MyInventory::operator =(MyInventory source){
53        Node * ptr;
54        Node * last;
55        MyInventory empty;
56        if(&source != this){
57            empty.~MyInventory();
58
59            head = getNode(source.head -> num,
60                           source.head -> name,
61                           source.head -> price,
62                           source.head -> quantity,
```

```
63                           NULL);
64            last = head;
65            ptr = source.head -> next;
66            while(ptr != NULL){
67                last->next = getNode(ptr->num, ptr->name,
68                                     ptr->price, ptr->quantity, NULL);
69                last = last->next;
70                ptr  = ptr->next;
71            }
72        }
73        else return;
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
74  }
75  int MyInventory::listLength(){
76      Node * ptr;
77      int count = 0;
78      ptr = head;
79      while(ptr != NULL){
80          ++count;
81          ptr = ptr->next;
82      }
83      return count;
84  }
85  bool MyInventory::isEmpty(){
86      Node * ptr;
87      ptr = head;
88      if(ptr == NULL)
89          return true;
90      else return false;
91  }
92  bool MyInventory::isPresent(long int t_num){
93      Node * ptr;
94      ptr = head;
95      while(ptr != NULL && ptr -> num != t_num){
96          ptr = ptr -> next;
97      }
98      return ptr != NULL;
```

```
99  }
100 void MyInventory::insert(long int new_num, string new_name,
101                           double new_p, int new_q){
102     Node* prev;
103     if(head == NULL || new_num < head -> num)
104         head = getNode(new_num, new_name, new_p, new_q, head);
105     else{
106         prev = head;
107         while(prev -> next != NULL && prev -> next -> num < new_num)
108             prev = prev ->next;
109         prev -> next = getNode(new_num, new_name, new_p, new_q,
110                                prev -> next);
111     }
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
112 }
113 void MyInventory::remove(long int old_num){
114     Node * prev;
115     Node * temp;
116     prev = head;
117     if(head -> num == old_num){
118         head = head -> next;
119         delete prev;
120     }
121     else{
122         while(prev -> next ->num < old_num)
123             prev = prev -> next;
124         temp = prev -> next;
125         prev -> next = temp -> next;
126         delete temp;
127     }
128 }
129 long int MyInventory::kthValue(long int numval){
130     Node * prev;
131     prev = head->next;
132     for(int i = 0; i < numval; i++)
133         prev = prev -> next;
134     return prev->num;
135 }
136 void MyInventory::readFile(ifstream& in){
137     Node * last;
```

```
138     long int num_f; string name_f; double p_f; int q_f;
139     in.open("F:\\School\\CSCI 301\\My Projects ECE 591\\Project 4\\f.dat");
140     if (!in)
141         cout << " File not found!" << endl;
142     else{
143         make_empty();
144         in >> num_f >> name_f >> p_f >> q_f;
145         head->next = getNode(num_f, name_f, p_f, q_f, NULL);
146         last = head->next;
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
147         while (!in.eof()){
148             last->next = getNode(num_f, name_f, p_f, q_f, NULL);
149             last = last->next;
150             in >> num_f >> name_f >> p_f >> q_f;
151         }
152     }
153 }
154 void MyInventory::make_empty(){
155     Node * prev;
156     prev = head->next;
157     if(prev == NULL){}
158     else{
159         while(prev->next != NULL){
160             head->next = prev->next;
161             delete prev;
162             prev = head->next;
163         }
164     }
165 }
166 /* ========================================================================*/
```

c. The testing file (main.cpp)

