项目说明文档

数据结构课程设计

——考试报名系统

作 者 姓 名： 罗佳瑞

学 号： 1952528

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](#_Toc12903)

[1.1 背景分析 1](#_Toc3048)

[1.2 功能分析 1](#_Toc5111)

[2 设计 2](#_Toc8260)

[2.1 数据结构设计 2](#_Toc10085)

[2.2 类结构设计 2](#_Toc405)

[2.3 成员与操作设计 2](#_Toc2360)

[2.4 系统设计 3](#_Toc16616)

[3 实现 4](#_Toc14974)

[3.1 插入功能的实现 4](#_Toc1655)

[3.1.1 插入功能流程图 4](#_Toc26699)

[3.1.2 插入功能核心代码 4](#_Toc25029)

[3.1.3 插入功能截屏示例 5](#_Toc2023)

[3.2 删除功能的实现 6](#_Toc20351)

[3.2.1 删除功能流程图 6](#_Toc13742)

[3.2.2 删除功能核心代码 7](#_Toc19312)

[3.2.3 删除功能截屏示例 7](#_Toc29616)

[3.3 查找功能的实现 8](#_Toc14091)

[3.3.1 查找功能流程图 8](#_Toc3732)

[3.3.2 查找功能核心代码 8](#_Toc24925)

[3.3.3 查找功能截图示例 9](#_Toc25161)

[3.4 修改功能的实现 10](#_Toc9053)

[3.4.1 修改功能流程图 10](#_Toc17802)

[3.4.2 修改功能核心代码 11](#_Toc23543)

[3.4.3 修改功能截屏示例 11](#_Toc23108)

[3.5 统计功能的实现 12](#_Toc12620)

[3.5.1 统计功能流程图 12](#_Toc12890)

[3.5.2 统计功能核心代码 12](#_Toc26864)

[3.5.3 统计功能截屏示例 13](#_Toc29532)

[3.6 总体系统的实现 14](#_Toc30521)

[3.6.1 总体系统流程图 14](#_Toc14639)

[3.6.2 总体系统核心代码 15](#_Toc9686)

[3.6.3 总体系统截屏示例 16](#_Toc14007)

[4 测试 17](#_Toc29765)

[4.1 功能测试 17](#_Toc19707)

[4.2 边界测试 17](#_Toc10472)

[4.2.1 初始化无输入数据 17](#_Toc21608)

[4.2.2 删除头结点 17](#_Toc32705)

[4.2.3 删除后链表为空 17](#_Toc28950)

[4.3 出错测试 18](#_Toc3111)

[4.3.1 考生人数错误 18](#_Toc21048)

[4.3.2 操作码错误 18](#_Toc19687)

[4.3.3 插入位置不存在 19](#_Toc32296)

[4.3.4 删除考号不存在 19](#_Toc11952)

[4.3.5 查找考号不存在 20](#_Toc6054)

[4.3.6 修改考号不存在 20](#_Toc22215)

# 1 分析

## 1.1 背景分析

考试报名系统是一个学校不可缺少的部分，对于学校加强考试管理有极其重要的作用。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。随着学生数量和考试数量的日益庞大，数据的管理显得极为复杂，传统的手工管理工作量大且容易出错。

而使用计算机在管理考试报名系统方面，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，也是学校走向信息化、科学化、国际化的重要条件之一。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同学们的考试报名信息并且可以予以显示。其次，考试报名系统还应该具有插入、删除、修改功能，以保证同学可以随时更改自己的考试报名情况，以及统计功能便于学校进行统计考试数据，进行考试安排。最后，考试报名系统软件还应该确保软件可以正常打开与关闭关闭。

综上所述，一个考试报名系统至少应该具有输入、输出、插入、删除、修改、统计、退出的功能。

# 2 设计

## 2.1 数据结构设计

如上功能分析所述，该系统要求大量的插入、删除、修改操作，而链表进行插入、删除等操作十分简便，因此考虑使用链表数据结构。同时，为了实现简易，在第一个结点之前附加一个头结点，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。

## 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（Node）与链表类（LinkedList），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class描述链表结点类（Node），这样使得链表类（LinkedList）可以访问链表结点。

## 2.3 成员与操作设计

**链表结点类（Node）**

**公有成员：**

string numble; //表示学生考号

string name;//表示学生姓名

string sex;//表示学生性别

string age;//表示学生年龄

string sort;//表示学生报考种类

Node \*next;//链表结点的指针域

**公有操作：**

Node::Node();//默认构造函数

Node::Node(string n,string na,string s,string a,string so);//构造函数

**链表类（LinkedList）**

**私有成员：**

Node \*head;//链表的头指针

**公有操作：**

LinkedList::Linkedlist(); //构造函数

void LinkedList::LinkedList::create(int num);//创建链表

void LinkedList::LinkedList::print();//打印链表

int LinkedList::search(string n);//按考号查找位置函数

Node\* LinkedList::locate(int n);//定位函数

bool LinkedList::insert(); //插入操作

bool LinkedList::find();//查找操作

bool LinkedList::clear();//删除操作

bool LinkedList::set();//修改操作

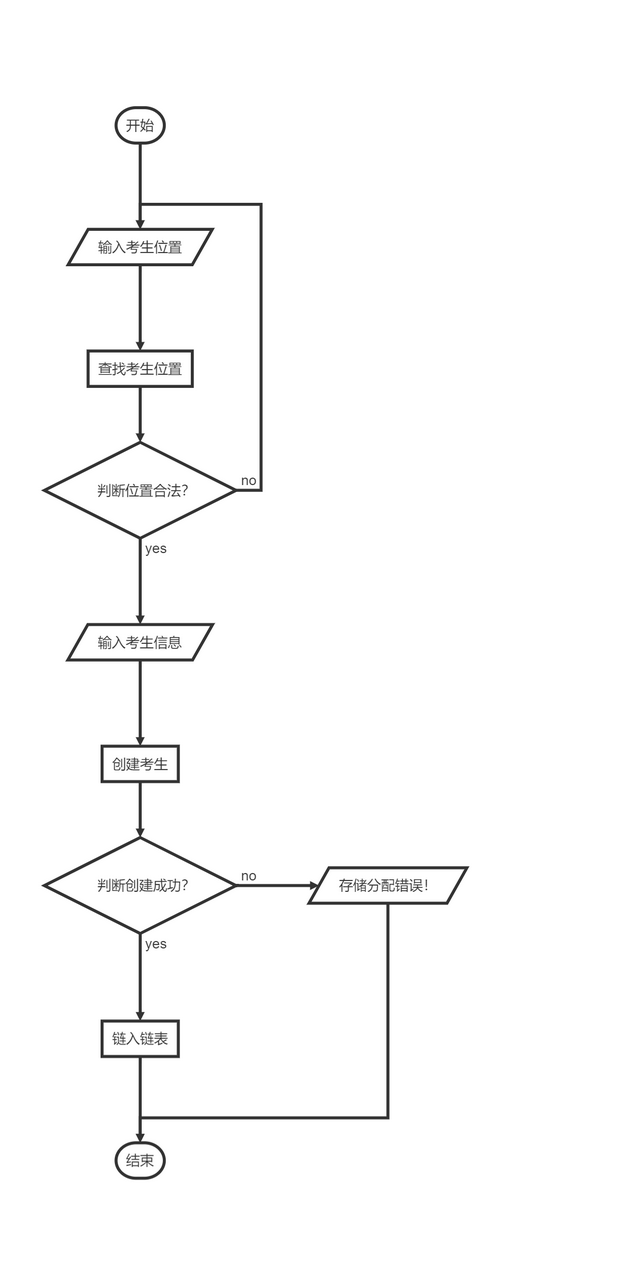
## 2.4 系统设计

系统首先提示用户输入，完成对链表Studentlist的创建和输入数据工作，然后根据用户所输入的操作码（op）执行链表Studentlist对应的成员函数。

# 3 实现

## 3.1 插入功能的实现

### 3.1.1 插入功能流程图



### 3.1.2 插入功能核心代码

bool Linkedlist::insert(){//插入操作

int site;

string n, na, s, a, so;

