**湖 州 学 院**

**学生实验报告**

课程名称： 数据结构

实验项目： 实验一 线性表

**湖州学院教务处印制**

**实验一 线性表**

**一、实验目的**

1．了解线性表的逻辑结构特性，以及这种特性在计算机内的两种存储结构。

2．掌握线性表的顺序存储结构的定义及C语言实现。

3．掌握线性表的链式存储结构——单链表的定义及C语言实现。

4．掌握线性表在顺序存储结构即顺序表中的各种基本操作。

5．掌握线性表在链式存储结构——单链表中的各种基本操作。

**二、实验内容**

**1.** 定义并创建一个包含学生信息（学号，姓名，成绩）的顺序表，使其具有如下功能：

(1) 根据指定学生个数，逐个输入学生信息并以顺序结构存储；

(2) 逐个显示学生表中所有学生的相关信息；

(3) 根据姓名进行查找，返回此学生的学号和成绩；

(4) 根据指定的位置可返回相应的学生信息（学号，姓名，成绩）；

(5) 给定一个学生信息，插入到表中指定的位置；

(6) 删除指定位置的学生记录；

(7) 统计表中学生个数。

**2.** 定义并创建一个包含学生信息（学号，姓名，成绩）的单链表，使其具有如下功能：

(1) 根据指定学生个数，逐个输入学生信息并以链式存储结构存储；

(2) 逐个显示学生表中所有学生的相关信息；

(3) 根据姓名进行查找，返回此学生的学号和成绩；

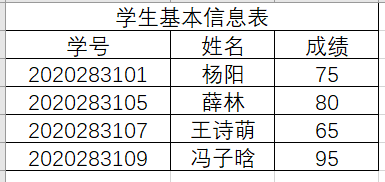
(4) 根据指定的位置可返回相应的学生信息（学号，姓名，成绩）；

(5) 给定一个学生信息，插入到表中指定的位置；

(6) 删除指定位置的学生记录；

(7) 统计表中学生个数。

**三、测试数据：**

****

**第1个学生的学号、姓名和成绩：2020283101 杨阳 75**

**第2个学生的学号、姓名和成绩：2020283105 薛林 80**

**第3个学生的学号、姓名和成绩：2020283107 王诗萌 65**

**第 4个学生的学号、姓名和成绩：2020283109 冯子晗 95**

**四、源程序清单及实例的运行结果（源程序中应该附有必要的注释）。**

1. 基于顺序存储的源程序清单及运行结果截图

#include<iostream>

#include<string>

#include<iomanip>

using namespace std;

typedef int Status;

#define OK 1

#define ERROR 0

#define Maxsize 50

//学生信息的定义

typedef struct

{

char no[11]; //10 位学号

char name[20]; //姓名

int score; //成绩

}Student;

//顺序表的定义

typedef struct

{

Student\* elem; //指向数据元素的基地址

int length; //线性表的当前长度

}SqList;

void InitList(SqList& L); // 构造空的顺序表 L

void CreatList(SqList& L, int n);//根据指定学生个数 n，逐个输入学生信息；

void OutputList(SqList L);//逐个显示学生表中所有学生的相关信息；

void FindList(SqList L, char name[]);//根据姓名进行查找，返回此学生的学号和成绩；

void GetList(SqList L, int i);//根据指定的位置 i 可返回相应的学生信息（学号，姓名，成绩）；

Status InsertList(SqList& L, int i, char no[], char name[], int score);//给定一个学生信息，插入到表中指定的位置；

Status DeleteList(SqList& L, int i);// 删除指定位置的学生记录；

Status CountList(SqList L);// 统计表中学生个数。

int main()

{

SqList L;

int n;//总人数

int i, m;

char a;

char no[11]; //10 位学号

char name[20]; //姓名

int score; //分数

cout << "请输入学生的个数：\n";

cin >> n;

InitList(L);

CreatList(L, n);

cout << "\n 依次显示学生表中所有学生的相关信息:\n";

OutputList(L);

cout << "----------------------\n";

cout << "插入功能请输 i 或 I\n";

cout << "删除功能请输 d 或 D\n";

cout << "查找功能请输 f 或 F\n";

cout << "求学生人数请输入 c 或 C\n";

cout << "输出功能请输 o 或 O\n";

cout << "定位功能请输 g 或 G\n";

cout << "退出功能请输 q 或 Q\n"; //以上为功能介绍

cout << "----------------------\n";

fflush(stdin); //清除键盘缓冲区

cout << "请输入功能选项：";

cin >> a;

while (a != 'q' && a != 'Q')

{

while (a != 'i' && a != 'I' && a != 'd' && a != 'D' &&

a != 'f' && a != 'F' && a != 'c' && a != 'C' &&

a != 'o' && a != 'O' && a != 'g' && a != 'G') //判断输入的值是否有相应的功能

{

cout << "暂时没有此功能，请重新输入：";

fflush(stdin);

cin >> a;