```
 1 /*
 2     **** This is the "main.cpp" testing file  ****
 3     This file tests the validity of the class implimentation
 4     functions. The "main.cpp" has the following major duties:
 5     1. It provides the user with a menu choice to enable them
 6         to choose from the available menus. It repeats the menu
 7         once the choosen task is completed and waits for the second
 8         choce until the user chooses to quit the program.
 9     2. It intializes the class LinkedList and calls its member functions
10         to perform the desired operations. Once the operation is over,
11         it anounces the result of that particular operation (choice).
12     3. It declare, implement, and run a function called
13         searchArray(const char [], int, char) that searches for the the
14         presence of the choice entered by a user in a constant array.
15         If the search is successful, the choice is performed.
16         If the search is unsuccessful, it displays a message accordngly.
17 */
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
18 #include <iostream>
19 #include "List.h"
20 #include <iomanip>
21 #include "List.cpp"
22 #include<ostream>
23 using namespace std;
24 bool searchArray(const char [], int, char);
25 int main(int argc, const char * argv[]){
26     char ch;
27     const char array[] = {'c', 'e', 'i', 'm', 'l', 'n', 's', 'q', 'r', 'w'};
28     int item;
29     bool in, ck;
30     LinkedList source, list, list2;
31     cout << " This program will perform the following"
32          << "tasks.\n You must choose and enter the task"
33          << "\n you want to perform according to the \n instructions"
34          << "in the lists\n";
35     cout << " ================================" << endl;
36     cout << " Please choose from the list below." << endl;
37     cout << " ============================================" << endl;
38     cout << " => I -- Insert Item"<<setw(30)<<"=> R -- Remove Item\n"
39          << " => E -- Check Emptiness"<<setw(25)<<"=> C -- Copy Items\n"
40          << " => L -- Lookup an Item"<<setw(27)<<"=> N -- Count Items\n"
41          << " => S -- Search Value"<<setw(32)<<"=> W -- Print Contents\n"
42          << " => M -- Make Empty"<<setw(31)<<"=> Q -- Exit Program" << endl;
43     cout << " ============================================" << endl;
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
44      cout << " ============================================" << endl;
45      cin >> ch;
46      in = searchArray(array, sizeof(array), ch);
47      if(in == false){
48          cout << " The choice you entered doesn't exist.\n";
49          cout << " See you later.\n Goodby!"<< endl;
50          exit(0);
51      }
52      else{
53          while(ch){
54              switch(ch){
55                  case 'i':
56                      cout << "Enter inventory to insert: ";
57                      cin >> num >> name >> p >> q;
58                      list.insert(num, name, p, q);
59                      list2.insert(num, name, p, q);
60                      cout << "Inventory inserted successfully."<<endl;
61                      break;
62                  case 'r':
63                      cout << "Enter inventory to remove: ";
64                      cin >> num;
65                      list.remove(num);
66                      cout << "Inventory removed successfully."<<endl;
67                      if(list.isEmpty() == true)
68                          list.~MyInventory();
```

```
69                      break;
70                  case 'e':
71                      ck = list.isEmpty();
72                      if(ck == true)
73                          cout << "List is empty."<<endl;
74                      else
75                          cout << "List is not empty."<<endl;
76                      break;
77                  case 'c':
78                      cout << "Copy of the initial list:\n";
79                      source = list2;
80                      cout << source << endl;
81                      break;
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
82                  case 'l':
83                      cout << "Enter an item to check: ";
84                      cin >> num;
85                      if(list.isPresent(num) == true)
86                          cout << "Stack Number"<<num <<" is in the list.\n";
87                      else
88                          cout << "Stack Number "<<num <<" is not in the list.\n";
89                      break;
90                  case 'n':
91                      cout << "The list has "<<list.listLength()<<" items"<<endl;
92                      break;
93                  case 's':
94                      cout << "Enter the index of the inventory you want to access: ";
95                      cin >> num;
96                      cout << "The inventory at index "<<num<<" is: ";
97                      cout << list.kthValue(num) << endl;
98                      break;
99                  case 'w':
100                     if(list.isEmpty())
101                         cout << "The list is empty\n";
102                     else{
103                         cout << "Here are the items in the current list:\n";
104                         cout << "=======================================\n";
105                         cout << list << endl;
106                     }
107                     break;
```

```
108                  case 'm':
109                      list.~MyInventory();
110                      cout << "List is re-initialized to empty.\n";
111                      break;
112                  case 'q':
113                      cout << "You chose to quit the program.\n";
114                      cout << "See you later!";
115                      exit(0);
116                      break;
117              }
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
116             cout << " Please choose from the list below." << endl;
117             cout << " ============================================" << endl;
118             cout << " => I -- Insert Item"<<setw(30)<<"=> R -- Remove Item\n"
119                  << " => E -- Check Emptiness"<<setw(25)<<"=> C -- Copy Items\n"
120                  << " => L -- Lookup an Item"<<setw(27)<<"=> N -- Count Items\n"
121                  << " => S -- Search Value"<<setw(32)<<"=> W -- Print Contents\n"
122                  << " => M -- Make Empty"<<setw(31)<<"=> Q -- Exit Program" << endl;
123             cout << " ============================================" << endl;
124             cin >> ch;
125         }
126
127     }
128     return 0;
129 }
```

```
130 bool searchArray(const char A[], int n, char ch){
131         int i = 0;
132         bool found = false;
133         while (i < n ){
134             if (ch == A[i] || tolower(ch) == A[i])
135                 found = true;
136             i++;
137         }
138         return found;
139 }
```

Test Results

1. Tests result for inventery file read in.

