项目说明文档

数据结构课程设计

——考试报名系统

作 者 姓 名： 杨煜

学 号： 1850217

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

目 录

[1 分析 1](#_Toc25596445)

[1.1 项目简介 1](#_Toc25596446)

[1.2 项目要求 1](#_Toc25596447)

[2 设计 1](#_Toc25596448)

[2.1 数据结构设计 1](#_Toc25596449)

[2.2 类结构设计 1](#_Toc25596450)

[2.3 成员与操作设计 2](#_Toc25596451)

[2.4 链表建立 2](#_Toc25596452)

[3 实现 3](#_Toc25596453)

[3.1 插入功能的实现 3](#_Toc25596454)

[3.1.1 插入功能流程图 3](#_Toc25596455)

[3.1.2 插入功能核心代码 4](#_Toc25596456)

[3.1.3 插入功能截屏示例 5](#_Toc25596457)

[3.2 删除功能的实现 6](#_Toc25596458)

[3.2.1 删除功能流程图 6](#_Toc25596459)

[3.2.2 删除功能核心代码 6](#_Toc25596460)

[3.2.3 删除功能截屏示例 8](#_Toc25596461)

[3.3 查找功能的实现 9](#_Toc25596462)

[3.3.1 查找功能流程图 9](#_Toc25596463)

[3.3.2 查找功能核心代码 9](#_Toc25596464)

[3.3.3 查找功能截图示例 10](#_Toc25596465)

[3.4 修改功能的实现 11](#_Toc25596466)

[3.4.1 修改功能流程图 11](#_Toc25596467)

[3.4.2 修改功能核心代码 11](#_Toc25596468)

[3.4.3 修改功能截屏示例 12](#_Toc25596469)

[3.5 输出功能的实现 13](#_Toc25596470)

[3.5.1 输出功能流程图 13](#_Toc25596471)

[3.5.2 输出功能核心代码 13](#_Toc25596472)

[3.5.3 输出功能截屏示例 14](#_Toc25596473)

[3.6 总体系统的实现 15](#_Toc25596474)

[3.6.1 总体系统流程图 15](#_Toc25596475)

[3.6.2 总体系统核心代码 15](#_Toc25596476)

[3.6.3 总体系统截屏示例 18](#_Toc25596477)

[4 测试 19](#_Toc25596478)

[4.1 功能测试 19](#_Toc25596479)

[4.1.1 插入功能测试 19](#_Toc25596480)

[4.1.2 删除功能测试 19](#_Toc25596481)

[4.1.3 查找功能测试 20](#_Toc25596482)

[4.1.4 修改功能测试 21](#_Toc25596483)

[4.1.5 统计功能测试 21](#_Toc25596484)

[4.2 边界测试 22](#_Toc25596485)

[4.2.1 初始化无输入数据 22](#_Toc25596486)

[4.2.2 删除后链表为空 23](#_Toc25596487)

[4.3 出错测试 23](#_Toc25596488)

[4.3.1 操作码错误 23](#_Toc25596489)

[4.3.2 插入位置不存在 24](#_Toc25596490)

[4.3.4 删除位置不存在 25](#_Toc25596491)

[4.3.5 查找考号不存在 25](#_Toc25596492)

[4.3.6 修改考号不存在 26](#_Toc25596493)

# 1 分析

## 1.1 项目简介

考试报名系统是一个学校不可缺少的部分，它对于学校的管理者和学生来说都至关重要，所以一个良好的考试报名系统应该能够为用户提供充足的信息和功能。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。本项目是对考试报名管理的简单模拟，用控制台选项的选择方式完成下列功能：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。

## 1.2 项目要求

本项目的实质是完成对考生信息的建立，查找，插入，修改，删除等功能。其中考生信息包括准考证号，姓名，性别，年龄和报考类别等信息。项目在设计时应首先确定系统的数据结构，定义类的成员变量和成员函数；然后实现各成员函数以完成对数据操作的相应功能；最后完成主函数以验证各个成员函数的功能并得到运行结果。

# 2 设计

## 2.1 数据结构设计

如上述项目要求所述，该报名系统要求进行大量的添加、删除操作，少量的修改操作，而链表进行增加、删除等操作的时间复杂度较低。因此考虑使用链表的数据结构来完成此报名系统。同时，为了实现简易，在第一个结点之前附加一个头结点，即使用带附加头结点的链表，这样就使得增加或者删除头结点与处理其他结点方法相同，使得程序更为简洁明了。

## 2.2 类结构设计

经典的链表一般包括两个抽象数据类型（ADT）——链表结点类（LinkNode）与链表类（List），而两个类之间的耦合关系可以采用嵌套、继承等多种关系。为方便处理，本系统采用友元的方式使得链表结点类以及链表类可以互相访问。

## 2.3 成员与操作设计

**链表结点类（LinkNode）**

class LinkNode

{

friend class List;//申明友元为链表类

private:

int studentID;//考生考号

string studentName;//姓名

string studentGender;//性别

int studentAge;//年龄

string studentCategory;//报考科目

LinkNode\* link;//指向下一节点的指针

};

**链表类（List）**

class List

{

public:

List();//构造函数

~List();//析构函数

void Input(int studentNumber);//输入构建链表

void Display()const;//输出

void Insert(int position);//插入

void Delete(int position);//删除

void Change(int ID);//修改

void Find(int ID)const;//查找

private:

int listLength;//链表长度

LinkNode\* first;//头节点

};

## 2.4 链表建立

系统首先调用Input()函数来建立一个链表，函数参数为考生人数，随后则依次输入各考生信息，完成链表建立。

void List::Input(int studentNumber)

{

LinkNode\* newNode, \* last;//创立新节点和尾结点指针

last = first;

for (int i = 0; i < studentNumber; i++)//根据输入人数构建链表

{

newNode = new LinkNode;

last->link = newNode;

last = newNode;

cin >> newNode->studentID >> newNode->studentName >> newNode->studentGender >> newNode->studentAge >> newNode->studentCategory;

}

listLength = studentNumber;

}

# 3 实现

## 3.1 插入功能的实现

### 绘图1.gif3.1.1 插入功能流程图

### 3.1.2 插入功能核心代码

void List::Insert(int position)

{

if (position > listLength + 1)//位置出错

{

cout << "该位置不存在" << endl;

return;

}

LinkNode\* current = first;

cout << "请依次输入要插入的考生的考号，姓名，性别，年龄以及报考类别！" << endl;

LinkNode\* newNode = new LinkNode;

cin >> newNode->studentID >> newNode->studentName >> newNode->studentGender >> newNode->studentAge >> newNode->studentCategory;

for (int i = 0; i < position-1; i++)//寻找插入位置

{

current = current->link;

