Project 1 List

Experiment 1: Sequential list

Objective:

Master the basic operations of linear sequential lists.

Contents:

1. Define the sequential list ADT: SeqList.
2. Realize the following basic operations on list by defining functions:

   SeqList * Create(): Create an empty sequential list and return it;

   void InitList(SeqList * L): Initialize a sequential list with elements input form keyboard;

   Void Destroy(SeqList * L): Destroy a sequential list;

   bool ListEmpty(SeqList * L) : Check whether a sequential list is empty or not

   int ListLength(SeqList * L): Return the number of elements in the list

   void DispList(SeqList * L): Display the list on screen

   ElemType GetElem(SeqList *L, int i) : Return the i^{th} element in the list

   int LocateElem(SeqList *L, ElemType e): Search the element e in the list and return the index

   int SeqInsert(SeqList *L, int i, ElemType e): Insert a new element into The list as the i-th element;

   int SeqDelete(SeqList *L, int i): Delete the i-th element and return its value;

3. Write a main function to use the SeqList ADT. The main function outputs a menu interface and prompts the user to choose the operations such as create an empty list, initialize a list, insert new elements, delete element or print out the list and so on. **Then design many testing cases and check whether the operations are executed properly.**

Experiment 2: Linked list

Objective:

Master the basic operations of linear linked lists.

Contents:

1. Define the linked list ADT: List.
2. Realize the following basic operations on list by defining functions:

   List Create(): Create an empty linked list and return it;

   void InitList(List L): Initialize a linked list with elements input form keyboard;

   void Destroy(List L): Destroy a linked list;

   bool isEmpty(List L) : Check whether a linked list is empty or not

   int ListLength(List L): Return the number of elements in the list

   void DispList(List L): Output the elements in the list

   position Find(List L, ElemType e): Search the element e in the list and return the node pointer

   position Findpre(List L, ElemType e): Search the predecessor node of the element e in the list and return the node pointer

   int Insert(List L, position p, ElemType e): Insert a new element e into the list after the node *p;

   int Delete(List L, position p): Delete the element after the element *p;

   int DeleteElem(List L, ElemType e): Delete the element e from the list.

3. First write a main function to use the linked list ADT. The main function outputs a menu interface and prompts the user to choose the operations such as create an empty list, initialize a list, insert new elements, delete element or print out the list and so on. **Then design many testing cases and check whether the operations are executed properly.**

Experiment 3: Polynomial

Objective:

Familiar with the application of lists.

Contents:

Define the polynomial ADT with linked list. Implement the operations of AddPoly() and MultiPoly() to realize addition and multiplication of two polynomials. Design many testing cases and check whether the operations can be executed properly in any case.