}

switch (a)

{

case 'i':

case 'I':

system("cls");

cout << "\n 请输入要插入的学生的信息（学号、姓名和成绩）：\n";

cin >> no >> name >> score;

cout << "\n 请输入待插入的位置：";

cin >> i;

if (InsertList(L, i, no, name, score) == ERROR)

cout << "插入失败！\n";

else {

cout << "插入成功，插入后的结果为:\n";

OutputList(L);

}

break;

case 'd':

case 'D':

system("cls");

cout << "\n 请输入待删除的位置：";

cin >> i;

if (DeleteList(L, i) == ERROR)

cout << "删除失败！\n";

else {

cout << "删除成功，删除后的结果为:\n";

OutputList(L);

}

break;

case 'f':

case 'F':

system("cls");

cout << "\n 请输入要查找的学生姓名：";

cin >> name;

FindList(L, name);

break;

case 'c':

case 'C':

system("cls");

{int m = CountList(L);

cout << "表总共有" << m << "个学生\n";

break;

}

case 'o':

case 'O':

system("cls");

OutputList(L);

break;

case 'g':

case 'G':

system("cls");

cout << "\n 请输入要定位的位置：";

cin >> i;

GetList(L, i);

break;

}

cout << "----------------------\n";

cout << "插入功能请输 i 或 I\n";

cout << "删除功能请输 d 或 D\n";

cout << "查找功能请输 f 或 F\n";

cout << "求学生人数请输入 c 或 C\n";

cout << "输出功能请输 o 或 O\n";

cout << "定位功能请输 g 或 G\n";

cout << "退出功能请输 q 或 Q\n"; //以上为功能介绍

cout << "----------------------\n";

fflush(stdin); //清除键盘缓冲区

cout << "请输入功能选项：";

cin >> a;

}

return 0;

}

void InitList(SqList& L) // 构造空的顺序表 L

{

L.elem = new Student[Maxsize];

if (!L.elem) exit(ERROR);

L.length = 0;

}

void CreatList(SqList& L, int n)

{

int i;

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

{

cout << "请输入第" << i + 1 << "个学生的学号、姓名和成绩：";

cin >> L.elem[i].no >> L.elem[i].name >> L.elem[i].score;

}

L.length = n;

}

void OutputList(SqList L)

{

int i;

for (i = 0; i < L.length; i++)

{

cout << "第 " << i + 1 << "个学生的学号、姓名和成绩：";

cout << L.elem[i].no << " " << L.elem[i].name << " " << L.elem[i].score << endl;

}

}

void FindList(SqList L, char name[])

{

int i;

for (i = 0; i < L.length; i++)

{

if (strcmp(L.elem[i].name, name) == 0)

{

cout << "\n查找成功，查找的结果为:\n";

cout << "第 " << i + 1 << "个学生的学号、姓名和成绩：";

cout << L.elem[i].no << " " << L.elem[i].name << " " << L.elem[i].score << endl;

}

}

}

void GetList(SqList L, int i)

{

cout << "第 " << i << "个学生的学号、姓名和成绩：";

cout << L.elem[i - 1].no << " " << L.elem[i - 1].name << " " << L.elem[i - 1].score << endl;

}

Status InsertList(SqList& L, int i, char no[], char name[], int score)

{

int j;

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

return ERROR;

if (L.length == Maxsize)

return ERROR;

for (j = L.length - 1; j >= i - 1; j--)

{

L.elem[j + 1] = L.elem[j];

}

strcpy(L.elem[i - 1].no, no);

strcpy(L.elem[i - 1].name, name);

L.elem[i - 1].score = score;

++L.length;

return OK;

}

Status DeleteList(SqList& L, int i)

{

int j;

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

return ERROR;

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

{

strcpy(L.elem[i - 1].no, L.elem[j].no);

strcpy(L.elem[i - 1].name, L.elem[j].name);

L.elem[i - 1].score = L.elem[j].score;

}

--L.length;

return OK;

}

Status CountList(SqList L)

{

return L.length;

}

日历

描述已自动生成

文本

中度可信度描述已自动生成

电脑屏幕的照片上有字

低可信度描述已自动生成

文本

描述已自动生成

图形用户界面

中度可信度描述已自动生成

文本

低可信度描述已自动生成

文本

描述已自动生成

1. 基于链式存储的源程序清单及运行结果截图

#include<iostream >

#include<string>

#include<iomanip>

using namespace std;

typedef int Status;

#define OK 1

#define ERROR 0

#define Maxsize 50

//学生信息的定义

typedef struct {

char no[11]; //10 位学号

char name[20]; //姓名

int score; //成绩

}Student;

typedef struct LNode {

Student data; //数据域

struct LNode\* next; //指针域

}LNode, \* LinkList;