```
This program will perform the following tasks.
You must choose and enter the task
you want to perform according to the
instructions in the lists
================================
Please choose from the list below.
============================================
=> I -- Insert Item          => R -- Remove Item
=> E -- Check Emptiness      => C -- Copy Items
=> L -- Lookup an Item       => N -- Count Items
=> S -- Search Value         => W -- Print Contents
=> M -- Make Empty           => Q -- Exit Program
============================================
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
=======================================
Number     Name     UnitPrice   Quantity       Value
-------------------------------------------------------
100001    Computer  $ 1239.99   10           $ 12399.90
100002    Books     $ 533.87    90           $ 48048.30
100003    Ink_Pens  $ 5.22      25           $ 130.50
100004    Pencils   $ 2.75      50           $ 137.50
100005    NotePads  $ 1.75      150          $ 262.50
100006    Scissors  $ 3.75      70           $ 262.50
100007    Phone     $ 1087.99   50           $ 54399.50
100007    Phone     $ 1087.99   50           $ 54399.50
-------------------------------------------------------
Total value of inventory =                $ 212039.80
```

2. Tests result for insert() function.

- The second inventor below was inserted.

```
100001    Computer  $ 1239.99   10           $ 12399.90
```

```
w
Here are the items in the current list:
=======================================
Number     Name     UnitPrice   Quantity       Value
-------------------------------------------------------
100001    Computer  $ 1239.99   10           $ 12399.90
100001    Computer  $ 1239.99   10           $ 12399.90
100002    Books     $ 533.87    90           $ 48048.30
100003    Ink_Pens  $ 5.22      25           $ 130.50
100004    Pencils   $ 2.75      50           $ 137.50
100005    NotePads  $ 1.75      150          $ 262.50
100006    Scissors  $ 3.75      70           $ 262.50
100007    Phone     $ 1087.99   50           $ 54399.50
100007    Phone     $ 1087.99   50           $ 54399.50
-------------------------------------------------------
Total value of inventory =                $ 224439.70
```

3. Tests result for remove() function.

- Let us remove inventory **100001** now.

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
w
Here are the items in the current list:
=====================================
Number     Name     UnitPrice   Quantity       Value
-------------------------------------------------------
100001    Computer  $ 1239.99   10           $ 12399.90
100002    Books     $ 533.87    90           $ 48048.30
100003    Ink_Pens  $ 5.22      25           $ 130.50
100004    Pencils   $ 2.75      50           $ 137.50
100005    NotePads  $ 1.75      150          $ 262.50
100006    Scissors  $ 3.75      70           $ 262.50
100007    Phone     $ 1087.99   50           $ 54399.50
100007    Phone     $ 1087.99   50           $ 54399.50
-------------------------------------------------------
Total value of inventory =                $ 212039.80
```

4. Tests result for copy () function.

```
c
Copy of the initial list:
Number     Name     UnitPrice   Quantity       Value
-------------------------------------------------------
100001    Computer  $ 1239.99   10           $ 12399.90
100002    Books     $ 533.87    90           $ 48048.30
100003    Ink_Pens  $ 5.22      25           $ 130.50
100004    Pencils   $ 2.75      50           $ 137.50
100005    NotePads  $ 1.75      150          $ 262.50
100006    Scissors  $ 3.75      70           $ 262.50
-------------------------------------------------------
Total value of inventory =                $ 49103.80
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

5. Tests result for isPresent() function.

```
1
Enter an item to check: 100003
Stack Number100003 is in the list.
 Please choose from the list below.
```

```
1
Enter an item to check: 100008
Stack Number 100008 is not in the list.
 Please choose from the list below.
```

6. Tests result for listLength() function.

```
n
The list has 7 items
 Please choose from the list below.
```

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

7. Tests result for kthValue() function.

