Lab 14

In this lab, we will practice coding with sorted linked list.

Name the source code to slist.cpp. This program reads in a number of string type values from the data file and inserts the items into a **sorted list** such that after each item is inserted, the list is kept in a sorted order. The format of the data file is as the following:

```
apple
orange
pineapple
milk
...
```

An incomplete program is given below. You are required to fill in the function declaration and function definition for the following 3 functions:

- "Insert": insert a value into the list such that after the insertion, the list remains to be sorted
- "Delete": delete a value from the list;
- "ReleaseList" : release the memory space of all the nodes in the list.

Here is the program template for you to start this lab.

```
#include <fstream>
#include <iostream>
#include <string>
#include <cassert>
using namespace std;

typedef string ItemType;
struct NodeType {
  ItemType data;
  NodePtr next;
};
typedef NodeType* NodePtr;


// Declare the function "Insert" here

// Declare the function "Delete" here

// Declare the function "ReleaseList" here


bool IsPresent(NodePtr head, ItemType data);
```

```
void DisplayList(NodePtr head);
void BuildList(ifstream & myIn, NodePtr & head);

int main()
{
   ifstream myIn;
   ItemType item;
   NodePtr  head=NULL;  // the pointer points to the beginning of the list

   myIn.open("grocery.dat");
   assert(myIn);

   BuildList(myIn, head);
   DisplayList(head);

   cout << "Enter an item to be deleted from the list:";
   cin >> item;

   // call the function "IsPresent" to determine if the item is in the list
   // If the item is in the list,
   //             call the function "Delete" to delete the item from the list
   //             call the function "DisplayList" to display the list after the deletion
   // else
   //     display a message "The item is not found in the list
   if (IsPresent(head, item))
   {
       Delete(head, item);
       cout << item << " deleted from the list." << endl;
       DisplayList(head);
   }
   else
       cout << "Item is not in the list." << endl;

   myIn.close();

   ReleaseList();

   return 0;
}

// Define the function "Insert" here. This function inserts a value into a list such that
// the list is always in sorted order, i.e., sorted in alphabetically ascending order
```

```
// Define the function "Delete" here. This function deletes a value from  the list
// If the list is empty, show an appropriate message indicating that fact;
// The item to be deleted may be the first item in the list,
// Or it may occur in the middle or at the end of the list.
// If the item is not found in the list, show an appropriate message indicating that;




// Define the function "ReleaseList" here. This function releases the memory of all the
nodes in the list




// function "BuildList" reads the values one by one from the datafile and calls
// the "Insert" function to insert each value into the linked list
void BuildList(ifstream & myIn, NodePtr & head)
{
   string item;

   while (myIn >> item)
   {
       Insert(head, item);
   }
}

// function "DisplayList" prints all the items in the list one by one
void DisplayList(NodePtr head)
{
   NodePtr cur;

   cur = head;
   cout << "The list of items are: ";
   while (cur != NULL)
   {
      cout << cur->data <<  ' ';
      cur = cur->next;
   }
}

// Function "IsPresent" returns true if the item to search for is
```

```
// in the list, otherwise it returns false
bool IsPresent(NodePtr head, ItemType item)
{
   NodePtr cur=head;
   while (cur != NULL)
   {
     if (cur->data == item)
     {
        return true;
     }
     cur = cur->next;
   }
   return false;
}
```

Test the program

Run the program 3 times. The first time, try deleting an item that is at the beginning of the list. The second time, try deleting an item at the end of the list. The third time, try deleting an item in the middle of the list.