//初始化链表

Status InitList(LinkList& L) {

L = new LNode;

if (!L) return ERROR;

L->next = NULL;

return OK;

}

//前插法建立链表

Status CreatList(LinkList& L, int n) {

LinkList p;

L = (LinkList)malloc(sizeof(LNode));

L->next = NULL;

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

p = new LNode;

cout << "请输入第" << i + 1 << "个学生的学号、姓名、成绩：";

cin >> p->data.no >> p->data.name >> p->data.score;

p->next = L->next;

L->next = p;

}

return OK;

}

//输出链表

Status OutputList(LinkList L) {

LNode\* p = L->next;

cout << "学号\t\t姓名\t\t成绩\n";

while (p) {

cout << p->data.no << "\t" << p->data.name << "\t\t" << p->data.score << endl;

p = p->next;

}

return OK;

}

//根据姓名进行查找，返回此学生的学号和成绩；

Status FindList(LinkList L, char name[]) {

LinkList p = L->next;

while (p) {

if (strcmp(p->data.name, name) == 0) {

cout << "\n查找成功，查找的结果为:\n";

cout << "学号\t\t姓名\t\t成绩\n";

cout << p->data.no << "\t" << p->data.name << "\t\t" << p->data.score << endl;

return OK;

}

p = p->next;

}

return ERROR;

}

//根据指定的位置 i 可返回相应的学生信息

Status GetList(LinkList L, int i) {

LNode\* p = L->next;

int j = 1;

while (p && j < i) {

p = p->next;

j++;

}

if (!p || j > i) return ERROR;

cout << "学号\t\t姓名\t\t成绩\n";

cout << p->data.no << "\t" << p->data.name << "\t\t" << p->data.score << endl;

return OK;

}

//统计表中学生个数。

Status CountList(LinkList L) {

LNode\* p = L->next;

int count = 0;

while (p) {

count++;

p = p->next;

}

cout << "学生个数为：" << count << endl;

return OK;

}

//给定一个学生信息，插入到表中。

Status InsertList(LinkList& L, int i, char no[11], char name[20], int score) {

LNode\* p = L->next;

int j = 1;

while (p && j < i - 1) {

p = p->next;

j++;

}

if (!p || j > i - 1) return ERROR;

LNode\* q = new LNode;

strcpy(q->data.no, no);

strcpy(q->data.name, name);

q->data.score = score;

q->next = p->next;

p->next = q;

return OK;

}

//删除指定位置的学生记录

Status DeleteList(LinkList& L, int i) {

LNode\* p = L->next;

LNode\* q = L;

int j = 1;

while (p && j < i) {

q = p;

p = p->next;

j++;

}

if (!p || j > i) return ERROR;

q->next = p->next;

delete p;

return OK;

}

int main()

{

LinkList L;

int n;//总人数

int i, m;

char a;

char no[11]; //10 位学号

char name[20]; //姓名

int score; //分数

cout << "请输入学生的个数：";

cin >> n;

InitList(L);

CreatList(L, n);

cout << "\n 依次显示学生表中所有学生的相关信息:\n";

OutputList(L);

cout << "----------------------\n";

cout << "插入功能请输 i 或 I\n";

cout << "删除功能请输 d 或 D\n";

cout << "查找功能请输 f 或 F\n";

cout << "求学生人数请输入 c 或 C\n";

cout << "输出功能请输 o 或 O\n";

cout << "定位功能请输 g 或 G\n";

cout << "退出功能请输 q 或 Q\n"; //以上为功能介绍

cout << "----------------------\n";

fflush(stdin); //清除键盘缓冲区

cout << "请输入功能选项：";

cin >> a;

while (a != 'q' && a != 'Q')

{

while (a != 'i' && a != 'I' && a != 'd' && a != 'D' &&

a != 'f' && a != 'F' && a != 'c' && a != 'C' &&

a != 'o' && a != 'O' && a != 'g' && a != 'G') //判断输入的值是否有相应的功能

{

cout << "暂时没有此功能，请重新输入：";

fflush(stdin);

cin >> a;

}

switch (a)

{

case 'i':

case 'I':

system("cls");

cout << "\n 请输入要插入的学生的信息（学号、姓名和成绩）：\n";

cin >> no >> name >> score;

cout << "\n 请输入待插入的位置：";

cin >> i;

if (InsertList(L, i, no, name, score) == ERROR)

cout << "插入失败！\n";

else {

cout << "插入成功，插入后的结果为:\n";

OutputList(L);

}

break;

case 'd':

case 'D':

system("cls");

cout << "\n 请输入要删除的学生的位置：";

cin >> i;

if (DeleteList(L, i) == ERROR)

cout << "删除失败！\n";

else

{

cout << "删除成功，删除后的结果为:\n";

OutputList(L);

