项目说明文档

**数据结构课程设计**

**——考试报名系统**

作 者 姓 名： 吴英豪

学 号： 1953608

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

1.分析

1.1背景分析

考试报名工作给各高校报名工作带来了新的挑战，给教务管理部门增加了很大的工作量。随着学生数量和考试数量的日益庞大，如何管理如此庞大的数据显得极为复杂，传统的手工管理工作量大而且容易出错。

随着计算机科学的不断成熟，使用计算机对考试报名系统进行管理，具有手工管理所无法比拟的优势。这些优点能够极大地提高学校和学生的效率。因此，开发一套考试报名系统具有十分重要的意义。

1.2功能分析

作为一个最简易的考试报名系统，首先应该有的功能就是输入学生们的考试报名情况并且可以给予显示。其次报名系统还应该具有插入、删除、修改功能，以保证同学们可以随时更改自己的考试报名情况。

综上，一个考试报名系统应该有这样的功能：输入考生信息；输出考生信息；查询考生信息；添加考生信息；修改考生信息；删除考生信息。

2.设计

2.1数据结构设计

该系统要求大量的增加、删除、修改操作，相对于这些操作，查询操作的影响程度显得不是很主要，因此考虑使用链表数据结构。在本题中使用了单链表的数据结构，同时在第一个节点之前加入了一个附加头，使得改动头结点变得简单，同时也让程序变得更加简洁。

2.2类结构设计

考虑到每个学生都有自己的信息，所以建立一个学生Student类，用于存储学生的基本信息，以及实现与学生相关的成员函数，如信息的修改，信息的输出。返回学生的信息等。

同时创建了一个链表类Linklist，其中含有一个头指针first.通过Linklist指针指向学生Student实现考试报名系统。

2.3成员与操作设计

**学生结点类（Student）**

私有成员

string \_exam\_number; //考号

string \_name; //姓名

string \_gender; //性别

int \_age; //年龄

string \_exam\_category; //报考类别

Student\* link; //指向下一个节点的指针

公有操作：:

Student(); //默认构造函数

Student(string exam\_number, string name, string gender, int age, string exam\_category);//通过相关信息建立对象

~Student(); //析构函数

void print(); //输出学生的相关信息

string getExamNumber()const; //返回学生的考号

string getName() const; //返回学生的姓名

string getGender()const; //返回学生的性别

int getAge()const; //返回学生年龄

string getExamCategory()const; //返回学生的报考类别

void setInformation(int i); //修改学生的信息

friend class Linklist; //友元类

**链表类（Linklist）**

私有成员：

Student\* first; //表头指针

公有操作

Linklist(); //创建一个带有附加头的链表

~Linklist(); //析构函数

void insertNode(int i, string exam\_number, string name, string gender, int age, string exam\_category); //链表的插入

void deleteNode(int i); //删除节点

void changeNode(int i); //修改节点

void findNode(int i); //查找节点

void printNode()const; //打印所有节点

int getLength()const; //获得节点长度

注：因为链表要访问结点内部的数据，所以将链表类LinkList设置为Student类的友元。

2.4文件设计

本系统支持windows系统和Linux系统。

(1)Windows系统

windows系统的相关文件在文件夹01\_1953608\_WuYinghao\_Windows中，

其中类的定义以及其成员函数的定义在Student\_Linklist.h中;main函数以及相关的函数定义在01\_1953608\_WuYinghao.cpp中.01\_1953608\_WuYinghao.exe为Windows平台下的可执行文件。

**01\_1953608\_WuYinghao.cpp和Student\_Linklist.h的编码格式均为**

**简体中文(GB2312)-代码页 936**

(2)Linux 系统

Linux系统的相关文件在文件夹01\_1953608\_WuYinghao\_Linux中,

其中类的定义以及其成员函数的定义在01\_1953608\_WuYinghao\_Linux.h中;main函数以及相关的函数定义在01\_1953608\_WuYinghao\_Linux.cpp中.01\_1953608\_WuYinghao.out为Linux平台下的可执行文件。