//位置定位操作

cout << "请输入你要插入的考生位置：";

cin >> site;

Node\* current = locate(site - 1);

if (current == NULL) { cout << "您输入的位置不合法," ;return false; }//插入定位不成功

//创建结点操作

cout << "请依次输入要插入的考生的考号，姓名，性别，年龄，报考类别：";

cin >> n >> na >> s >> a >> so;

Node\* newnode = new Node(n, na, s, a, so);

if (newnode == NULL) { cerr << "存储分配错误！" << endl;exit(1); }//创建不成功

//插入操作

newnode->next = current->next;

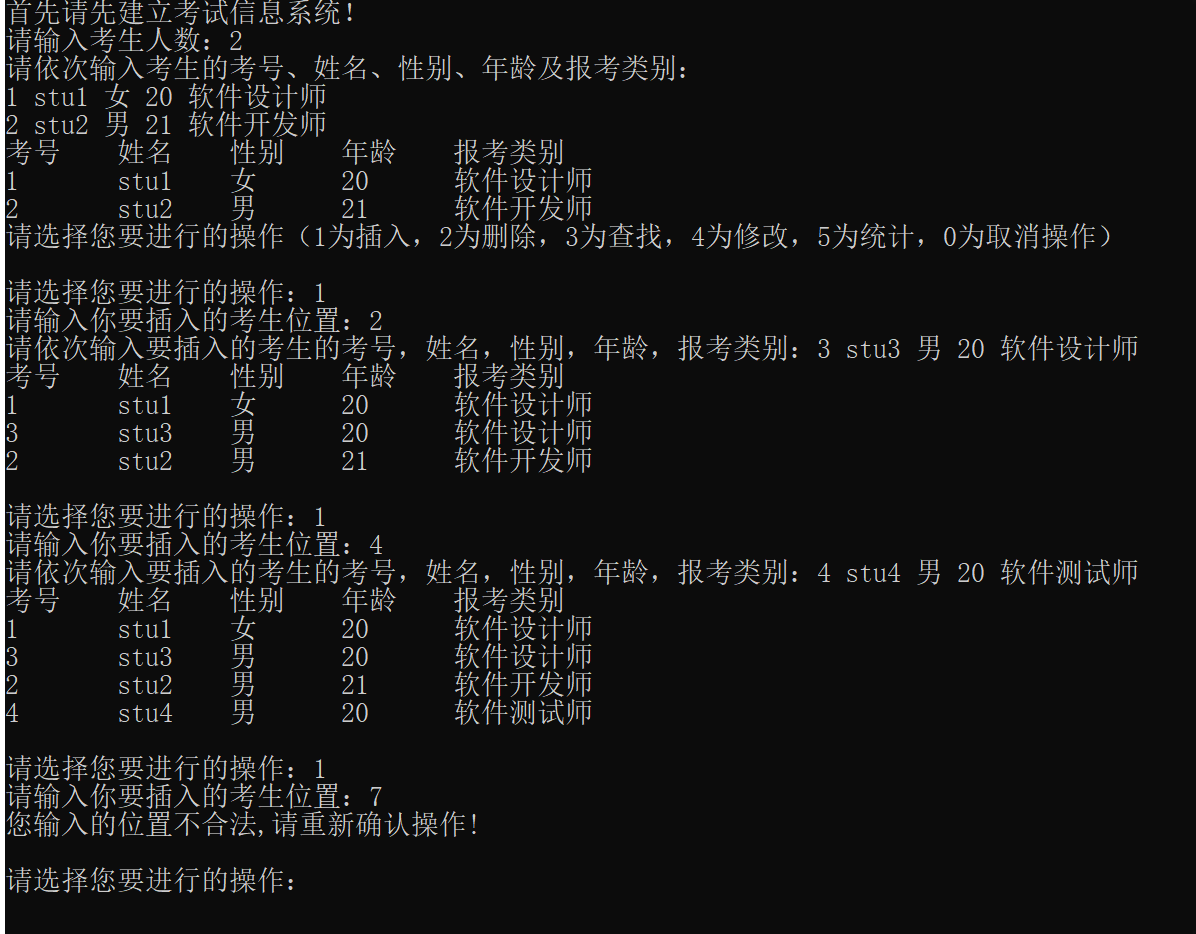
current->next = newnode;

print();

return true;

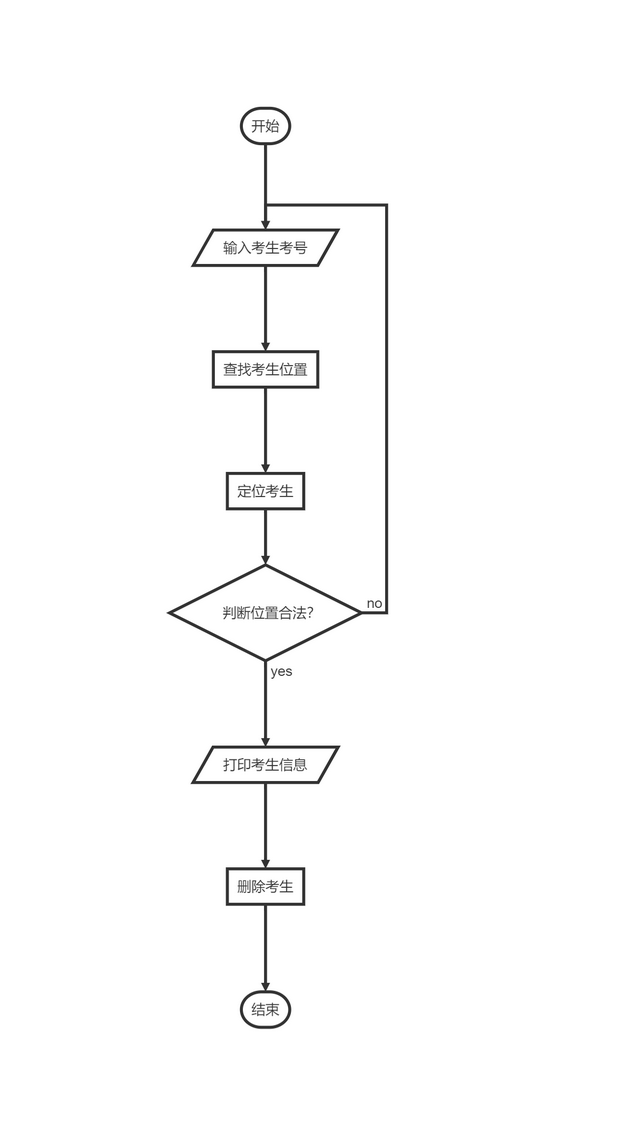
}

### 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

bool Linkedlist::clear(){//删除操作

string n;

cout << "请输入你要删除的考生考号：";

cin >> n;

