# 实验2 线性表的表示和实现

**2020 级 计科 专业 2 班 学号： 姓名：**

**实验日期： 2021 年 3 月 19 日---- 3月 19 日 报告日期： 2021 年 3 月 19 日**

### 一、实验目的

**1、掌握线性表中元素的前驱、后续的概念。**

**2、掌握顺序表与链表的建立、插入元素、删除表中某元素的算法。**

**3、对线性表相应算法的时间复杂度进行分析。**

**4、理解顺序表、链表数据结构的特点（优缺点）。**

### 二、实验预习

**复习以下概念**

**1、线性表：线性表是数据结构的一种，一个线性表是n个具有相同特性的数据元素的有限序列**

**2、顺序表：采用顺序存储结构的线性表简称为“ 顺序表”**

**3、链表：链表是一种物理存储单元上非连续、非顺序的存储结构，数据元素的逻辑顺序是通过链表中的指针链接次序实现的**

### 三、实验内容和要求

**1、线性表的顺序存储结构。阅读下面程序，在横线处填写欠缺内容，并补充函数的基本功能注释。运行程序，写出结果（运行截图）。请将本题文件命名为："学号 姓名 实验2-1.cpp"**

**#include<stdio.h>**

**#include<malloc.h>**

**#define ERROR 0**

**#define OVERFLOW -2**

**#define OK 1**

**#define MAXSIZE 100 /\*初始分配的顺序表长度\*/**

**typedef int ElemType; /\*定义表元素的类型\*/**

**typedef Status;**

**typedef struct {**

**ElemType \*elem; //存储空间的基地址**

**int length; //当前长度**

**} SqList;**

**int InitList\_sq(SqList &L); /\* \*/**

**int CreateList\_sq(SqList &L,int n); /\* \*/**

**int ListInsert\_sq(SqList &L,int i,ElemType e);/\* \*/**

**int PrintList\_sq(SqList L); /\*输出顺序表的元素\*/**

**int ListDelete\_sq(SqList &L,int i); /\*删除第i个元素\*/**

**int ListLocate(SqList &L,ElemType e); /\*查找值为e的元素\*/**

**Status InitList\_sq(SqList &L) //算法2.1 顺序表的初始化(构造一个空的顺序表L)**

**{**

**L.elem=(ElemType\*)malloc(MAXSIZE\*sizeof(ElemType));**

**//为顺序表分配一个大小为MAXSIZE的数组空间**

**if (!L.elem)**

**return(OVERFLOW); //存储分配失败退出**

**; //空表长度为0**

**return OK;**

**}/\*InitList\*/**

**int CreateList\_sq(SqList &L,int n){**

**ElemType e;**

**int i;**

**for(i=1;i<=n;i++){**

**printf("input data %d: ",i+1);**

**scanf("%d",&e);**

**if(!ListInsert\_sq(L, ,e))**

**return ERROR;**

**}**

**return OK;**

**}/\*CreateList\*/**

**/\*输出顺序表中的元素\*/**

**int PrintList\_sq(SqList L){**

**int i;**

**for(i=1;i<=L.length;i++)**

**printf("%5d", );**

**return OK;**

**}/\*PrintList\*/**

**int ListInsert\_sq(SqList &L,int i,ElemType e){**

**int k;**

**if(i<1||i>L.length+1)**

**return ERROR;**

**if(L.length>=MAXSIZE){**

**return ERROR;**

**}**

**for(k=L.length-1;k>=i-1;k--){**

**L.elem[ ]= L.elem[ ];**

**}**

**L.elem[i-1]=e;**

**;**

**return OK;**

**}/\*ListInsert\*/**

**/\*在顺序表中删除第i个元素,并用e返回其值，i值的合法范围是1<=i<=L.length \*/**

**Status ListDelete\_Sq(SqList &L, int i,ElemType &e) {**

**}**

**/\*在顺序表中查找指定值元素，返回其序号\*/**

**int ListLocate(SqList &L,ElemType e){**

**}**

**int main(){**

**SqList sl;**

**int n,m,k;**

**printf("please input n:"); /\*输入顺序表的元素个数\*/**

**scanf("%d",&n);**

**if(n<=0)**

**{ printf("参数错误，退出。\n");**

**return 0;**

**}**

**printf("\n1-Create SqList:\n");**

**InitList\_sq(sl);**

**CreateList\_sq(sl,n);**

**printf("\n2-Print SqList:\n");**

**PrintList\_sq(sl);**

**printf("\nplease input insert location and data:(location,data)\n");**

**scanf("%d,%d",&m,&k);**

**ListInsert\_sq(sl,m,k);**

**printf("\n3-Print SqList:\n");**

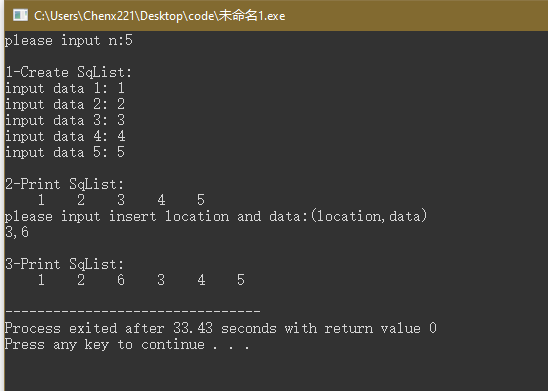
**PrintList\_sq(sl);**

**printf("\n");**

**return 0;**

**}**

* **运行结果**



* **算法分析**