**01\_1953608\_WuYinghao\_Linux.cpp和01\_1953608\_WuYinghao\_Linux.h的编码格式均为Unicode(UTF-8 无签名)-代码页65001**

3.实现

3.1插入功能的实现

**特别说明：本系统中输入的值均为大于等于0的数（如插入删除等，详细情况见后文**）

3.1.1功能说明：

输入的位置为插入后的位置。如现在有两个人，现插入学生a，输入1表示插入到最前面，即插入后a在第一位。输入3即表示插入到队尾，即插入后a的位置为3

3.1.2插入功能核心代码

void Linklist:: insertNode(int i, string exam\_number, string name, string gender, int age, string exam\_category)

{

Student\* newNode = new Student(exam\_number,name,gender,age,exam\_category);

if (NULL == newNode)

{

cerr << "创建新节点失败";

exit(1);

}

else

{

Student\* current = this->first;

for (; i > 0; i--)

{

current = current->link;

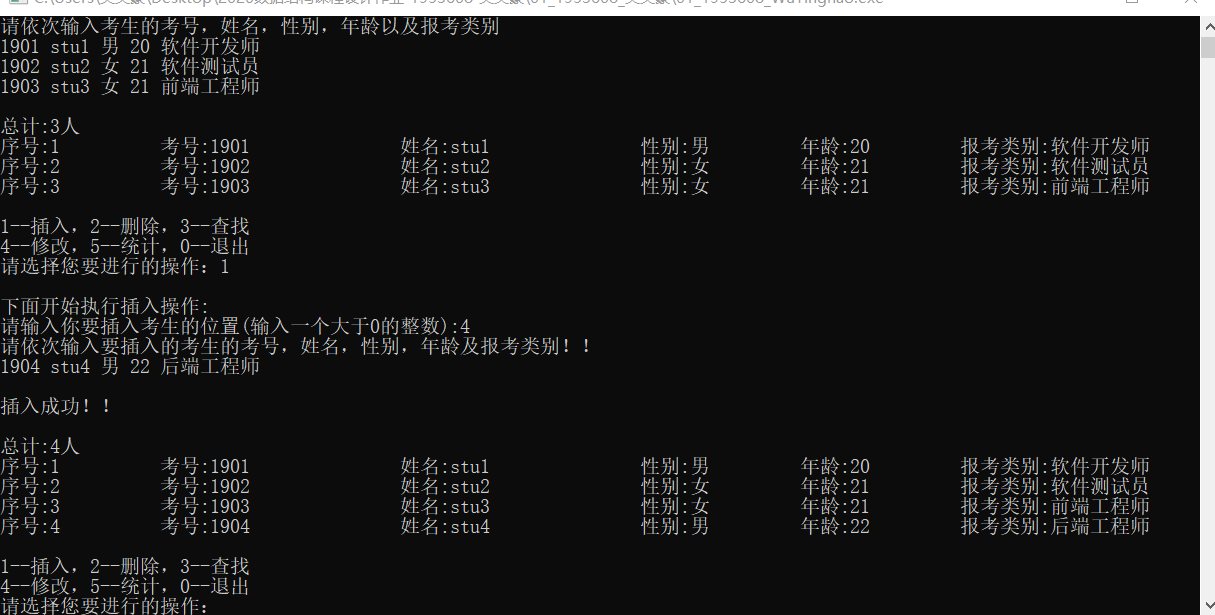
}

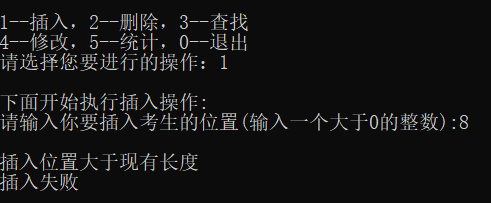
newNode->link = current->link;

current->link = newNode;

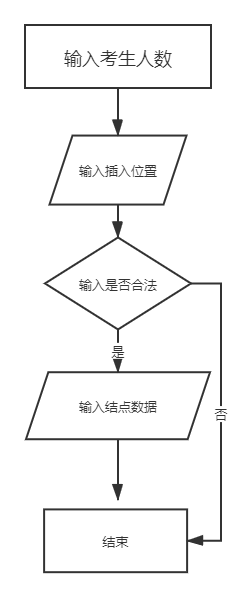
}

}

**3.1.3插入功能截屏展示**



**3.1.4插入功能流程图**



**3.2删除功能的实现**

**3.2.1功能说明**

同插入操作，输入的数代表要删除的位置，输入的数应该大于等于0并且符合实际逻辑（有做输入检测）

**3.2.2 删除功能核心代码**

void Linklist::deleteNode(int i)