//定位操作

int site = search(n);

Node\* current = locate(site - 1);

if (current == NULL || current->next == NULL) { cout << "您要删除的考生不存在," ;return false; }//删除定位不成功

//删除操作

Node\* del = current->next;

cout << "你删除的考生信息是：" << del->numble << " " << del->name << " " << del->sex << " " << del->age << " " << del->sort << endl;

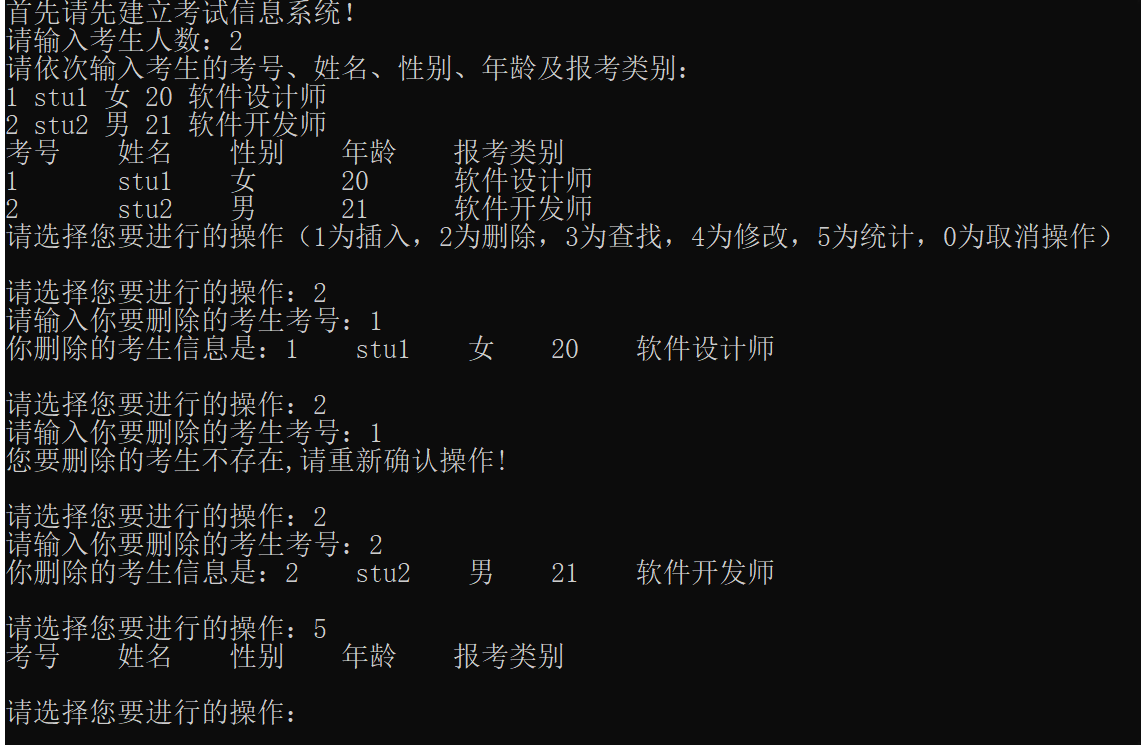
current->next = del->next;

delete del;

return true;

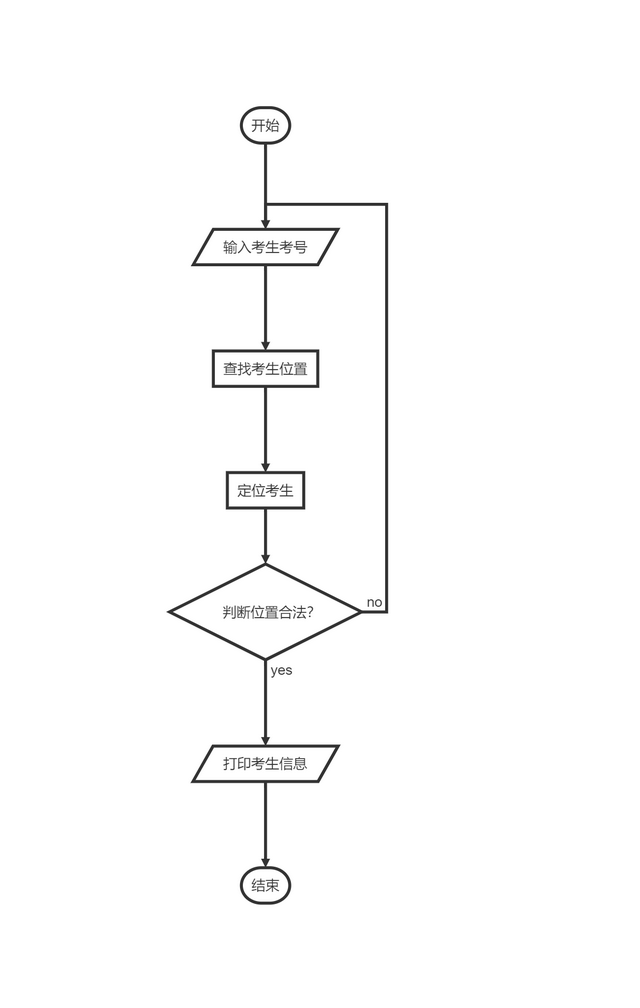
}

### 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

bool Linkedlist::find()//查找操作

{

string n;

cout << "请输入你要查找的考生考号：";

cin >> n;

//定位操作

int site = search(n);

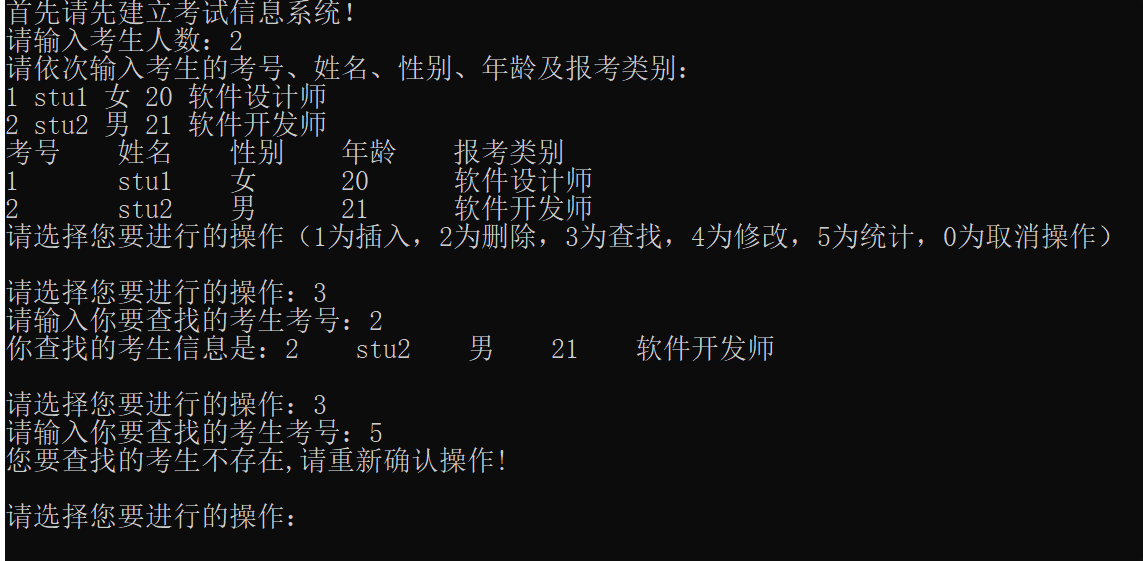
Node\* current = locate(site);