}

break;

case 'f':

case 'F':

system("cls");

cout << "\n 请 输 入 要查找的学生的姓名：";

cin >> name;

FindList(L, name);

break;

case 'c':

case 'C':

system("cls");

CountList(L);

break;

case 'o':

case 'O':

system("cls");

OutputList(L);

break;

case 'g':

case 'G':

system("cls");

cout << "\n 请输入要定位的位置：";

cin >> i;

GetList(L, i);

break;

}

cout << "----------------------\n";

cout << "插入功能请输 i 或 I\n";

cout << "删除功能请输 d 或 D\n";

cout << "查找功能请输 f 或 F\n";

cout << "求学生人数请输入 c 或 C\n";

cout << "输出功能请输 o 或 O\n";

cout << "定位功能请输 g 或 G\n";

cout << "退出功能请输 q 或 Q\n"; //以上为功能介绍

cout << "----------------------\n";

fflush(stdin); //清除键盘缓冲区

cout << "请输入功能选项：";

cin >> a;

}

return 0;

}

图片包含 图示

描述已自动生成图示, 示意图

描述已自动生成图片包含 图示

描述已自动生成文本

中度可信度描述已自动生成图形用户界面, 应用程序

描述已自动生成

图片包含 图示

描述已自动生成

图片包含 QR 代码

描述已自动生成文本

描述已自动生成

**五、实验小结**

1.说明遇到的问题及解决方法；

图形用户界面, 文本, 应用程序

描述已自动生成

错误原因：顺序表未初始化

解决办法：初始化顺序表

文本

中度可信度描述已自动生成



错误原因：InsertList已是自定义函数类型，不需要再加Status。

解决办法：删去Status

图形用户界面, 文本, 应用程序, Word

描述已自动生成

错误原因：数组位置错误

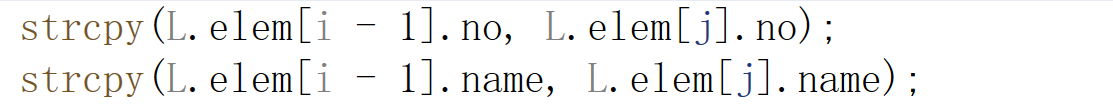
解决办法：

图形用户界面, 文本, 应用程序

描述已自动生成

错误原因：no为char类型，无法直接赋值

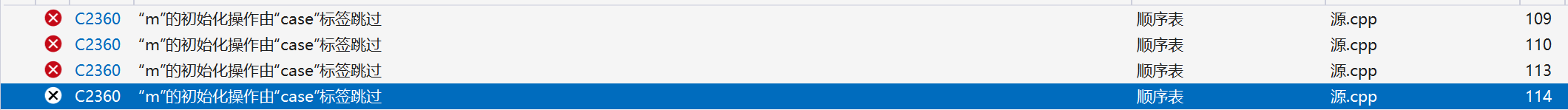
解决办法：使用strcpy进行赋值



图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

错误原因：此错误提示我们目标(Buffer)过小。实际上第二个参数是合并字符串后的字符数量。即，源串大小 + 目标串大小 + 字符串结束符大小("\0")

解决办法：将数组大小改为11

错误原因：在case语句中进行了初始化变量

解决办法：如果要在case后面初始化变量就要用{}将这些语句括起来

图片包含 日历

描述已自动生成

错误原因：查找函数和求学生人数函数无输出语句

解决办法：增加输出语句

图形用户界面, 文本, 应用程序

描述已自动生成

错误原因：findlist返回值为void类型

解决办法：改为status类型

图形用户界面, 文本, 应用程序

描述已自动生成



黑白色的牌子

描述已自动生成

错误原因：for循环语句括号位置出错

解决办法：更改括号位置

文本

描述已自动生成

图形用户界面, 文本, 应用程序

描述已自动生成

错误原因：利用前插法创建单链表时初始化错误

解决办法：对单链表进行初始化

图片包含 图示

描述已自动生成文本

低可信度描述已自动生成

错误原因：查找姓名时错误选择了学号进行查找

解决办法：改为使用姓名进行查找

2.关于程序的特色和改进设想；

程序特色：程序使用了system("cls")系统函数，在输入下一项功能时会对之前的函数进行清屏操作，使界面更加简洁

1. 经过此实验所得到的收获,其他需要说明的情况。

在完成了实验一后，我对顺序表以及链表有了更深刻的了解，在课堂学完线性表之后未进行任何的上机练习，所以本次实验是一次很好的机会对线性表进行巩固与复习。本次实验中增加了独立思考的时间，发现了很多在之前学习时没有遇见或发现的问题，例如顺序表以及链表的初始化，相关数据的赋值比较以及数据溢出以及位置非法等情况。同时由于以前未接触过c++，本次使用c++进行编程，对c++有了更深刻的理解。