}

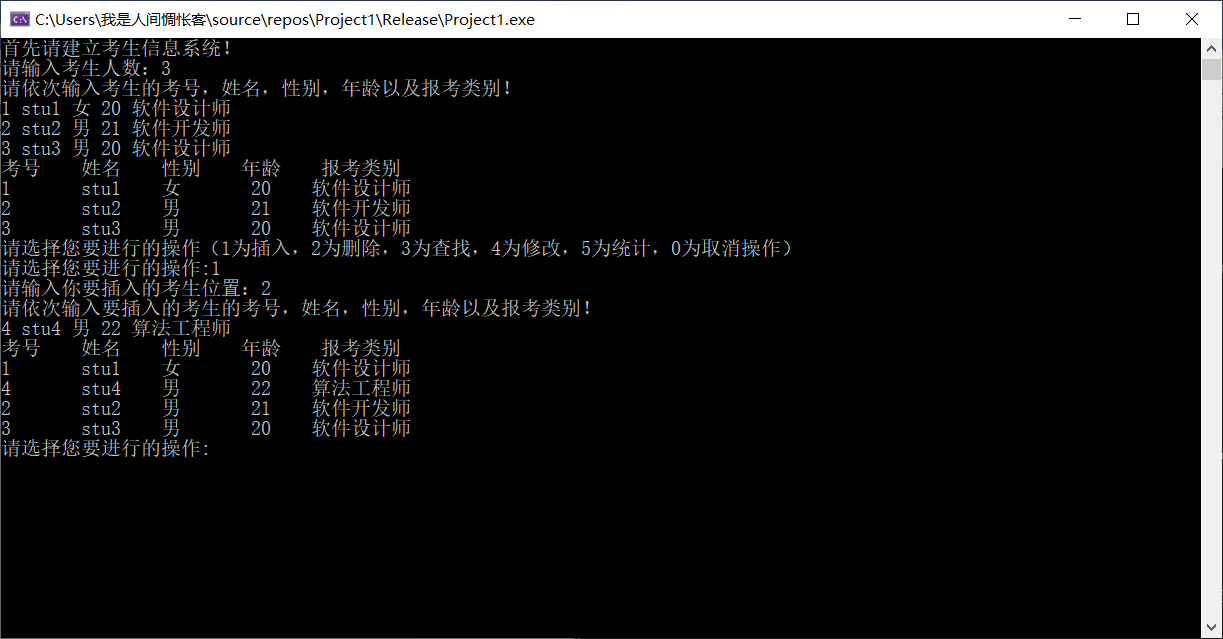
newNode->link = current->link;

current->link = newNode;

listLength++;

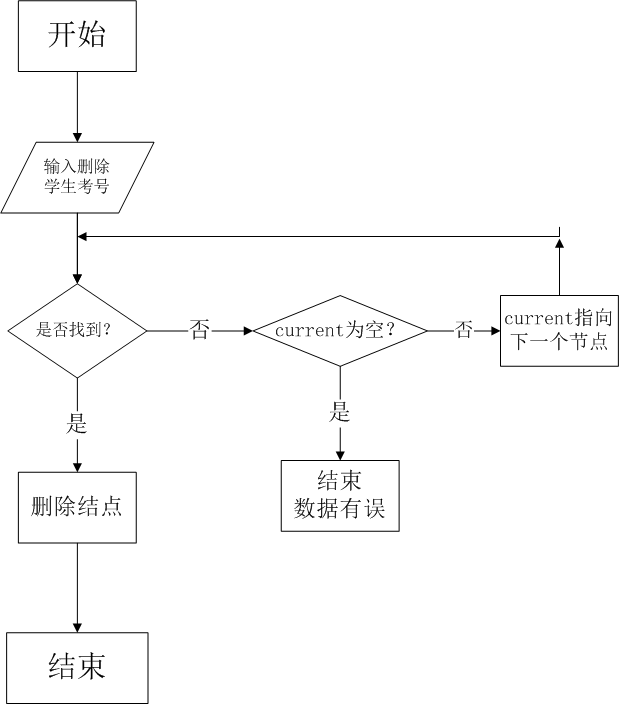
}

### 3.1.3 插入功能截屏示例



## 3.2 删除功能的实现

### 3.2.1 删除功能流程图



### 3.2.2 删除功能核心代码

void List::Delete(int position)

{

if (position > listLength)//位置不存在

{

cout << "该位置不存在" << endl;

return;

}

LinkNode\* current = first, \* temp;

for (int i = 0; i < position - 1; i++)//找到位置

{

current = current->link;

}

//删除操作

temp = current->link;

cout << "你删除的考生信息是：";

cout << temp->studentID << " "

<< temp->studentName << " "

<< temp->studentGender << " "

<< temp->studentAge << " "

<< temp->studentCategory << " "

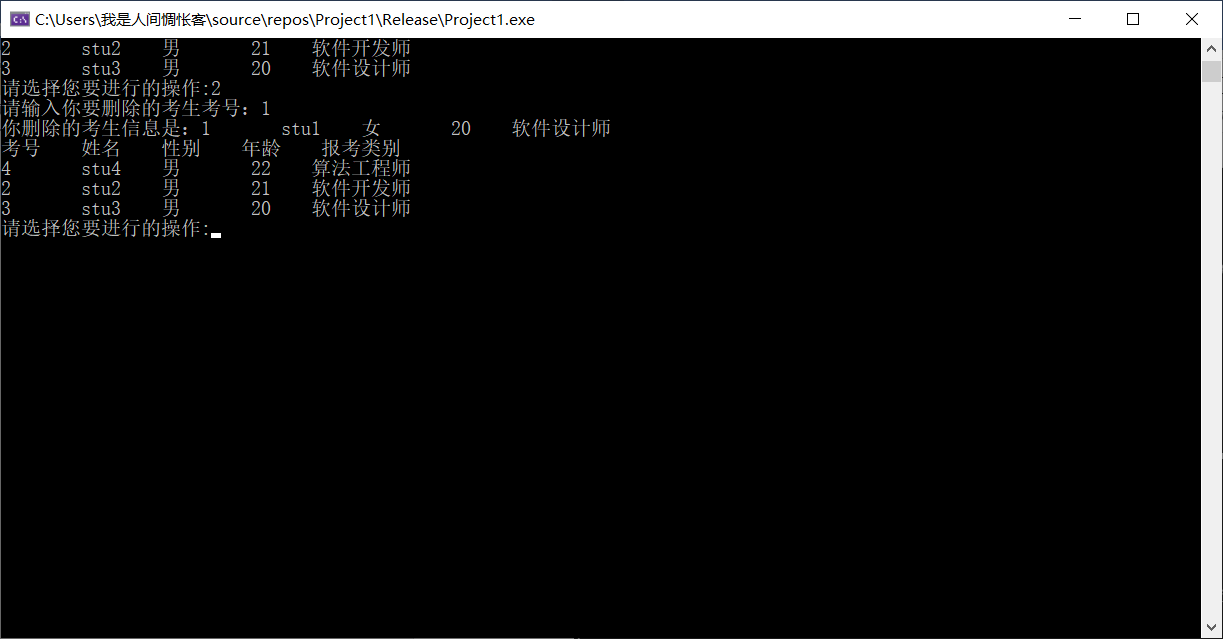
<< endl;

current->link = temp->link;

delete temp;