```
W
Here are the items in the current list:
=======================================
Number     Name     UnitPrice   Quantity       Value
-------------------------------------------------------
100001    Computer  $ 1239.99   10           $ 12399.90
100001    Computer  $ 1239.99   10           $ 12399.90
100002    Books     $ 533.87    90           $ 48048.30
100003    Ink_Pens  $ 5.22      25           $ 130.50
100004    Pencils   $ 2.75      50           $ 137.50
100004    Pencils   $ 2.75      50           $ 137.50
100005    NotePads  $ 1.75      150          $ 262.50
100006    Scissors  $ 3.75      70           $ 262.50
-------------------------------------------------------
Total value of inventory =                $ 61641.20
```

```
s
Enter the index of the inventory you want to access: 7
The inventory at index 7 is: 100006
 Please choose from the list below.
```

```
s
Enter the index of the inventory you want to access: 4
The inventory at index 4 is: 100004
 Please choose from the list below.
```

Note: If we try to access an index that is not in the list the function will exhaust searching that index and exit the program as shown in the above screenshot.

8. Tests result for write () function

I am not running a separate test for the write () function since I am running it for almost every other step as part of checking the program to see if it is doing what it supposed to do. Please look at the end of the other programs where I occasionally run the write() function.

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

9. Tests result for makeEmpty () function

```
m
List is re-initialized to empty.
 Please choose from the list below.
 ----------------------------------------
```

```
e
List is empty.
 Please choose from the list below.
```

10. Tests result for quit

```
q
You chose to quit the program.
See you later!
--------------------------------
Process exited after 134.7 seconds with return value 0
Press any key to continue . . .
```

Moreover, if the user enters a choice that is not listed the program will announce that and exit as shown below.

User document

This program can perform different tasks on a linked list as shown in the menu below. In order to run the program, you must perform the following steps.

- The program name is `main.cpp`. on the terminal enter the following command to compile and run the program.
  `g++ -o main main.cpp`
- The program will compile and open the following window:

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
This program will perform the following tasks.
You must choose and enter the task
you want to perform according to the
instructions in the lists
==================================
Please choose from the list below.
============================================
=> I -- Insert Item          => R -- Remove Item
=> E -- Check Emptiness      => C -- Copy Items
=> L -- Lookup an Item       => N -- Count Items
=> S -- Search Value         => W -- Print Contents
=> M -- Make Empty           => Q -- Exit Program
============================================
```

- Once the window opens, make a choice from the displayed menu. For example to insert an item type i or I and then enter.

```
i
Enter an item to insert:
```

- Next, type the item you want to insert and then enter. For example, type 15 and enter.

```
i
Enter an item to insert: 15
```

- The program will announce that the item is entered successfully and display the menu to make the next choice.

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

```
Item 15 inserted successfully.
 Please choose from the list below.
 ================================================
 => I -- Insert Item          => R -- Remove Item
 => E -- Check Emptiness      => C -- Copy Items
 => L -- Lookup an Item       => N -- Count Items
 => S -- Search Value         => W -- Print Contents
 => M -- Make Empty           => Q -- Exit Program
 ================================================
```

- If you want to repeat the insert repeat the above procedure; otherwise make the next selection.
- The program will perform in the same manner for all other tasks as for insert. Hence, all the other eight functions will perform in the same manner.
- Feel free to play around with the other choices (alphabets) and see what the program is meant to do.
- If you wish to exit the program, type q (Q) and enter.

```
q
You chose to quit the program.
See you later!
--------------------------------
Process exited after 794.8 seconds with return value 0
Press any key to continue . . .
```

- Now you can close the window.

Here are very important points while using this program

1. You must insert integer values only. If you try to enter something else other than an integer, the program may crash.
2. Do not try to access the ends of the list. If your list has 3 nodes only and you try to access the 4^{th} node, the program will stop and exit.
3. The program will save a copy of the current list you are working with. You can just type c(C) and access that copy. Of course, once you exit the program that copy will not exits.

Taddese Erba
Section – I
Project – Four
Due: January 30, 2020

Summery

The project implements a linked list operation such as inserting a new item, removing an item from the list, making the list empty, checking for the presence of an item, displaying a copy of the original item that contains all the elements of the list and so on. The knowledge of linked lists and their implementation is very crucial because data are often stored and retrieved as lists.

This program can further be improved by making the necessary changes to make the program accommodate various types of data such as characters and strings. This will make the program a more useful data structure where we can store important records such as student records. Furthermore, data could also be made available as a file and a permanent copy of that file is kept with all current updates included while we still have the old data for reference.

I have gained a significant level of confidence and the necessary knowledge to work with linked lists by completing this project. I believe, this project is one of the projects I would keep working on and refer to the most even the future programming computations. [1]

[1] This materials in this document is mostly from the previous project due to the close similarity of the two projects.