《数据结构》实验报告

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 学号 | 2016010207 | 姓名 | 孙博文 | 专业 | 石油工程 | 时间 | 2018.3.25 |

1. **实验题目**

顺序表基本操作

1. **实验目的**

1. 熟练掌握线性表的顺序存储方式下，基本操作的实现算法，巩固和体会顺序表操作特点；

2. 理解动态内存分配；

3. 通过本次实验，熟练掌握C语言中函数调用、参数传递、结构体类型的使用。

**三、算法设计分析**

（一）数据结构的定义

线性表是最常用且最简单的一种数据结构。简言之，一个线性表是n个数据元素的有限序列。

（二）总体设计

实验由主函数、顺序表建立函数、插入函数、删除函数、查找函数、销毁函数组成。

1. 主函数：调用各个函数，达成操作

void main()；

1. 线性表建立函数：创建一个空的线性表

Status InitList\_Sq(SqList &L);

1. 插入函数：向给定位置插入指定数据。

Status ListInsert\_Sq(SqList &L,int i,ElemType e);

1. 删除函数：删除指定位置的数据。

Status ListDelet\_Sq(SqList &L,int i,ElemType &e);

1. 查找函数：实现给定元素的线性查找

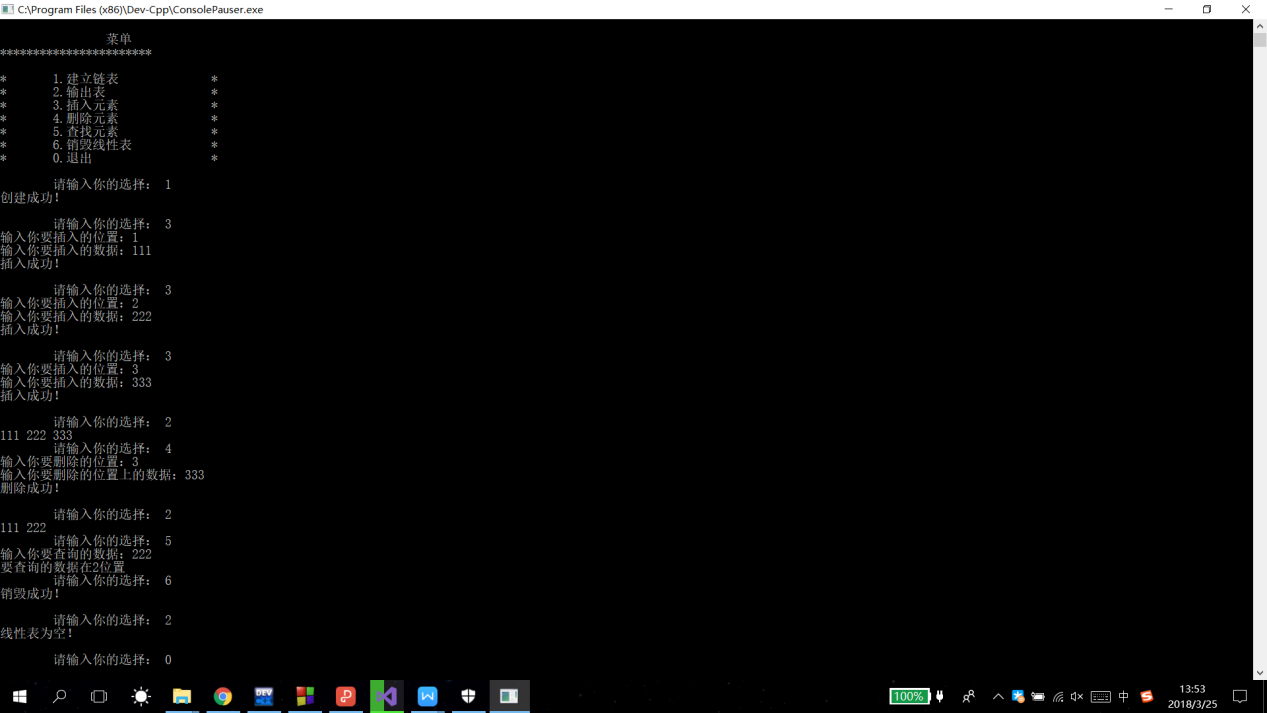
LocateElem\_Sq(SqList L,ElemType e);

1. 销毁函数：销毁整个线性表

DestroyList(SqList &L);

**四、实验测试结果及结果分析**

（一）测试结果（此处给出程序运行截图）



1. 结果分析

通过分别输入命令编号，向指定位置插入指定元素，执行指定操作。

首先输入1，创建空表；再输入3，1,111，意为向1位置前一位插入元素111,。以此类推向后插入222,333；然后输入2，输出线性表；然后输入4,3,333，意为删除位置3前一位的333元素；输入2执行线性表输出，发现333确实已被删除；然后输入5，222，查询222元素所在的位置（首元素为位置1）；然后输入6，销毁整个线性表；然后输入2，显示线性表确实为空。最后输入0，退出系统。