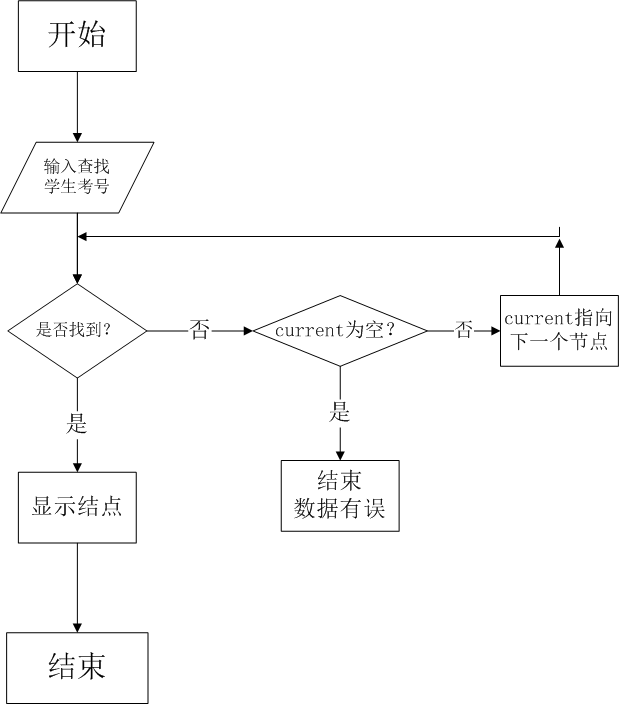
}

### 3.2.3 删除功能截屏示例



## 3.3 查找功能的实现

### 3.3.1 查找功能流程图



### 3.3.2 查找功能核心代码

void List::Find(int ID) const

{

LinkNode\* current = first->link;

while (current != nullptr && current->studentID != ID)

{

current = current->link;

}//查找并输出

if (current != nullptr)

{

cout << current->studentID << " "

<< current->studentName << " "

<< current->studentGender << " "

<< current->studentAge << " "

<< current->studentCategory << " "

<< endl;

}

//找不到

else

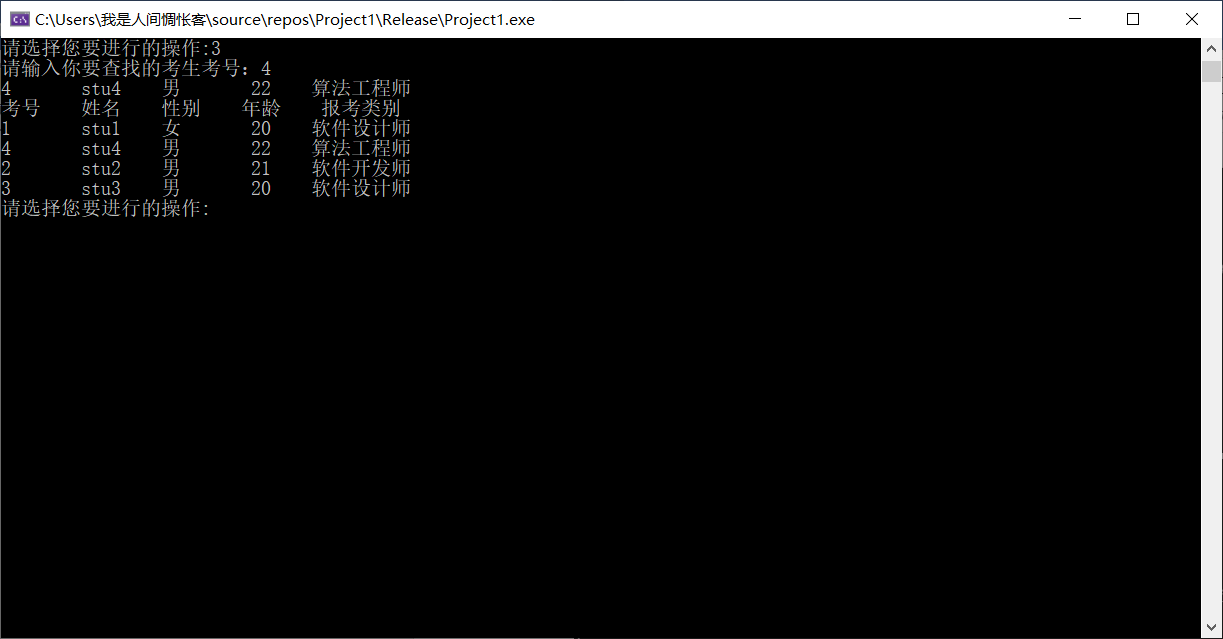
{

cout << "该考生号不存在" << endl;

}

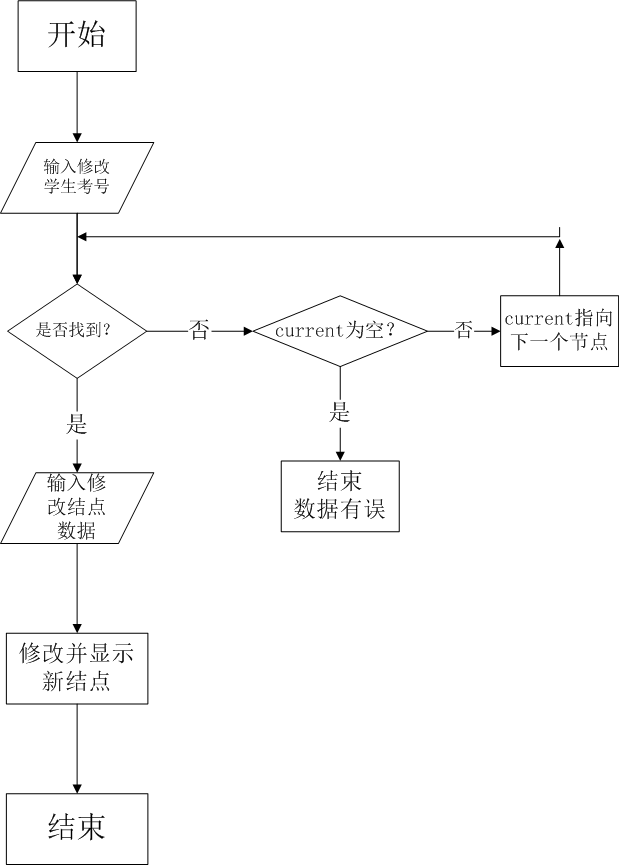
}

### 3.3.3 查找功能截图示例



## 3.4 修改功能的实现

### 3.4.1 修改功能流程图



### 3.4.2 修改功能核心代码

void List::Change(int ID)

{

LinkNode\* current = first->link;

while (current != nullptr && current->studentID != ID)

{

current = current->link;

}//找到该位置并修改

if (current != nullptr)

{

cout << "请依次输入考生的考号，姓名，性别，年龄以及报考类别！" << endl;

cin >> current->studentID >> current->studentName >> current->studentGender >> current->studentAge >> current->studentCategory;