{

Student\* current = this->first;

for (; i > 0; i--)

{

current = current->link;

}

Student\* temp = current;

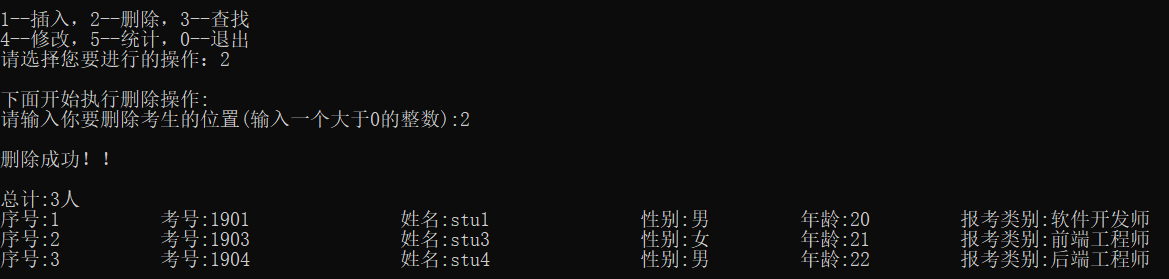
current = current->link;

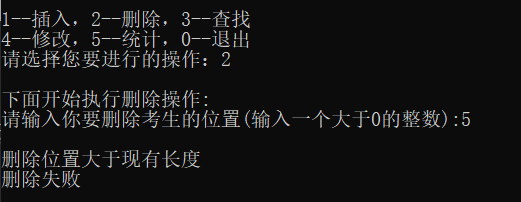
temp->link = current->link;

delete current;

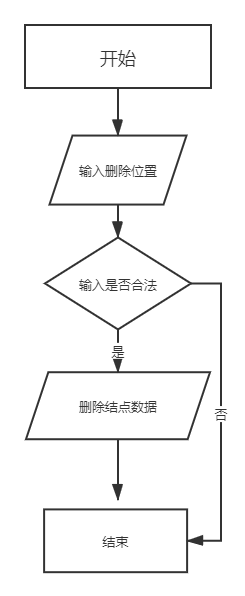
}

**3.2.3 删除功能截屏示例**





**3.2.4 删除功能流程图**



**3.3 查找功能的实现**

**3.3.1 功能说明**

查找某个结点，并返回此学生相关的数据

**3.3.2 查找功能核心代码**

void Linklist::findNode(int i)

{

Student\* current = this->first;

for (; i >= 0; i--)

{

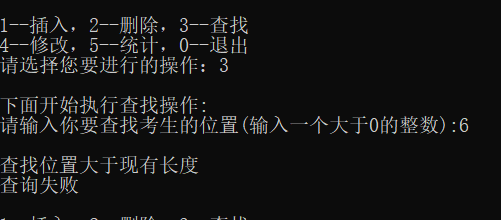
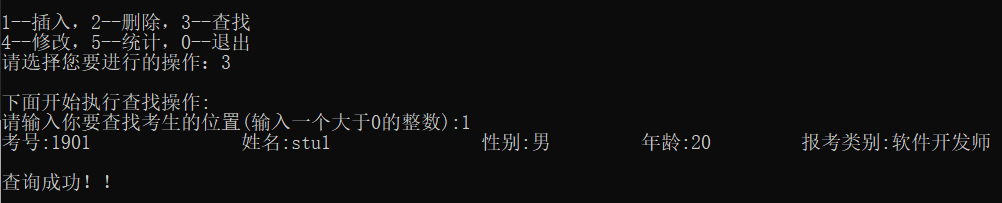
current = current->link;

}

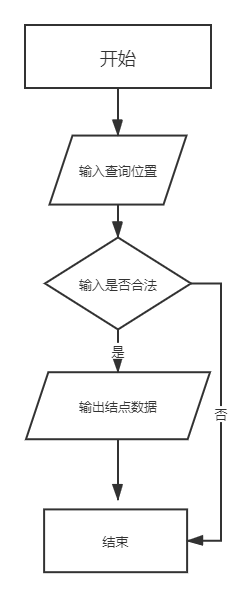
current->print();