1. **实验总结**

试验通过定义结构体和变量类型重定义，实现了自定义数据结构。插入删除等操作均在这个数据结构上实现。通过上机实习，收获确实很大。

**附录 实验程序代码(该部分请加注释)**

#include <stdio.h>

#include <stdlib.h>

#include <malloc.h>

#define LIST\_INIT\_SIZE 100 //线性表初始分配量

#define LISTINCREMENT 2 //存储空间分配增量

#define OVERFLOW -2

#define OK 1

#define ERROR 0

typedef int ElemType;

typedef int Status;

typedef struct{

ElemType \*elem; //存储空间基址

int lenth; //线性表当前的大小

int listsize; //当前分配的存储容量

}SqList;

Status InitList\_Sq(SqList &L);

Status ListInsert\_Sq(SqList &L,int i,ElemType e);

Status ListDelet\_Sq(SqList &L,int i,ElemType &e);

int LocateElem\_Sq(SqList L,ElemType e);

int DestroyList(SqList &L);

void print(SqList &L);

Status InitList\_Sq(SqList &L) {

L.elem=(ElemType\*)malloc(LIST\_INIT\_SIZE\*sizeof(ElemType));

if (!L.elem){

printf("创建失败,请重新创建！\n");

exit(OVERFLOW);

}

L.lenth=0;

L.listsize=LIST\_INIT\_SIZE;

printf("创建成功！\n");

return OK;

}

void mainmenu()

//输出菜单

{

printf("\n 菜单");

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

printf("\* 1.建立链表 \*\n");

printf("\* 2.输出表 \*\n");

printf("\* 3.插入元素 \*\n");

printf("\* 4.删除元素 \*\n");

printf("\* 5.查找元素 \*\n");

printf("\* 6.销毁线性表 \*\n");

printf("\* 0.退出 \*\n");

}

int main() {

int choose;

SqList La;

mainmenu();

while(1) {

printf("\n 请输入你的选择： ");

scanf("%d",&choose);

switch(choose) {

case 1: InitList\_Sq(La);break;

case 2: print(La);break;

case 3: {

int i,e;

printf("输入你要插入的位置：");

scanf("%d",&i);

printf("输入你要插入的数据：");

scanf("%d",&e);

ListInsert\_Sq(La,i,e);

break;

}

case 4: {

int i,e;

printf("输入你要删除的位置：");

scanf("%d",&i);

printf("输入你要删除的位置上的数据：");

scanf("%d",&e);

ListDelet\_Sq(La,i,e);

break;

}

case 5: {

int e;

printf("输入你要查询的数据：");

scanf("%d",&e);

printf("要查询的数据在%d位置",LocateElem\_Sq(La,e)+1);

break;

}

case 6: DestroyList(La);break;

case 0: exit(0);break;

default: printf("\n输入错误，重新输入！\n");

}

}

return 0;

}

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

if (i<1||i>L.lenth+1)

{

printf("插入失败!\n");

return ERROR;

}

if (L.lenth>=L.listsize) {

ElemType\* newbase=(ElemType\*)realloc(L.elem,(L.listsize+LISTINCREMENT)\*sizeof(ElemType));

if (!newbase) exit(OVERFLOW);

L.elem=newbase;

L.listsize+=LISTINCREMENT;

}

ElemType\* q=&(L.elem[i-1]);

for (ElemType\* p=&(L.elem[L.lenth-1]);p>=q;--p)

\*(p+1)=\*p;

\*q=e;

++L.lenth;

printf("插入成功！\n");

return OK;

}

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

if (i<1||i>L.lenth)

{

printf("删除失败!\n");

return ERROR;

}

ElemType\* p=&(L.elem[i-1]);

e=\*p;

ElemType\* q=L.elem+L.lenth-1;

for (++p;p<=q;++p) \*(p-1)=\*p;

--L.lenth;

printf("删除成功！\n");

return OK;

}

int LocateElem\_Sq(SqList L,ElemType e) {

int i=0;

ElemType\* p=L.elem;

while (i<=L.lenth&&(\*p++)!=e) i++;

if (i<=L.lenth) return i;

else return 0;

}

int DestroyList(SqList &L) {

ElemType\* p=L.elem;

free(p);

L.elem=NULL;

//ElemType\* q=L.elem+L.lenth-1;

//for (++p;p<=q;++p) free(L);

printf("销毁成功！\n");

return 0;

}

void print(SqList &L){

ElemType\* p=L.elem;

if (!p){

printf("线性表为空！\n");

return;

}

ElemType\* q=L.elem+L.lenth-1;

for (;p<=q;++p) printf("%d ",\*p);

}

**实验报告雷同者均视为未做。抄袭请慎重！**

【注意】上机实验报告只提交电子版，要求在课堂上完成并提交。