}//找不到该位置

else

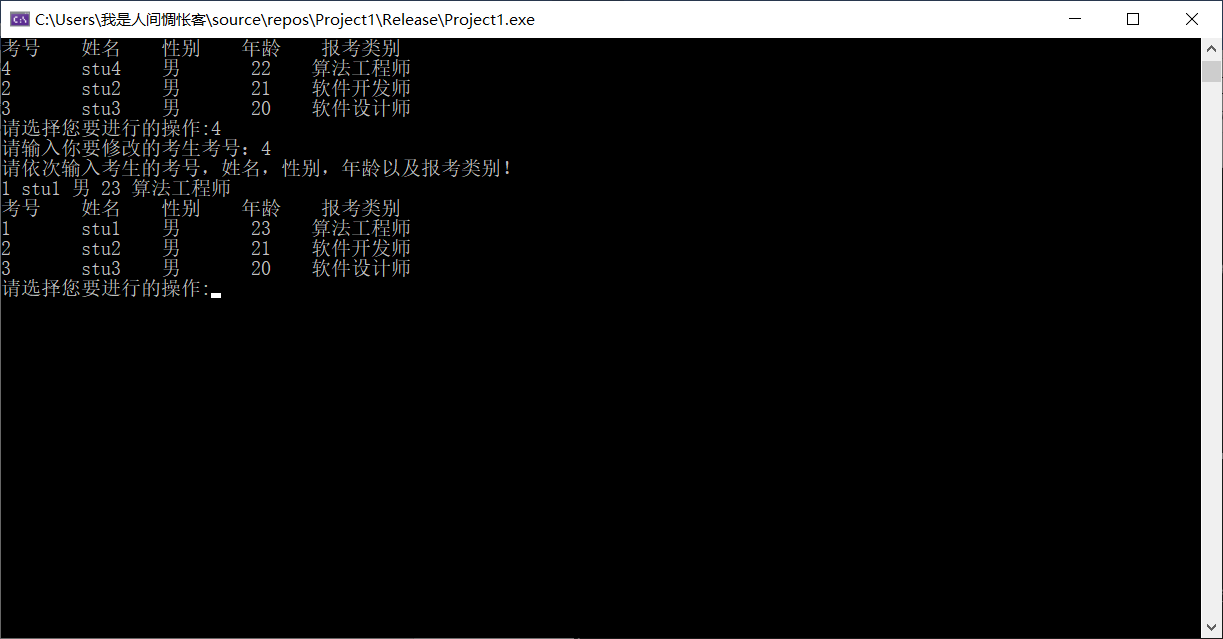
{

cout << "该考生号不存在"<< endl;

}

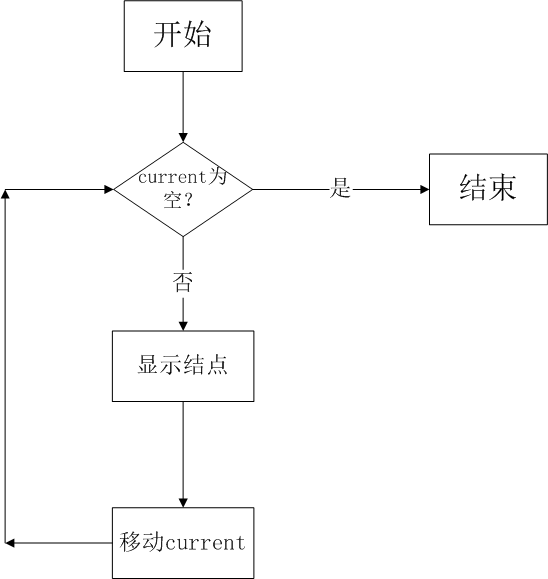
}

### 3.4.3 修改功能截屏示例



## 3.5 输出功能的实现

### 3.5.1 输出功能流程图



### 3.5.2 输出功能核心代码

void List::Display() const

{

LinkNode\* current;

current = first->link;

cout <<"考号"<< " "

<<"姓名"<< " "

<<"性别"<< " "

<<"年龄"<< " "

<<"报考类别"<< " "

<< endl;

while (current != nullptr)//依次输出

{

cout << current->studentID << " "

<< current->studentName << " "

<< current->studentGender << " "

<< current->studentAge << " "

<< current->studentCategory << " "

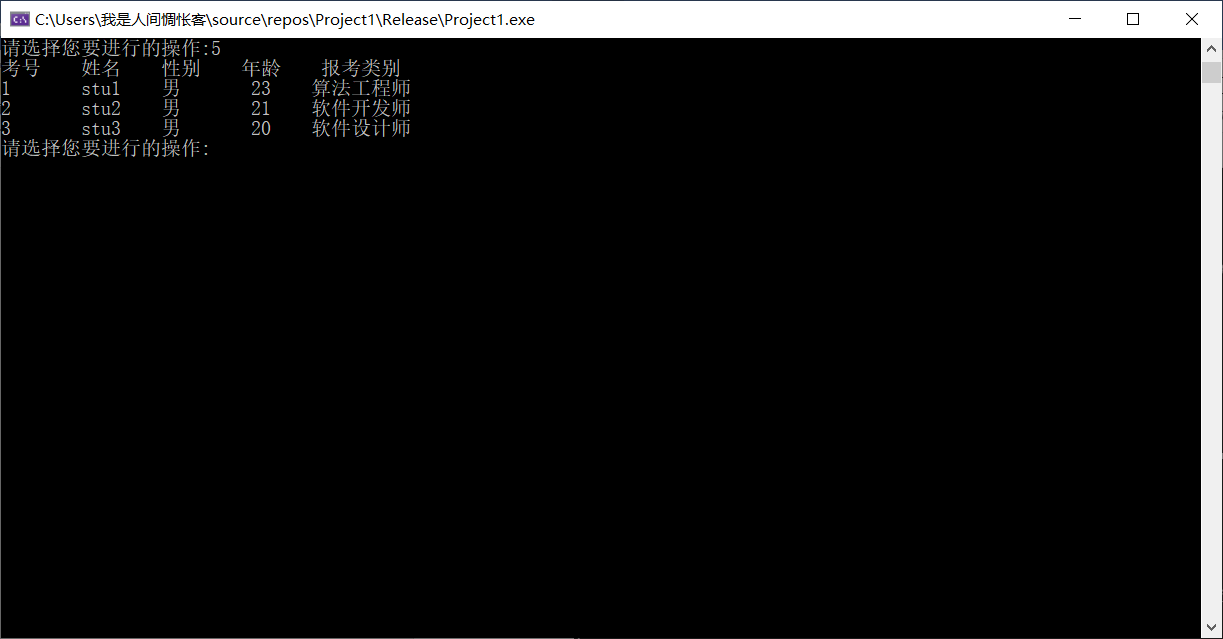
<< endl;

current = current->link;

}

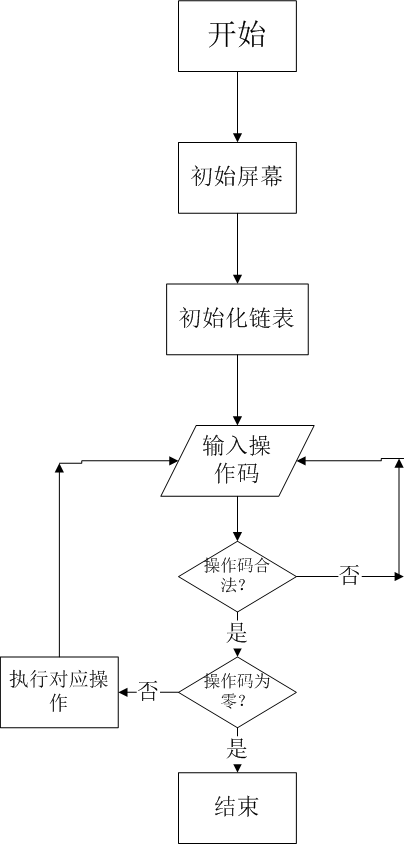
}

### 3.5.3 输出功能截屏示例



## 3.6 总体系统的实现

### 3.6.1 总体系统流程图



### 3.6.2 总体系统核心代码

int main()