}

**3.3.3查找功能截图示例**



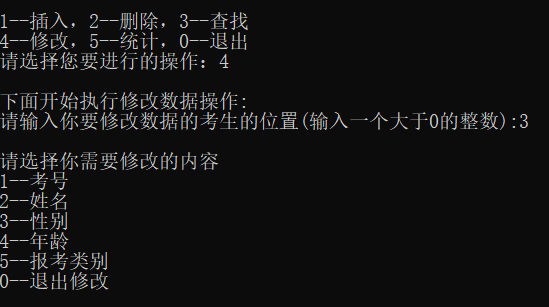
**3.3.4查找功能流程图**



**3.4 修改功能的实现**

**3.4.1 功能说明**

选择需要修改的选项，修改内容



**3.4.2 修改功能核心代码**

void Linklist::changeNode(int i)

{

Student\* current = this->first;

for (; i >= 0; i--)

{

current = current->link;

}

int choose = 0;

while (true)

{

cout << endl << "请选择你需要修改的内容\n1--考号\n2--姓名\n3--性别\n4--年龄\n"

<< "5--报考类别\n0--退出修改\n";

cin >> choose;

if (cin.fail())

{

cin.clear();

cerr << endl << "输入错误，请重新输入" << endl;

cin.get();

continue;

}

current->setInformation(choose);

current->print();

if (0 == choose)

{

break;

}

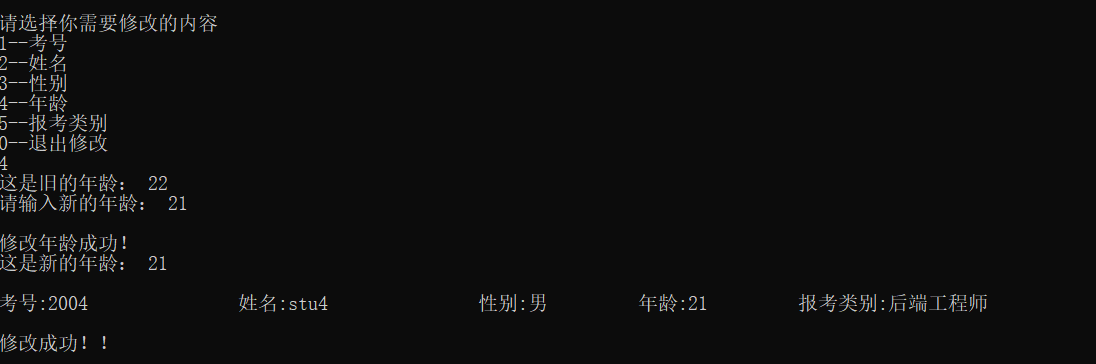
cout << endl << "修改成功！！" << endl;

}

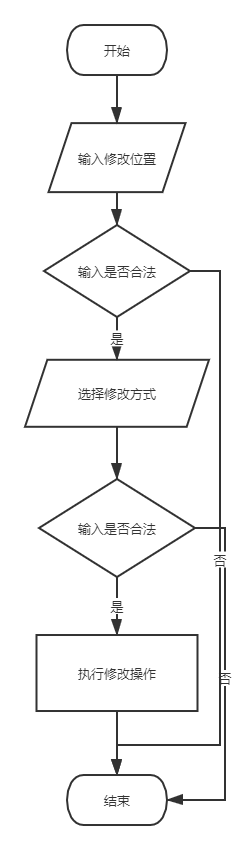
}

**3.4.3 修改功能截屏示例**





**3.4.4修改功能流程图**



**3.5 统计功能的实现**

**3.5.1功能说明**

展示系统内所有学生的信息

**3.5.2 统计功能核心代码**

void Linklist::printNode()const//打印所有节点

{

int length = this->getLength();

int number = 1;

cout <<endl<< "总计:" << length << "人" << endl;

Student\* current = this->first->link;

while (current->link != NULL)

{

current = current->link;

cout << "序号:" << number << "\t\t";

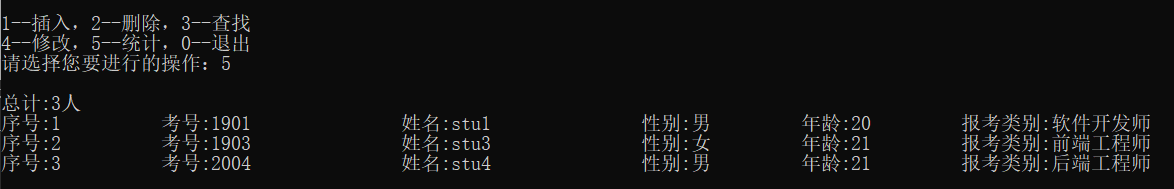
current->print();

number++;