if (current == NULL) { cout << "您要查找的考生不存在," ;return false; }//查找定位不成功

cout << "你查找的考生信息是：" << current->numble << " " << current->name << " " << current->sex << " " << current->age << " " << current->sort << endl;

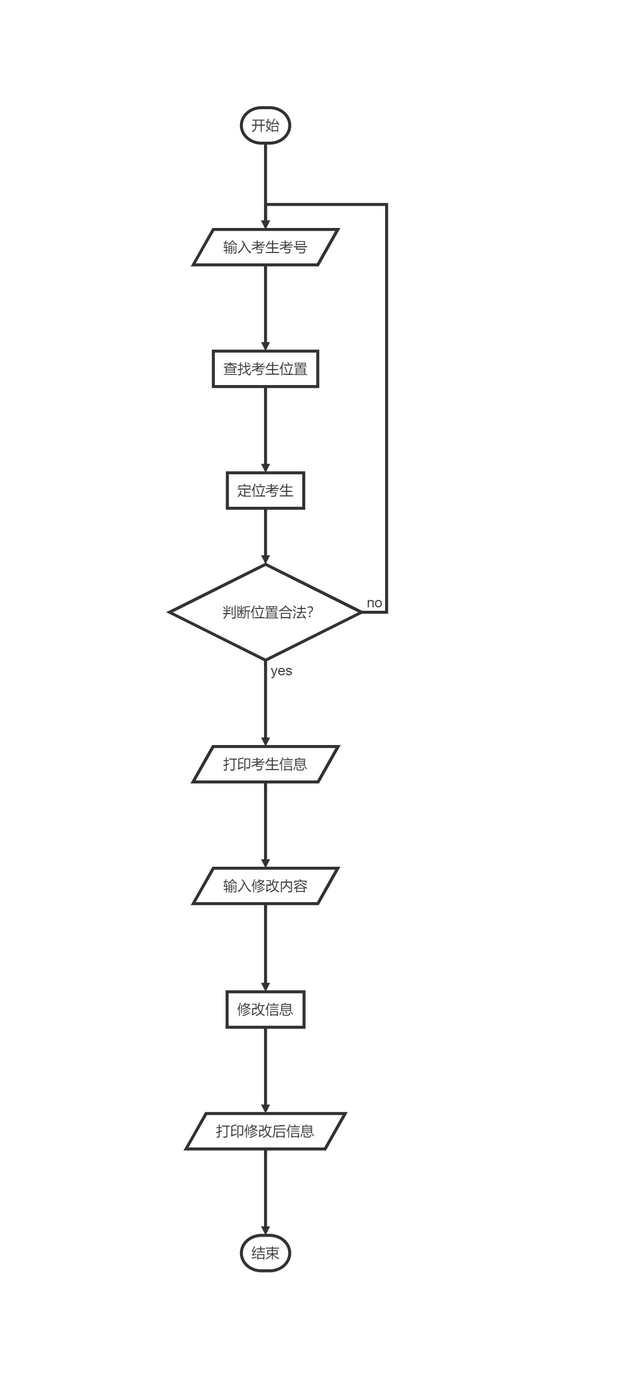
return true;}

### 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

bool Linkedlist::set(){//修改操作

string n;

int change;

cout << "请输入你要修改的考生考号：";

cin >> n;

//修改操作

int site = search(n);

Node\* current = locate(site);

if (current == NULL) { cout << "您要修改的考生不存在," ;return false; }//修改定位不成功

cout << "你要修改的考生信息是：" << current->numble << " " << current->name << " " << current->sex << " " << current->age << " " << current->sort << endl;

cout << "请输入你要修改的内容序号（1为考号，2为姓名，3为性别，4为年龄，5为报考类别）：";

cin >> change;

cout << "请输入你要修改的内容：";

switch (change) {

case 1:cin >> current->numble;break;

case 2:cin >> current->name;break;

case 3:cin >> current->sex;break;

case 4:cin >> current->age;break;

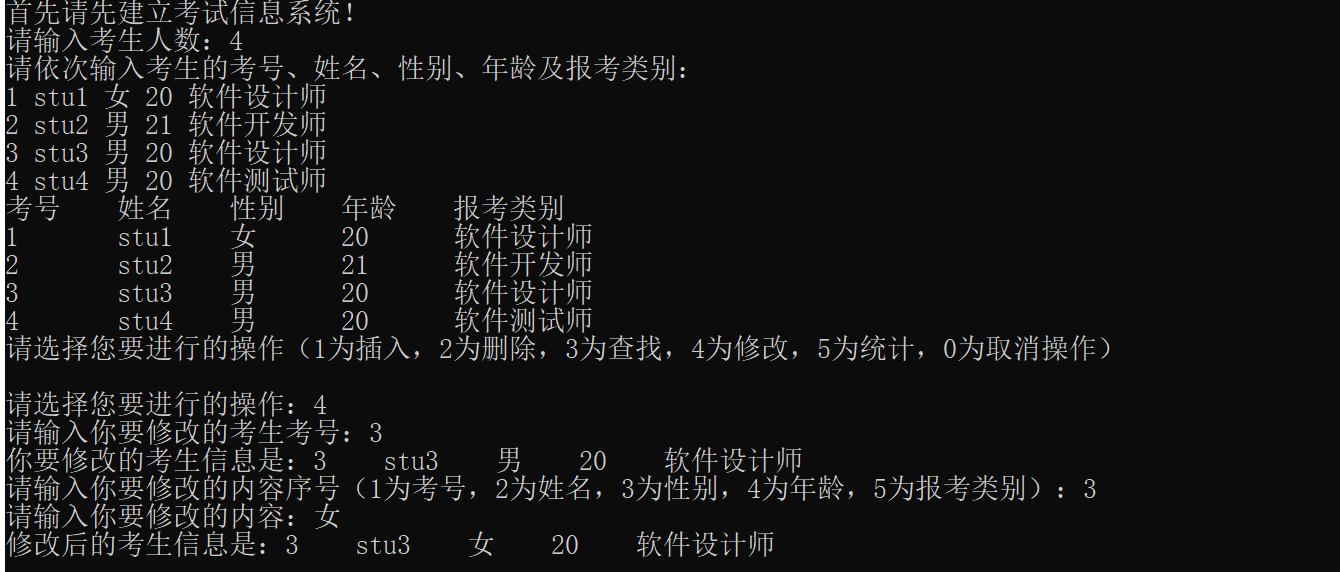
case 5:cin >> current->sort;break;