{

int number,p,operation;

cout << "首先请建立考生信息系统！" << endl;

cout << "请输入考生人数：";

cin >> number;

while (number < 0)//如果输入不规范

{

cout << "请输入一个自然数"<< endl;

cout << "请输入考生人数：";

cin >> number;

}

cout << "请依次输入考生的考号，姓名，性别，年龄以及报考类别！" << endl;

List mylist;

mylist.Input(number);//建立链表

mylist.Display();

cout << "请选择您要进行的操作（1为插入，2为删除，3为查找，4为修改，5为统计，0为取消操作）" << endl;

cout << "请选择您要进行的操作:";

cin >> operation;

while (operation != 0)//分支选择

{

if (operation == 1)

{

cout << "请输入你要插入的考生位置：";

cin >> p;

mylist.Insert(p);

mylist.Display();

}

if (operation == 2)

{

cout << "请输入你要删除的考生考号：";

cin >> p;

mylist.Delete(p);

mylist.Display();

}

if (operation == 3)

{

cout << "请输入你要查找的考生考号：";

cin >> p;

mylist.Find(p);

mylist.Display();

}

if (operation == 4)

{

cout << "请输入你要修改的考生考号：";

cin >> p;

mylist.Change(p);

mylist.Display();

}

if (operation == 5)

{

mylist.Display();

}

if (operation > 5 || operation < 0)//不在选择中

{

cout << "不存在该操作" << endl;

}

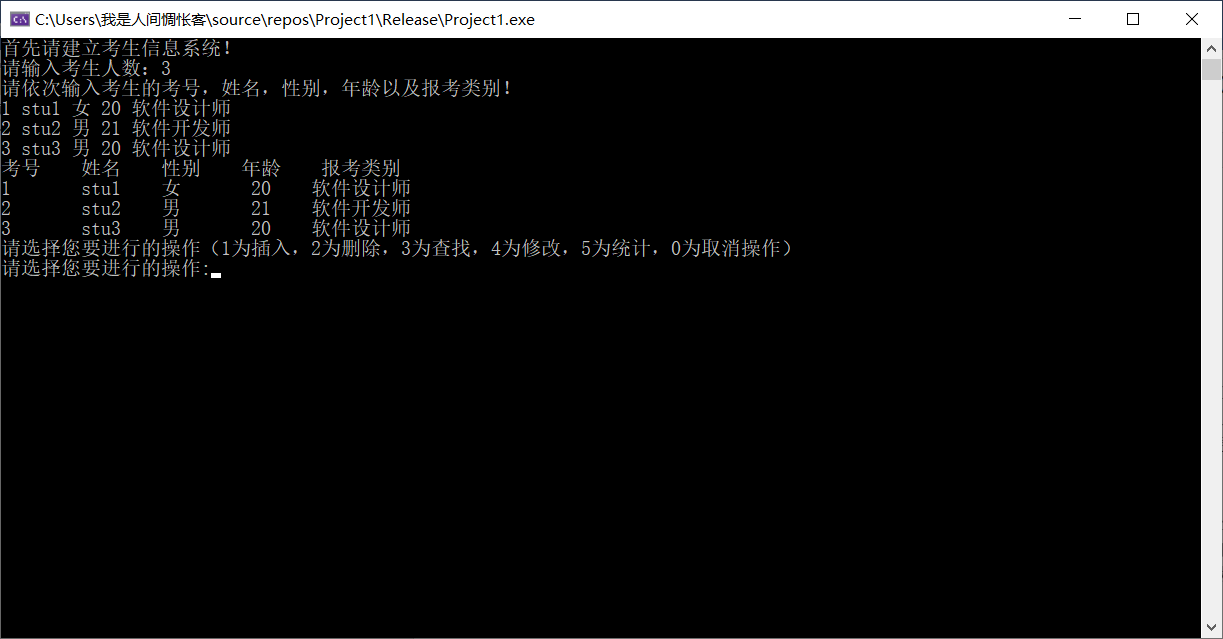
cout << "请选择您要进行的操作:";

cin >> operation;