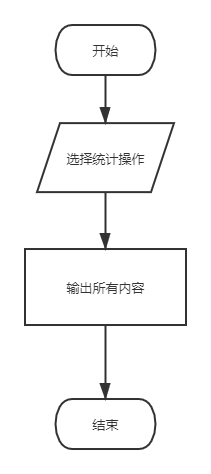
}

}

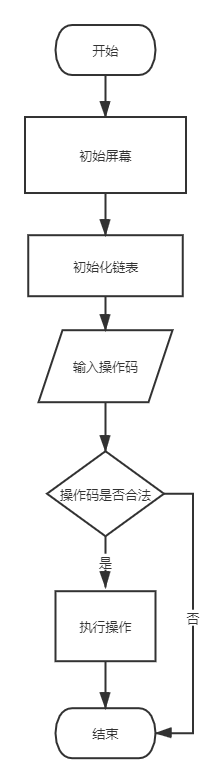
**3.5.3 统计功能截屏实例**



**3.5.4 统计功能流程图**



**3.6 总体流程图**



**4.测试**

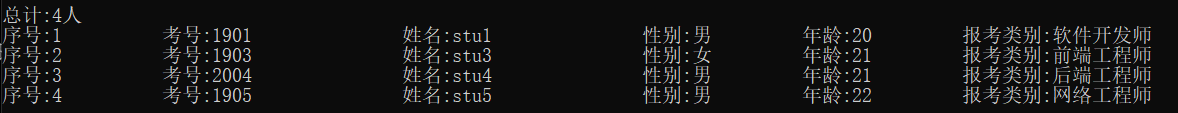
**4.1功能测试**

**4.1.1插入功能测试**

系统已有结果

测试用例：1905 stu5 男 22 网络工程师

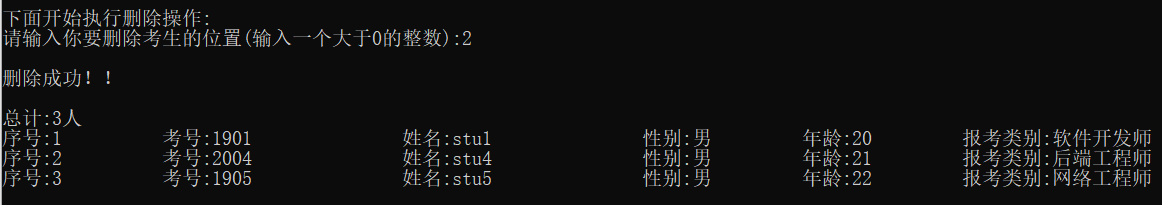
实验结果



**4.1.2 删除功能测试**

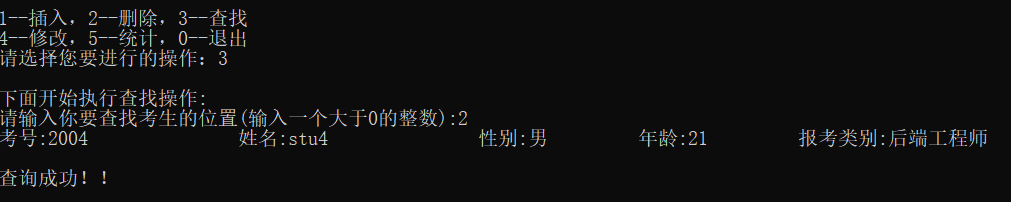
删除序号2 的女生

实验结果：