}

cout << "修改后的考生信息是：" << current->numble << " " << current->name << " " << current->sex << " " << current->age << " " << current->sort << endl;

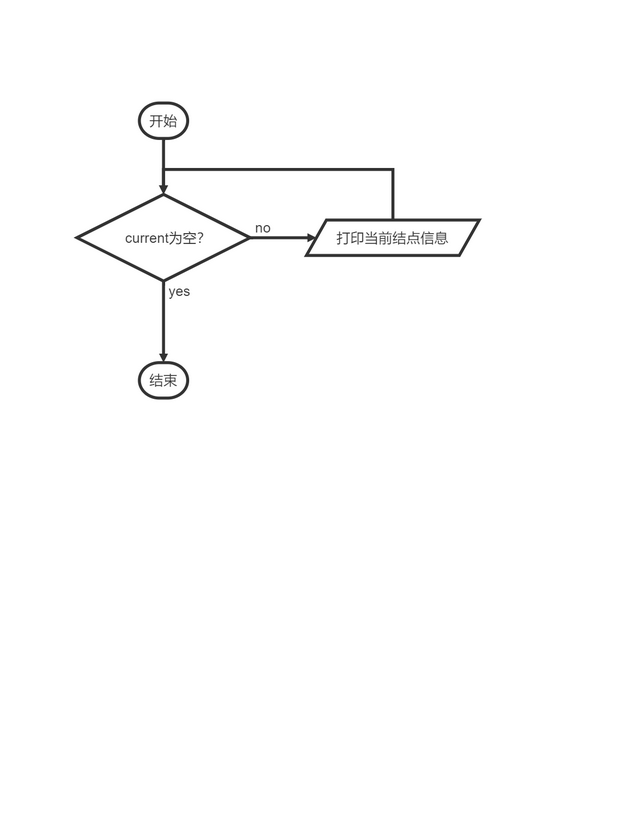
return true;}

### 3.4.3 修改功能截屏示例



## 3.5 统计功能的实现

### 3.5.1 统计功能流程图



### 3.5.2 统计功能核心代码

void Linkedlist::print(){

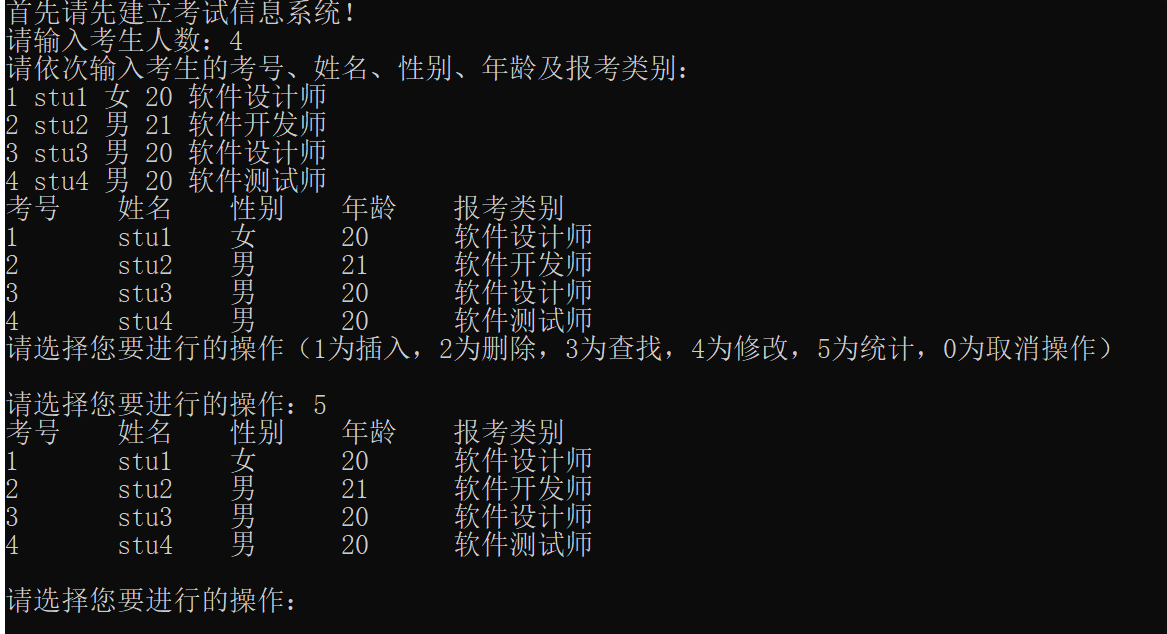
//打印链表

cout << setiosflags(ios::left) << setw(8) << "考号" << setiosflags(ios::left) << setw(8) << "姓名" << setiosflags(ios::left) << setw(8) << "性别" << setiosflags(ios::left) << setw(8) << "年龄" << setiosflags(ios::left) << setw(8) << "报考类别" << endl;

for (Node\* current = head->next;current != NULL;current = current->next)

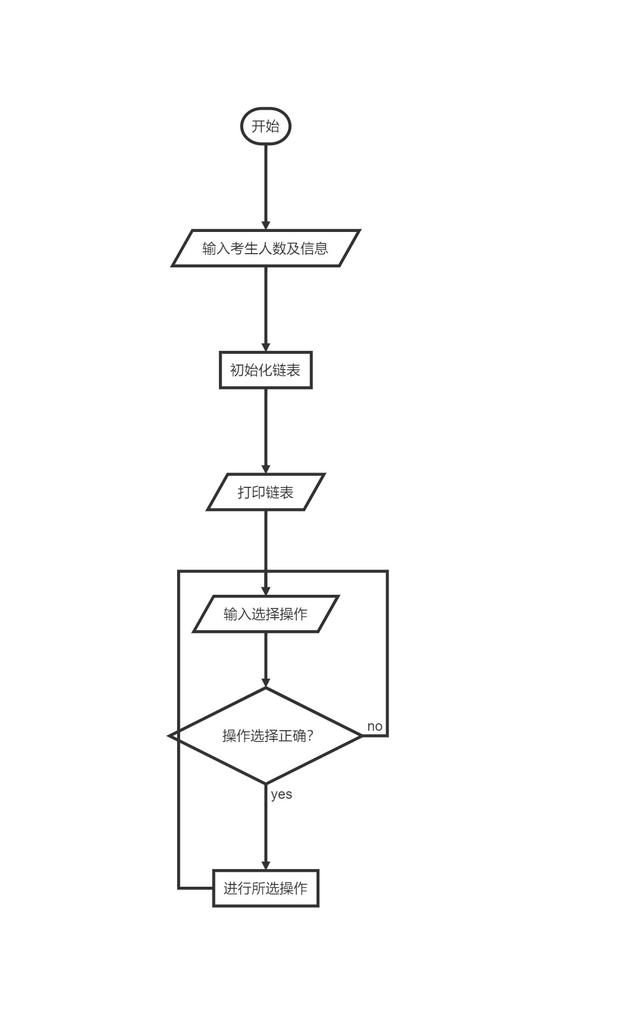
cout << setiosflags(ios::left) << setw(8) << current->numble << setiosflags(ios::left) << setw(8) << current->name << setiosflags(ios::left) << setw(8) << current->sex << setiosflags(ios::left) << setw(8) << current->age << setiosflags(ios::left) << setw(8) << current->sort << endl;}