}

}

### 3.6.3 总体系统截屏示例



# 4 测试

## 4.1 功能测试

### 4.1.1 插入功能测试

**测试用例**：3 stu3 男 22 网络工程师

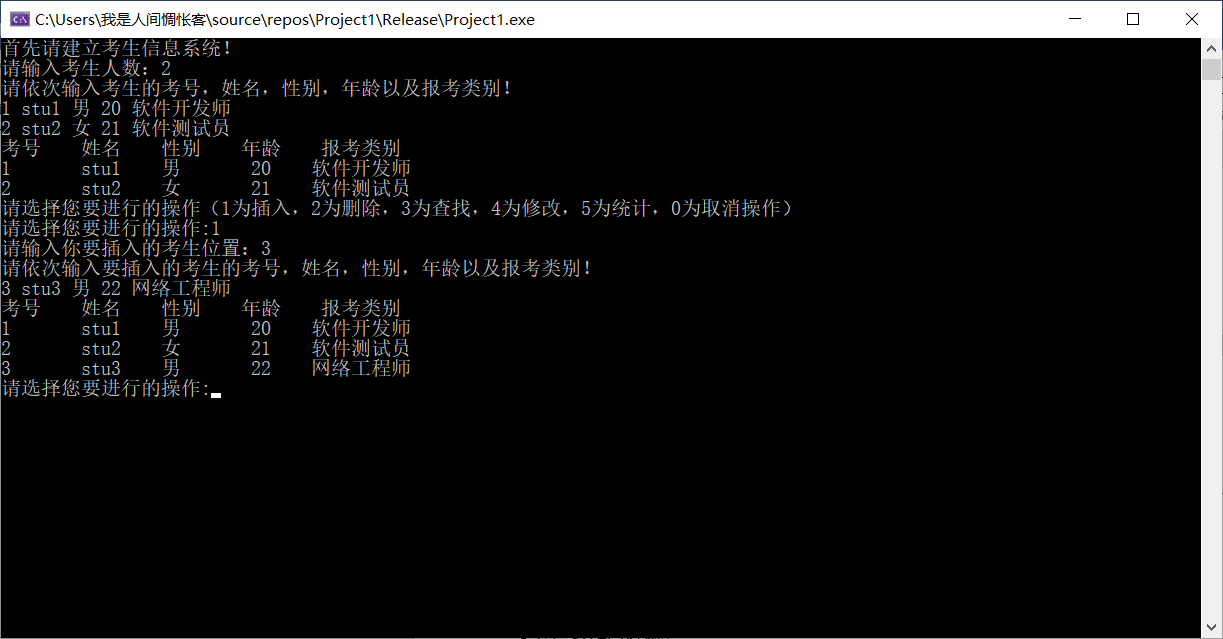
**预期结果**：

1 stu1 男 20 软件开发师

2 stu2 女 21 软件测试员

3 stu3 男 22 网络工程师

**实验结果**



### 4.1.2 删除功能测试

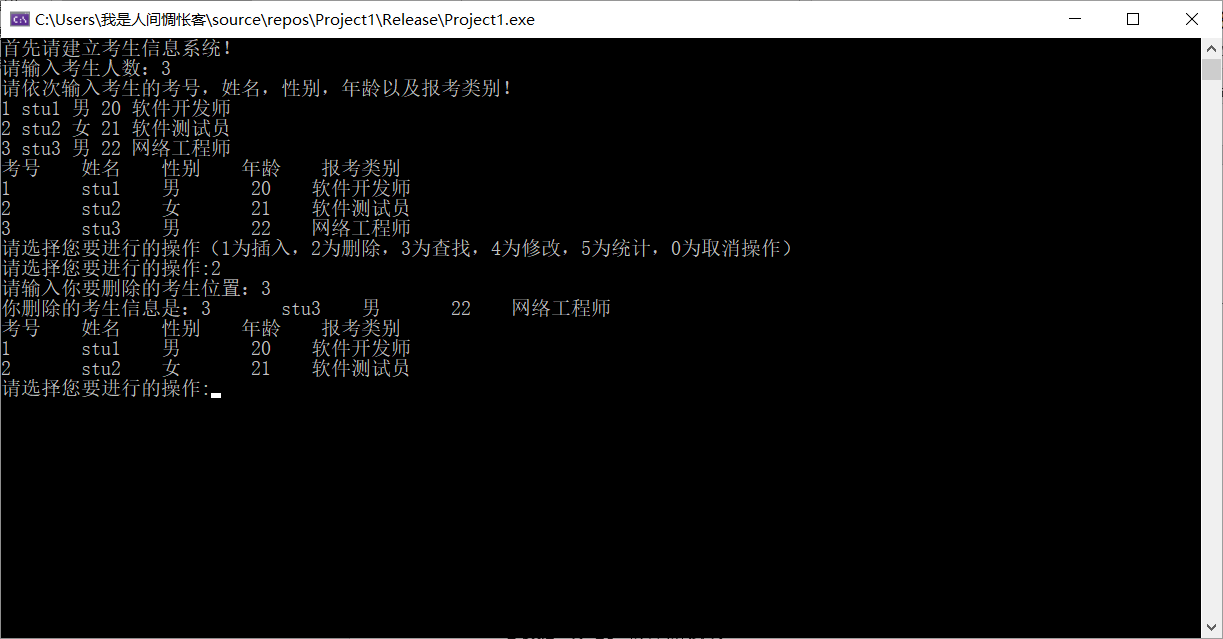
**测试用例：**删除位置为3的考生

**预期结果：**

1 stu1 男 20 软件开发师

2 stu2 女 21 软件测试员

**实验结果：**



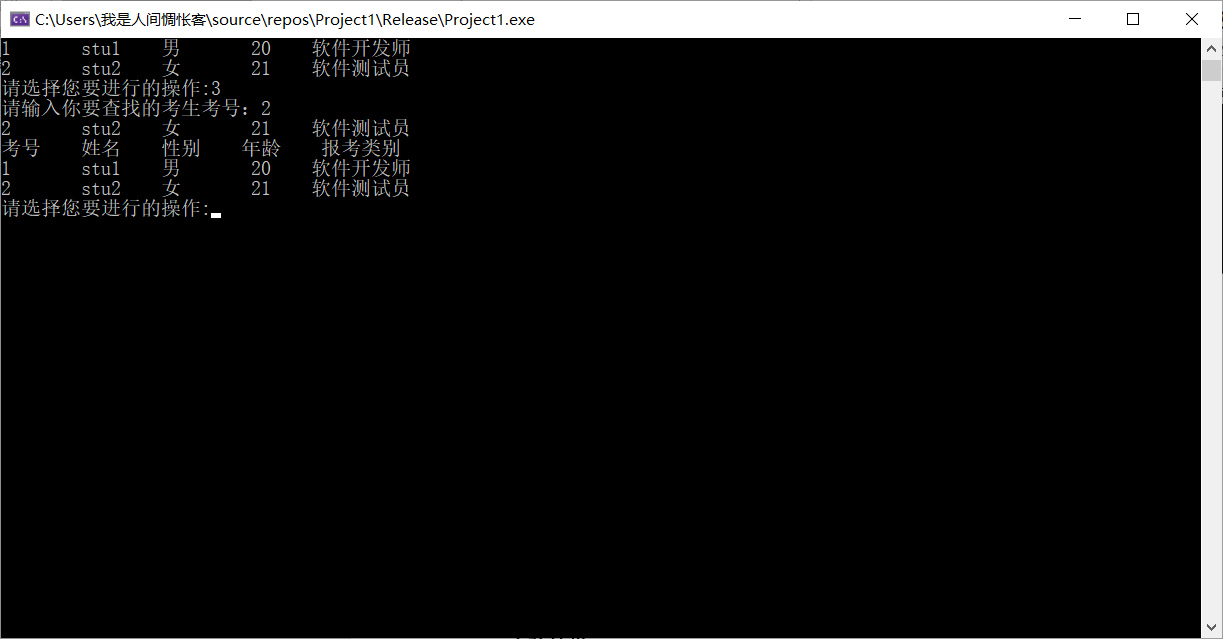
### 4.1.3 查找功能测试

**测试用例：**查找考号为2的考生

**预期结果：**

2 stu2 女 21 软件测试员

**实验结果：**



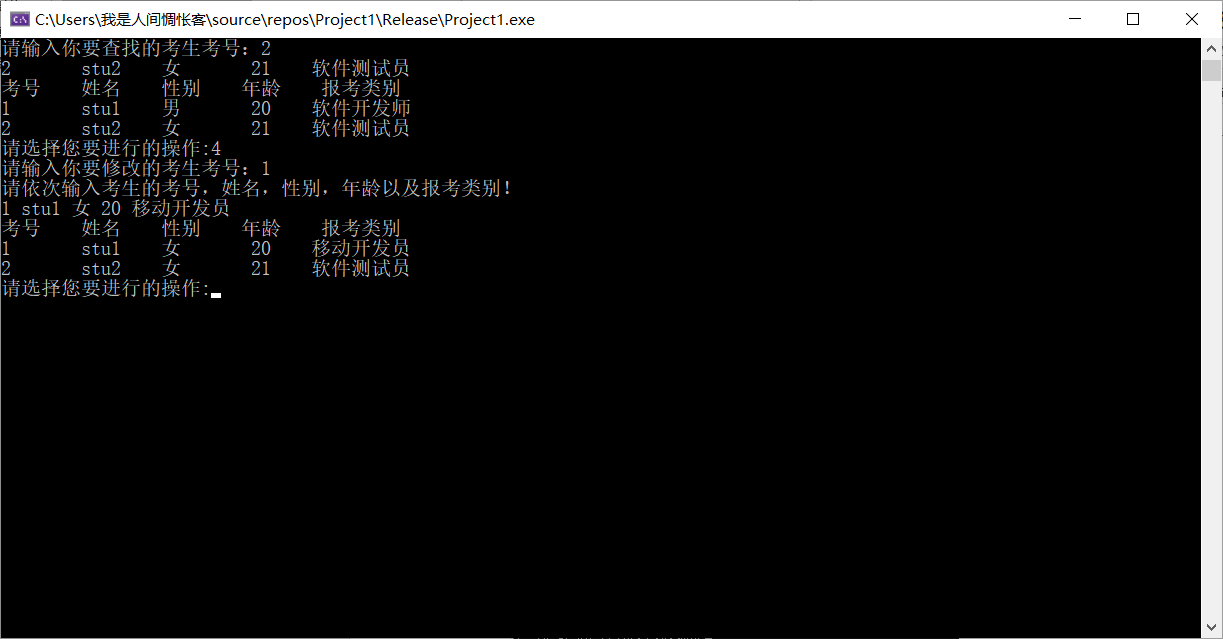