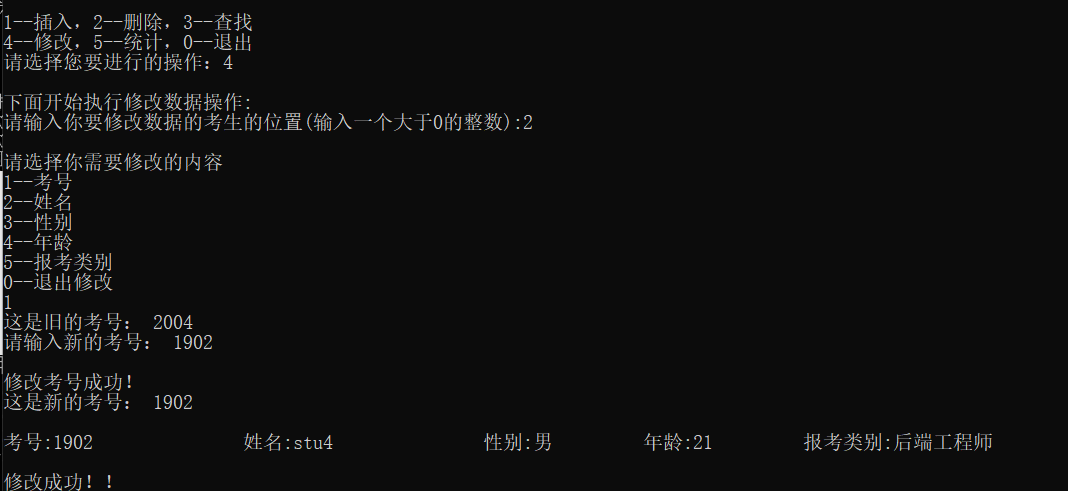
**4.1.3 查询功能测试**

查询学号为2004的学生的数据



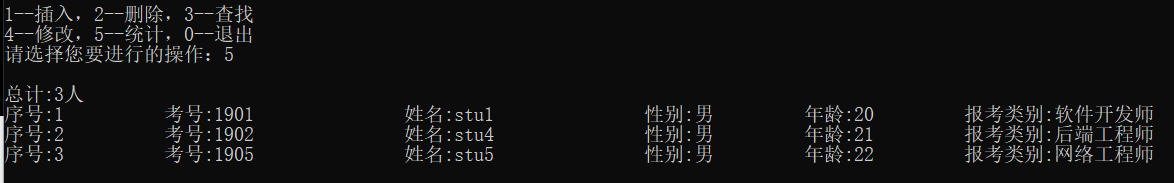
**4.1.4 修改功能测试**

将姓名为stu4的学生的考号从2004改为1902



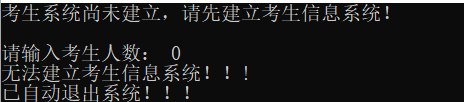
4.1.5统计功能测试

展示系统内所有学生的数据



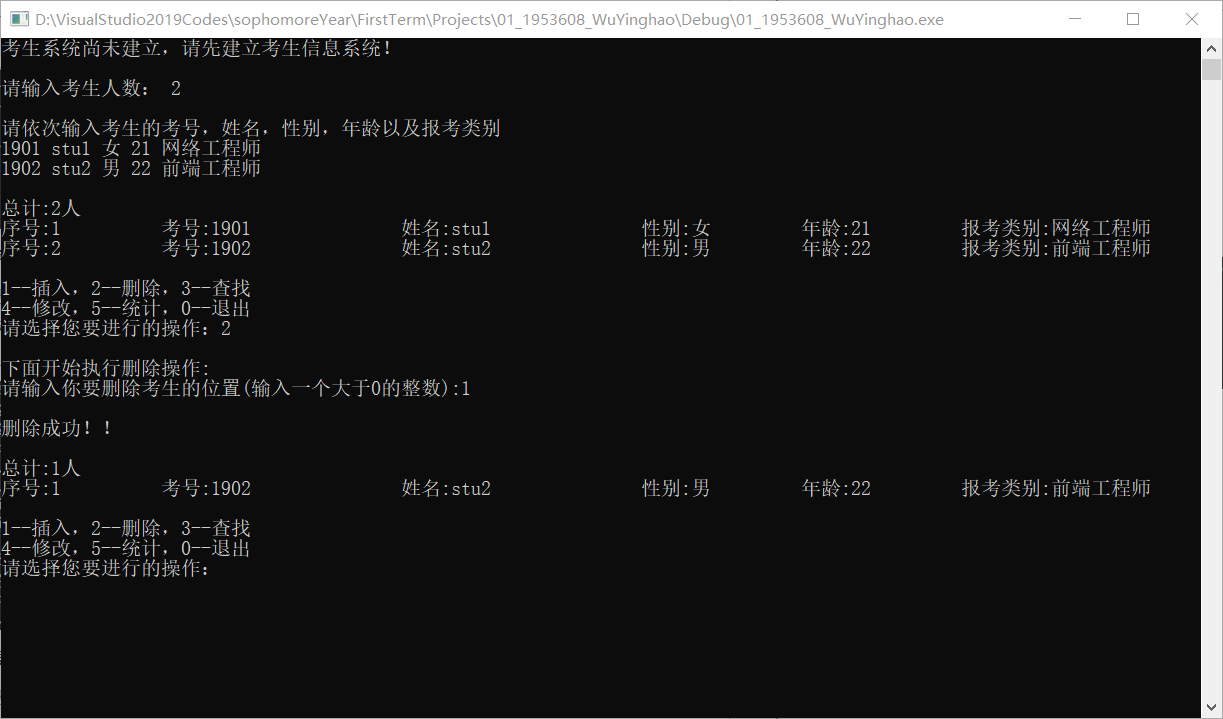
**4.2边界测试**

**4.2.1初始化无输入数据**



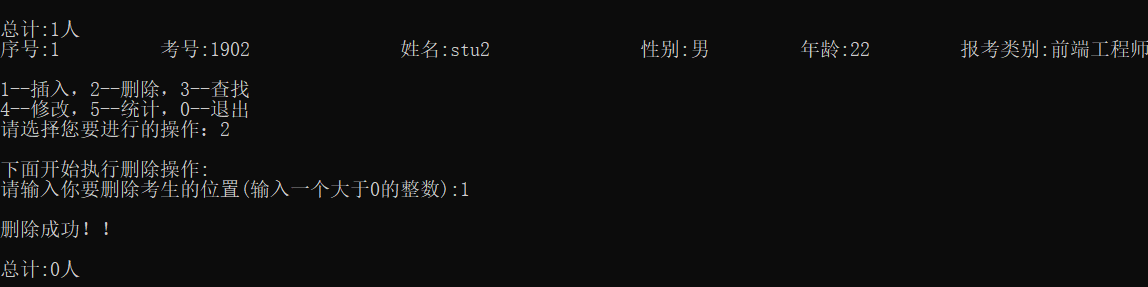
**4.2.2删除头节点**

程序正常运行，不崩溃