### 3.5.3 统计功能截屏示例



## 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() {

cout << "首先请先建立考试信息系统！" << endl;

int n=0;

while (n <= 0) {

cout << "请输入考生人数：";

cin >> n;

if (n > 0)

break;

cout << "请输入一个正整数。" << endl;

}

Linkedlist Studentlist;

cout<< "请依次输入考生的考号、姓名、性别、年龄及报考类别：" << endl;

Studentlist.create(n);

Studentlist.print();

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;

int op=1;

while (op)

{

bool wrong = true;

cout <<'\n'<< "请选择您要进行的操作：";

cin >> op;

switch (op) {

case 1:wrong=Studentlist.insert();break;

case 2:wrong = Studentlist.clear();break;

case 3:wrong = Studentlist.find();break;

case 4:wrong = Studentlist.set();break;

case 5:Studentlist.print();break;

case 0:cout << "正在退出……" << endl;break;

default:cout << "抱歉，您想要的操作不存在，请重新选择！" << endl;break;

}

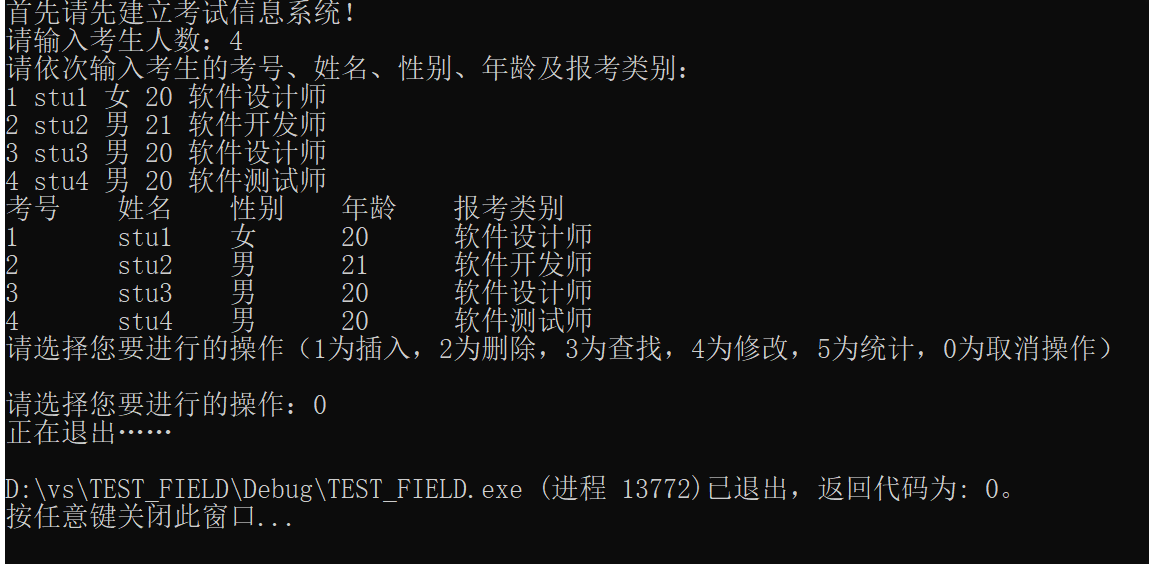
if(wrong==false)

cout<<"请重新确认操作!"<<endl;