### 4.1.4 修改功能测试

**测试用例：**将考号1修改为性别女，年龄20，报考种类移动开发员。

**预期结果：**

1 stu1 女 20 移动开发员

**实验结果：**



### 4.1.5 统计功能测试

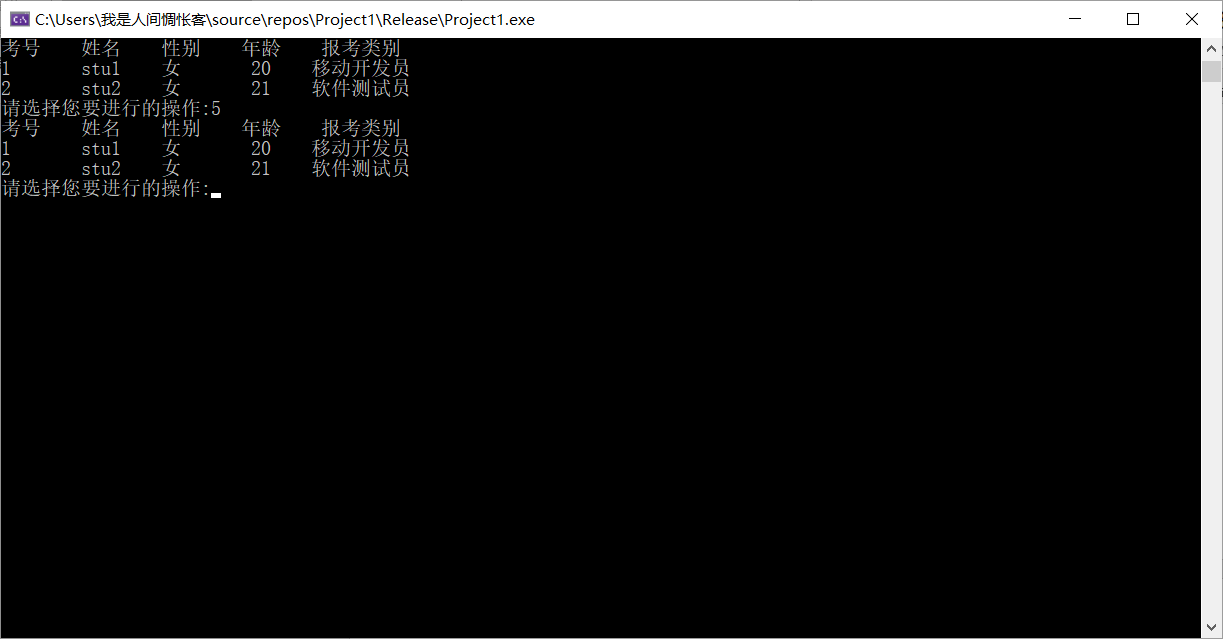
**测试用例：**统计当前数据

**预期结果：**

1 stu1 女 20 移动开发员

2 stu2 女 21 软件测试员

**实验结果：**



## 4.2 边界测试

### 4.2.1 初始化无输入数据

**测试用例：**初始输入数据不正确

**预期结果：**给出错误提示，程序运行正常不崩溃。

**实验结果：**

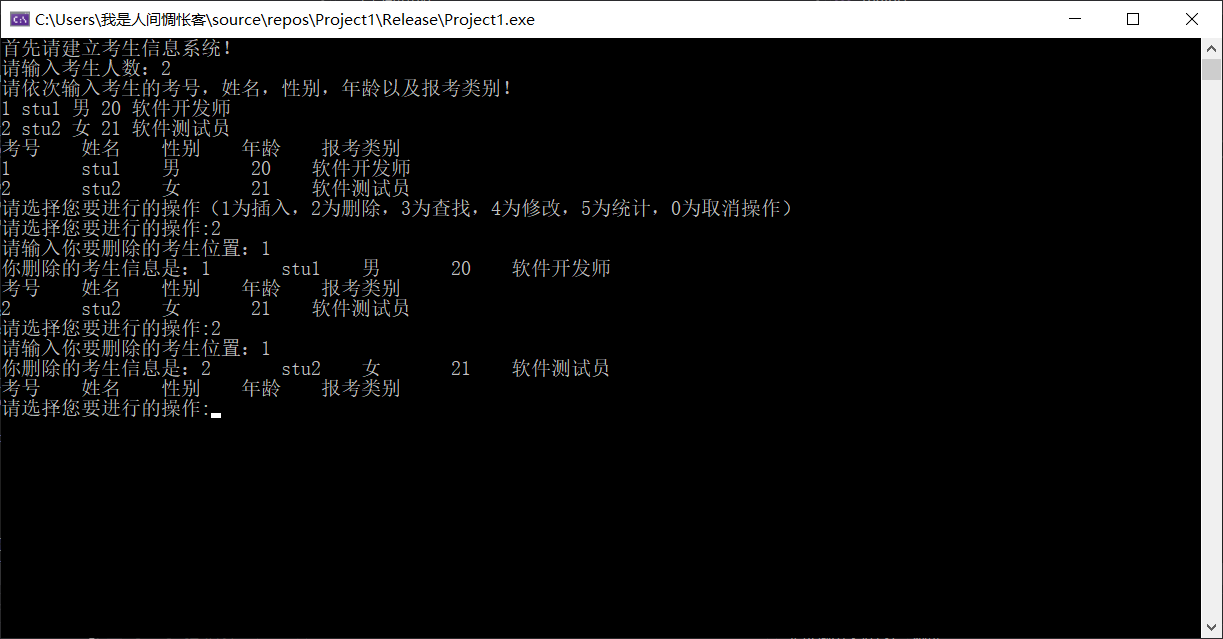


### 4.2.2 删除后链表为空

**测试用例：**删除前链表只有一个结点，删除后链表为空

**预期结果：**程序正常运行，不崩溃。

**实验结果：**



## 4.3 出错测试

### 4.3.1 操作码错误

**测试用例：**输入操作码错误

**预期结果：**程序给出提示信息，程序正常运行不崩溃。