**4.2.3. 删除后链表为空**

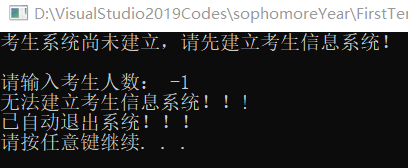
正常运行，不崩溃



**4.3 出错测试**

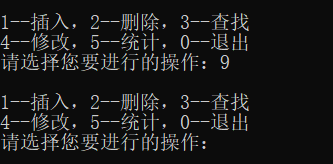
**4.3.1考试人数错误**

输入考生人数为负数，程序给出提示信息，不崩溃



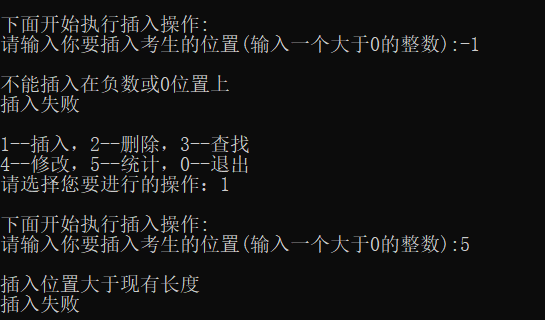
4.3.2操作码错误

输入操作码错误，程序正常运行，不崩溃

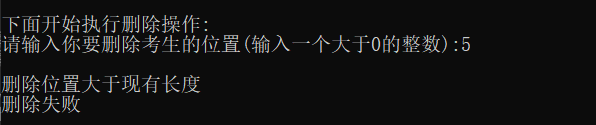


4.3.3 插入位置不存在

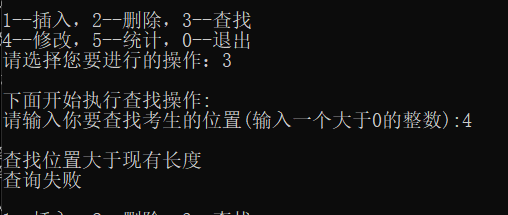
当插入非正数位置或者插入位置大于链表长度的时候，程序正常运行不报错



4.3.4 删除的考号不存在



4.3.5 查询的考号不存在



4.3.6 修改考号不存在