}

return 0;}

### 3.6.3 总体系统截屏示例



# 4 测试

## 4.1 功能测试

**见各功能截屏示例**

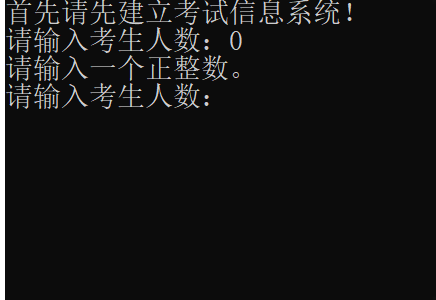
## 4.2 边界测试

### 4.2.1 初始化无输入数据

**测试用例：**初始无输入数据

**预期结果：**给出错误提示，程序运行正常不崩溃。

**实验结果：**

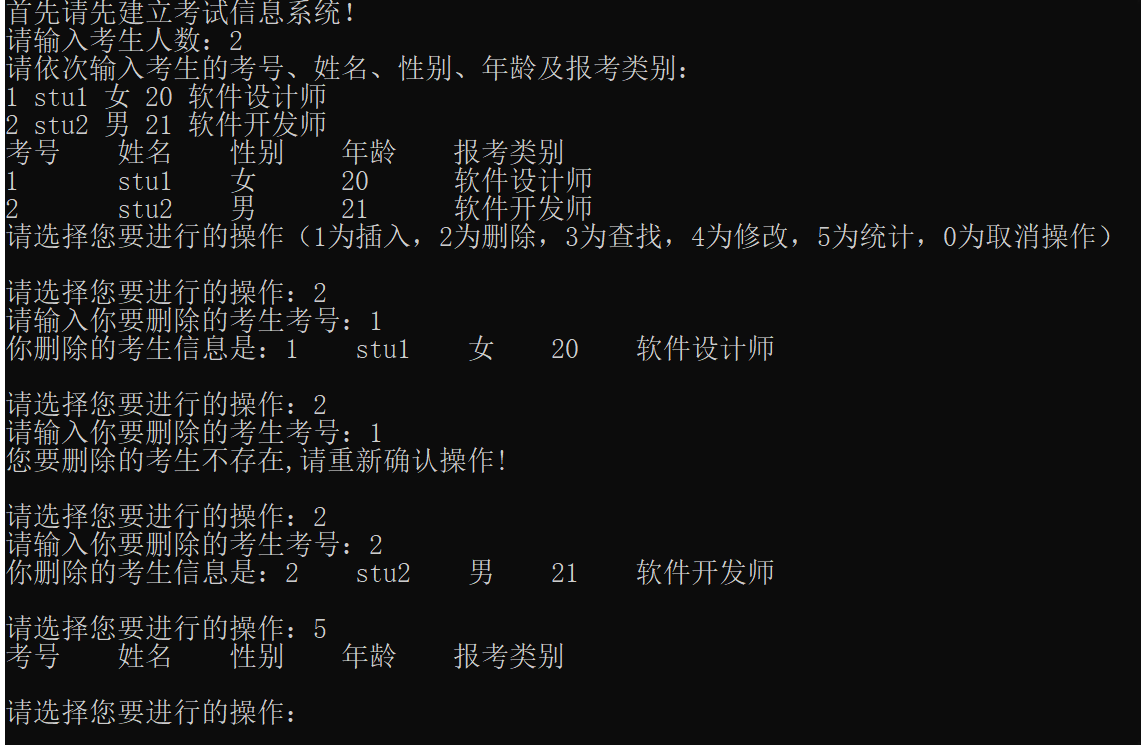


### 4.2.2 删除头结点

**测试用例：**删除头结点

**预期结果：**程序正常运行，不崩溃。

**实验结果：**

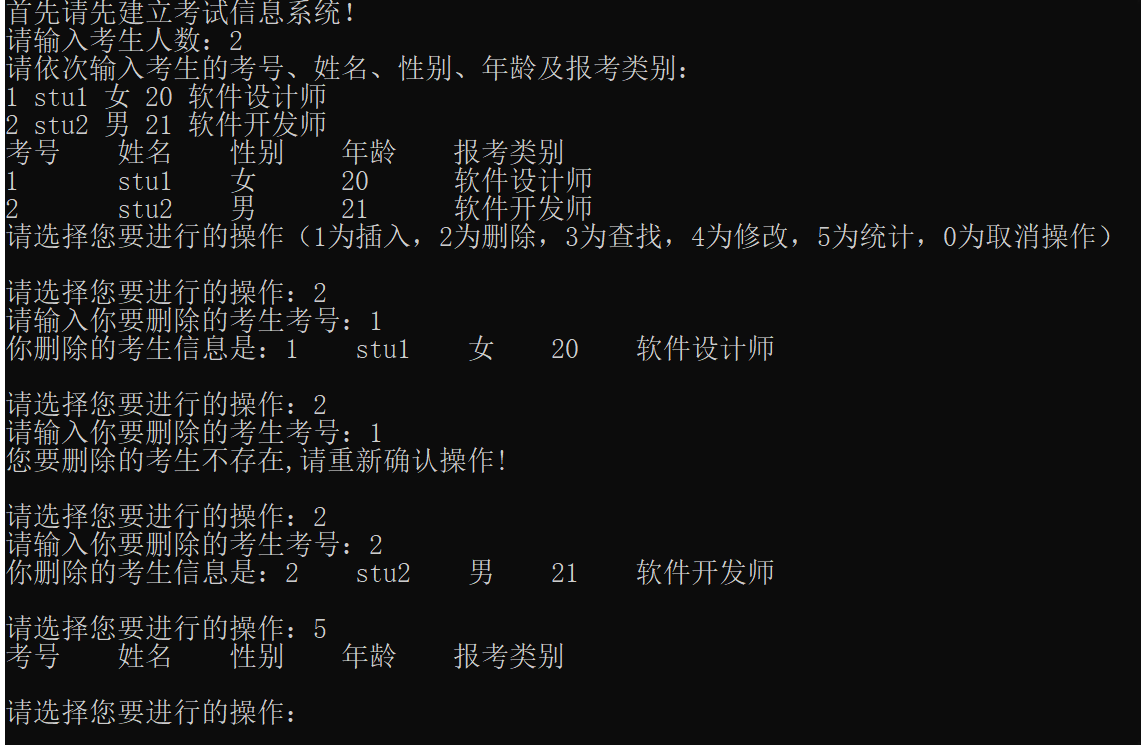


### 4.2.3 删除后链表为空

**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：**



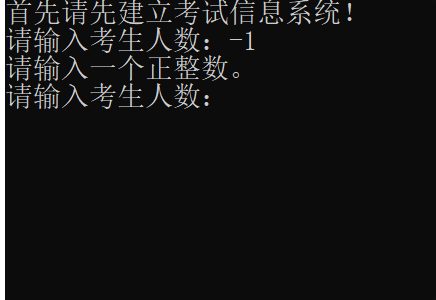
## 4.3 出错测试

### 4.3.1 考生人数错误

**测试用例：**输入考生人数为负数

**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：**

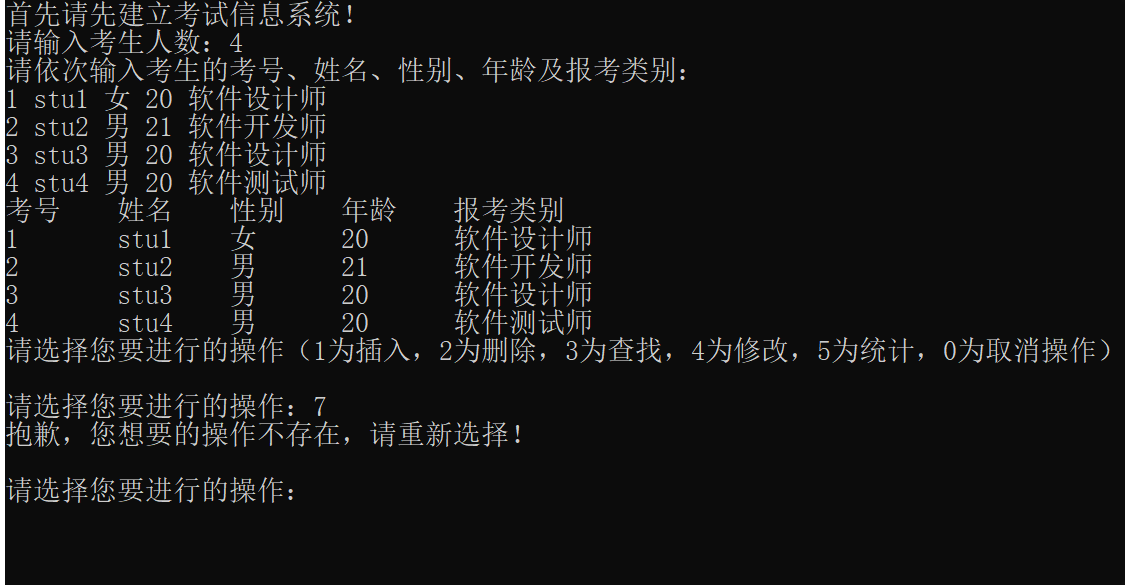