**实验结果：**

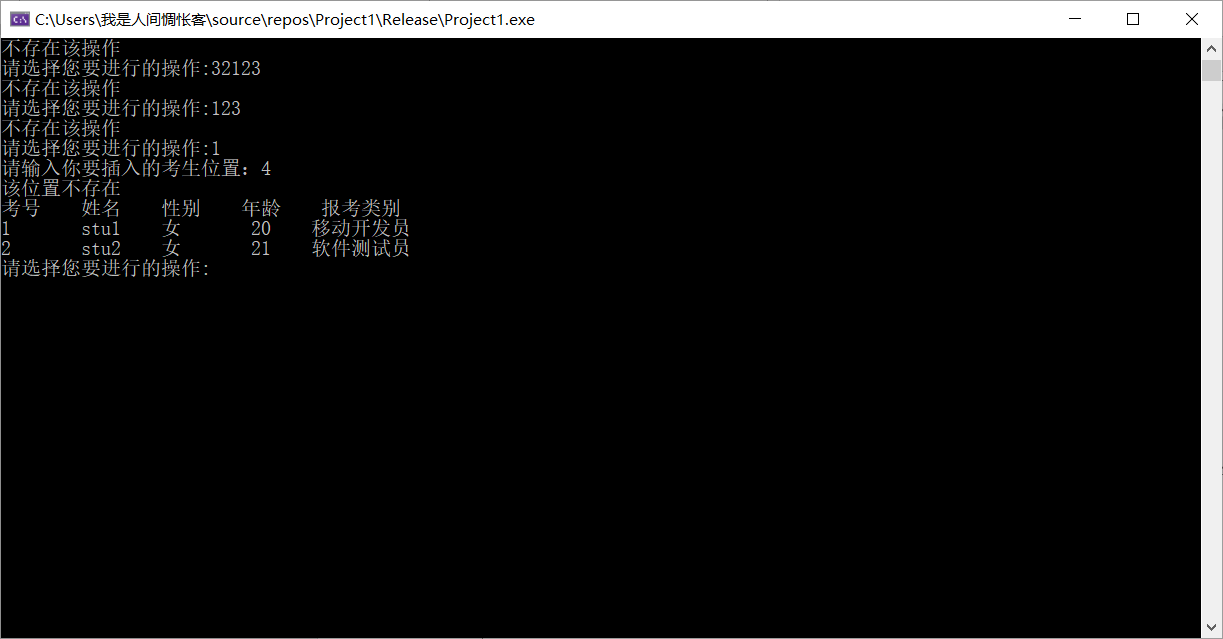


### 4.3.2 插入位置不存在

**测试用例：**插入不存在的位置

**预期结果：**程序给出错误信息，程序正常运行不崩溃。

**实验结果：**

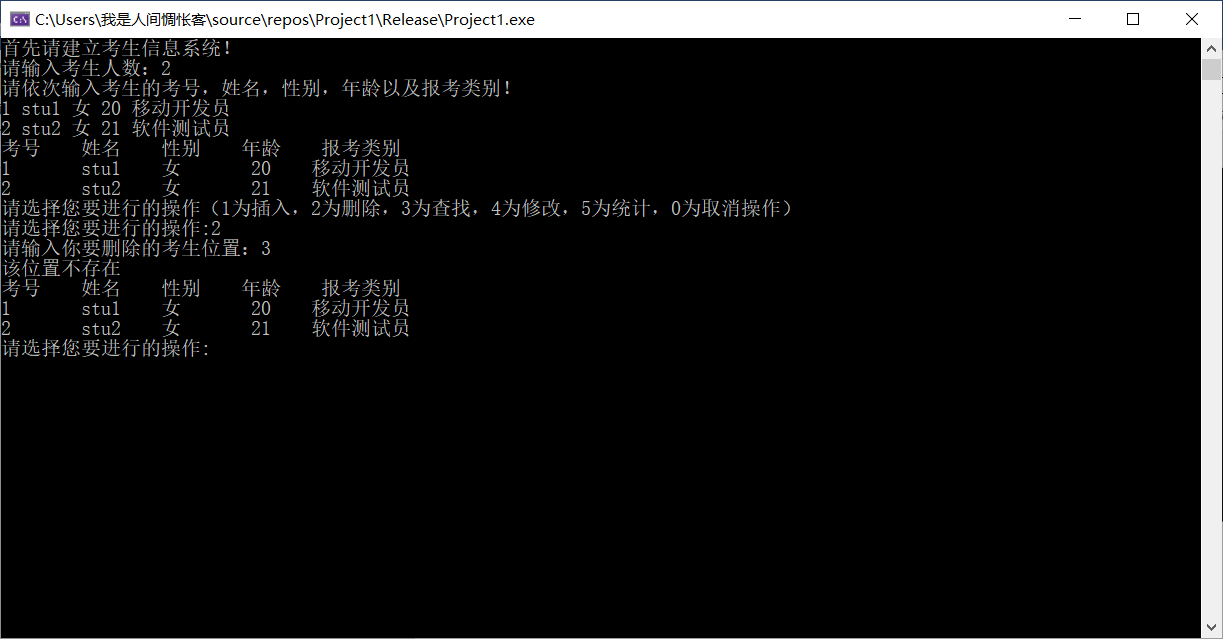


### 4.3.4 删除位置不存在

**测试用例：**要删除的位置不存在

**预期结果：**程序给出错误信息，程序正常运行不崩溃。

**实验结果：**

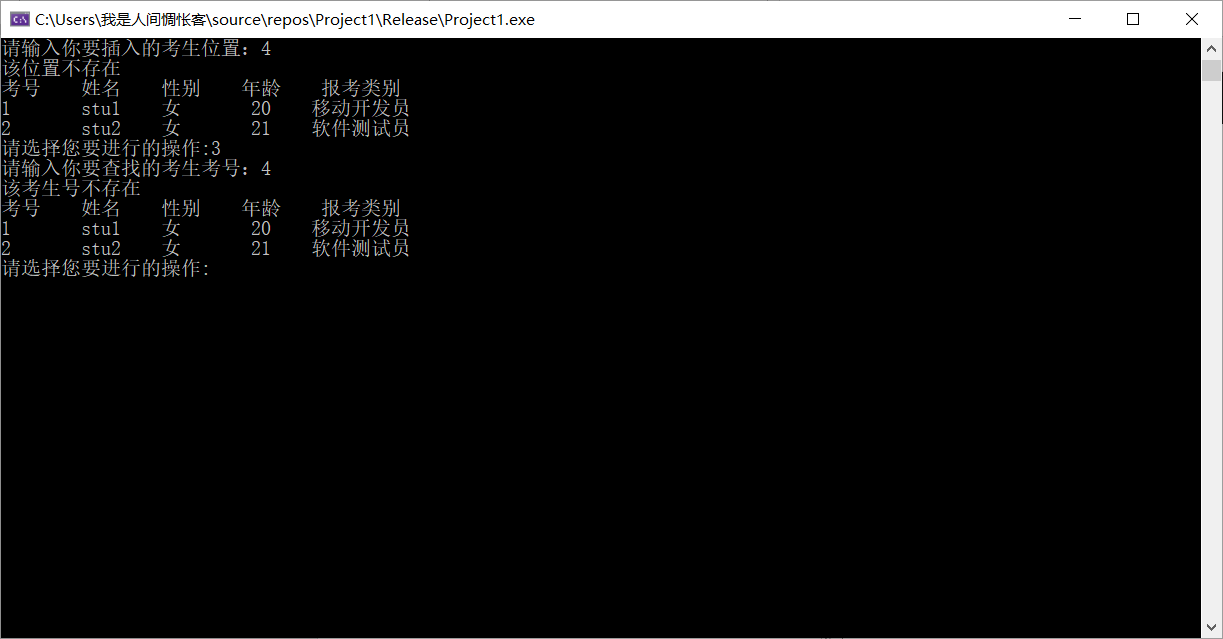


### 4.3.5 查找考号不存在

**测试用例：**要查找的考号不存在

**预期结果：**程序给出错误信息，程序正常运行不崩溃。

**实验结果：**



### 4.3.6 修改考号不存在

**测试用例：**要修改的考号不存在

**预期结果：**程序给出错误信息，程序正常运行不崩溃。

**实验结果：**