**初始化和创建部分没什么好分析的**

**输出元素用的是简单的顺序输出（用for循环挨个输出**

**插入元素则在开头有判断参数是否合法，插入位置及其后的元素依次后移 后再放入需要插入的数，长度+1**

**删除元素、查找元素部分的代码缺失，已补上**

**删除：待删除元素位置后的元素依次前移，长度-1**

**查找，指针p指向L.elem第一个元素，变量i记录序号，向后依次检查是否存在指定值**

**2、在第1题的文件中补全删除和查找函数，并在主函数中补充代码，验证这两个函数的正确性。**

* **运行结果**
* **算法分析**

**请见上题的算法分析**

**3、线性表的链式存储结构。阅读下面程序，在横线处填写欠缺内容，并补充函数的基本功能注释。运行程序，写出结果（运行截图）。请将本题文件命名为："学号 姓名 实验2-2.cpp"**

**#include<stdio.h>**

**#include<malloc.h>**

**#define ERROR 0**

**#define OK 1**

**typedef int ElemType; /\*定义表元素的类型\*/**

**typedef ;**

**typedef struct LNode{ /\*线性表的循环单链表存储\*/**

**ElemType data;**

**struct LNode \*next;**

**}LNode,\*LinkList;**

**LinkList CreateList(int n); /\*建立带头结点的循环单链表\*/**

**void PrintList(LinkList L); /\* \*/**

**int GetElem(LinkList L,int i,ElemType &e); /\* \*/**

**LinkList CreateList(int n){**

**LNode \*p,\*q,\*head;**

**int i;**

**head=(LinkList)malloc(sizeof(LNode));**

**head->next=head;**

**p= ;**

**for(i=0;i<n;i++){**

**q=(LinkList)malloc(sizeof(LNode));**

**printf("input data %i:",i+1);**

**scanf("%d", ); /\*输入元素值\*/**

**p->next=q; /\*新结点连在表末尾\*/**

**p=q;**

**}**

**q->next= ;**

**return head;**

**}/\*CreateList\*/**

**void PrintList(LinkList L){**

**LNode \*p;**

**p=L->next; /\*p指向单链表的第1个元素\*/**

**while(p!= ){**

**printf("%5d %d %d\n",p,p->data,p->next);**

**;**

**}**

**}/\*PrintList\*/**

**int GetElem(LinkList L,int i,ElemType &e){**

**LNode \*p;int j=1;**

**p=L->next;**

**while(p!=L &&j<i){**

**p=p->next;j++;**

**}**

**if( )**

**return ERROR;**

**e= ;**

**return OK;**

**}/\*GetElem\*/**

**int main(){**

**int n,i;ElemType e;**

**LinkList L=NULL; /\*定义指向单链表的指针\*/**

**printf("please input n:"); /\*输入单链表的元素个数\*/**

**scanf("%d",&n);**

**if(n>0){**

**printf("\n1-Create LinkList:\n");**

**L=CreateList(n);**

**printf("\n2-Print LinkList:\n");**

**PrintList(L);**

**printf("\n3-GetElem from LinkList:\n");**

**printf("input i=");**

**scanf("%d",&i);**

**if(GetElem( ))**

**printf("No%i is %d",i,e);**

**else**

**printf("not exists");**

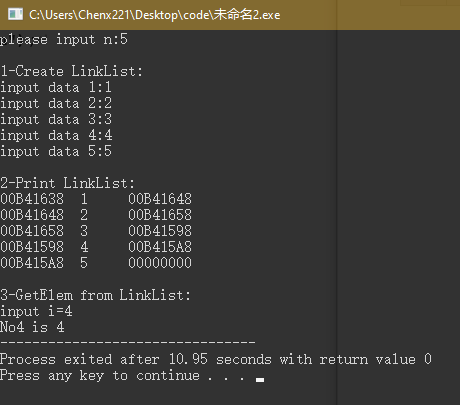
**}else**

**printf("ERROR");**

**return 0;**

**}**

* **运行结果**



* **算法分析**

**Address1 ### Address2**

**Address2 Value1 Address3**

**Address3 Value2 Address4**

**Address4 Value3 Address5**

**Address5 Value4 Address6**

**Address6 Value5 Address7**

**…………………………………………………**

**\*line4没什么用，已被注释掉了**

**4、为第3题补充求单链表长度函数、插入函数、删除函数。并在主函数中补充代码,验证这3个函数的正确性。**

**（1）求单链表长度：**

**int ListLength(LinkList L)**

**//求带头结点的单循环链表L的长度（元素个数）**

**{**

**}**

**（2）插入算法代码：**

**Status ListInsert(LinkList &L, int i, ElemType e)**

**//单链表的插入:在带头结点的单循环链表L中第i个位置插入值为e的新结点**

**{**

**}**

**（3）删除算法代码：**

**Status ListDelete(LinkList &L, int i,ElemType &e)**

**// 单链表的删除:在带头结点的单循环链表L中，删除第i个位置,并将删除元素的数据存入e**

**{**

**}**

* **运行结果**
* **算法分析**

**在插入数据时使用额外的指针存储原插入顺序后的元素的地址，修改插入顺序前的next到新地址上，新地址的next改为前面提到的额外的指针**

**在删除数据时将删除顺序前next改到删除顺序后的地址上，跳过将要被删除的元素即可**

**以下为选做实验：**

**\*5.参看教师的演示资料，任选一种存储结构，实现集合的并、交、差运算。（可只完成其中一种）**

* **算法代码**
* **运行结果**

### 四、实验小结

**在实验2中学习了顺序表和链表的初始化、创建、添改数据、输出，代码比较长，需要静下心看**