****

### 4.3.2 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：**

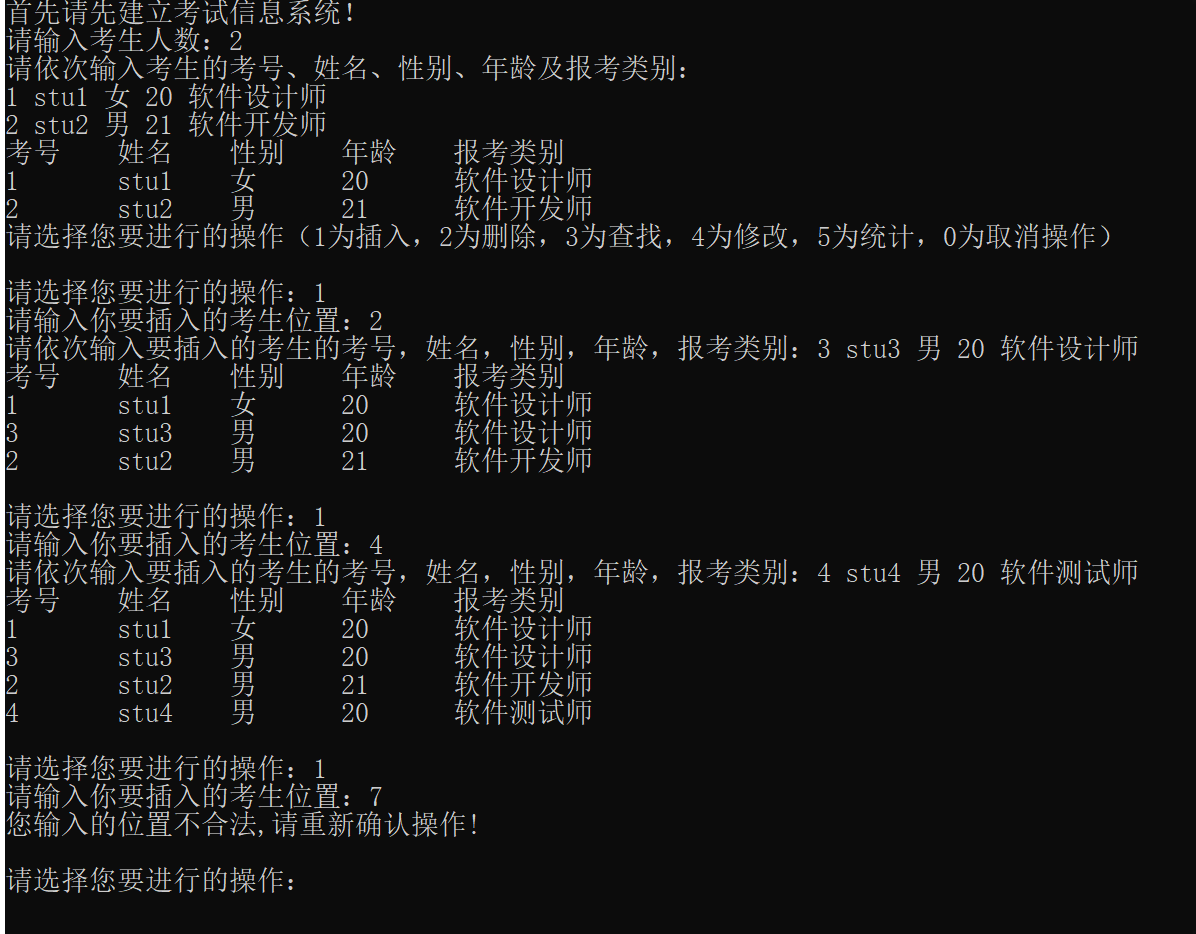
****

### 4.3.3 插入位置不存在

**测试用例：**链表里有四条记录，向链表的第七个位置插入结点

**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：**

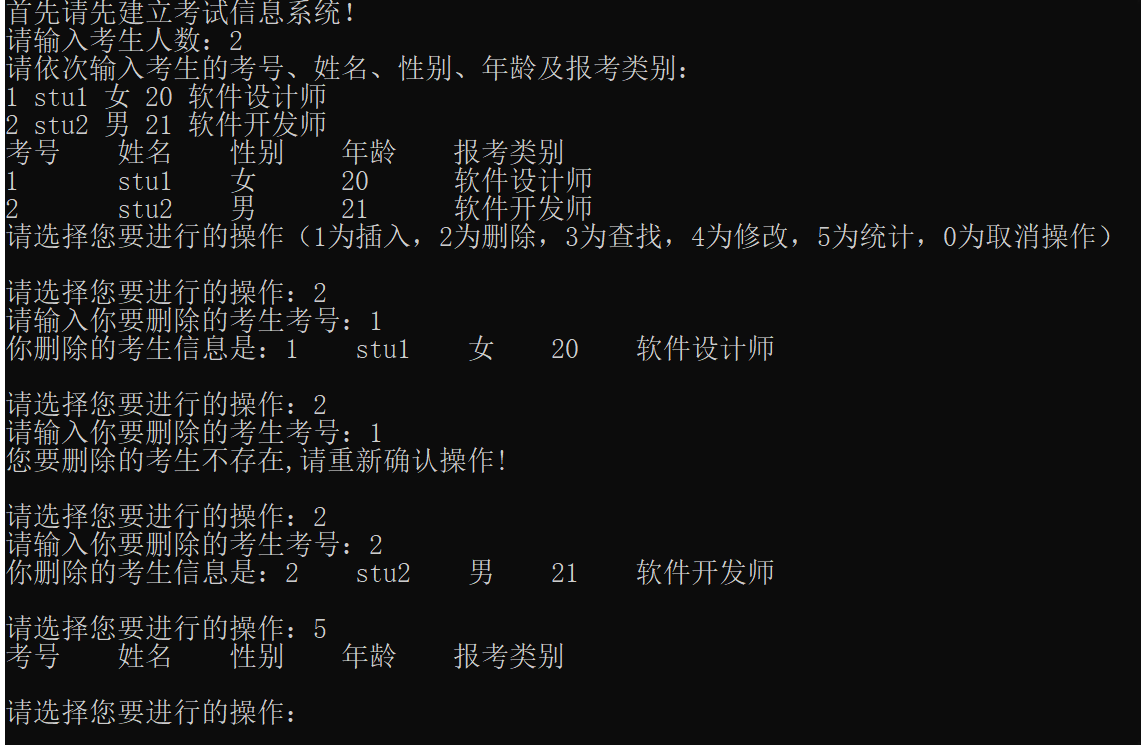


### 4.3.4 删除考号不存在

**测试用例：**要删除的考号不存在

**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：**

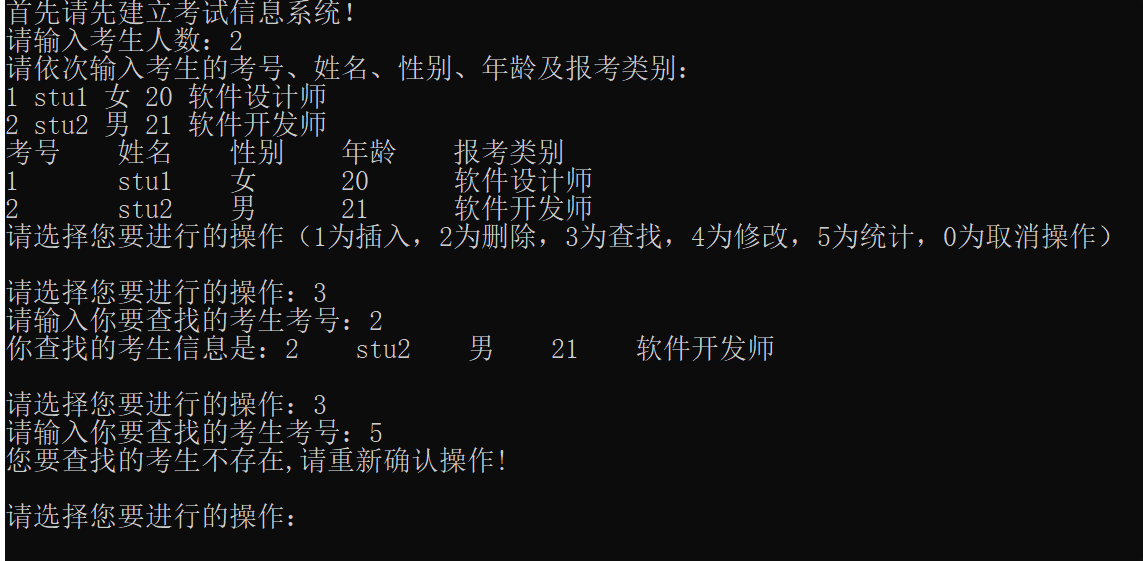


### 4.3.5 查找考号不存在

**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：**

