项目说明文档

数据结构课程设计

——考试报名系统

作 者 姓 名： 崔鑫宇

学 号： 1853444

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

考试报名工作对于各大高校的报名工作一直是一个比较大的挑战，给教务管理部门增加了很大的工作量。尤其是在当下，学生的数量与日俱增，教务管理部门手动对考试报名系统进行各项操作也就变得愈发困难，而犯错误的概率也有所增长。因此开发一款考试报名系统，来代替人工来完成繁杂而又重复的工作便显得尤为重要。

## 1.2 功能分析

作为一个最简易的考试报名系统，首先应该可以提取用户输入的考生信息。其次，还应该可以做到输出考生信息以及查询考生信息。另外还应该具备插入考生信息与删除考生信息的功能以及修改考生信息的功能。此外，还应该可以打印出考生信息的功能，以供检查者检查信息的准确性。

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、退出的功能。

# 2 设计

## 2.1 数据结构设计

综上所述，由于我们需要对每个模块进行大量的增加、修改以及删除等操作。链表这一数据结构就完美的满足了我们的需求。链表相对于其他的所有数据结构在这个方面有着与生俱来的优势，因此最后我决定使用链表数据结构。

## 2.2 类结构设计

在我的设计中使用了一个链表类（LinkList），由于一般的链表一般包括链表类（LinkList）和链表节点类（LinkNode），所以我在本系统的链表类（LinkList）中设置了结构体（Candidate）用来模拟一般链表类中的链表节点类（LinkNode）。

## 2.3 成员与操作设计

**链表结点结构体（Candidate）**

string id;//考号

string name;//考生姓名

string gender;//考生性别

string age;//考生年龄

string major;//考生类型

Candidate\* next;//下一个节点

//初始化

Candidate(Candidate\* ptr = NULL) { next = ptr; }

Candidate(string i, string n, string g, string a, string t, Candidate\* ptr = NULL){

id = i;

name = n;

gender = g;

age = a;

major = t;

next = ptr;

}

//打印一个结点的信息

void print(){

cout << id << '\t' << name << '\t' << gender << '\t' << age << '\t' << major << endl;

}

**链表类（LinkList）**

public:

LinkList();

~LinkList() = default;

LinkList& operator=(LinkList& l);

bool isEmpty() { return length == 0; }//判断链表是否为为空

//插入元素

void Insert(Candidate& mem, int pos);

void Insert(Candidate& mem) { Insert(mem, length + 1); }

//搜索元素

Candidate\* Search(string id);

//前往目标位置

Candidate\* LocatePos(int pos);

Candidate\* Delete(string id);//删除指定元素

void Statistic();//统计功能

int getLength() { return length; }//取得链表长度

bool Exist(string id);//判断目标考号考生是否存在

private:

int length;//链表长度

## Candidate\* head;//头节点

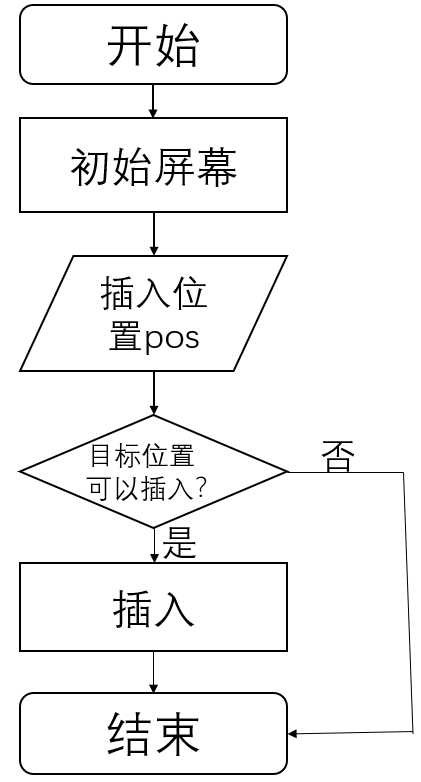
## 2.4 系统设计

系统首先调用BuildList()函数建立一个链表数据结构，然后统计链表。之后根据每次输入的操作码（opt）执行相应操作，每次操作结束后自动对系统进行统计。

# 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

void LinkList::Insert(Candidate& mem, int pos){

Candidate\* p = new Candidate(mem.id, mem.name, mem.gender, mem.age, mem.major);

//插入节点

Candidate\* pre = LocatePos(pos);

p->next = pre->next;

pre->next = p;

length++;

}

void Insert(LinkList& list){

int pos;

cout << "请选择要插入的考生的位置： ";

cin >> pos;

if (pos < 0) {

cerr << "请输入一个正整数!" << endl;

}

else if (pos > list.getLength() + 1 || pos != (int)pos) {

cerr << "无法插入此位置!" << endl;

}

else {

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

string unit;

getchar();

getline(cin, unit);

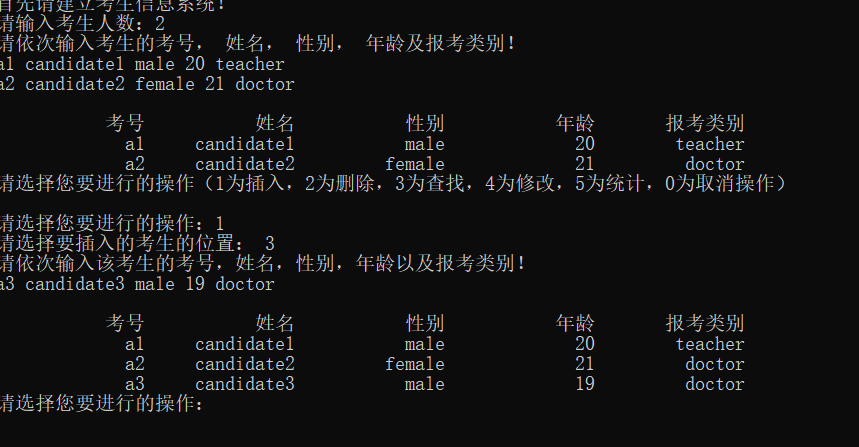
Candidate mem = DistributeUnit(unit);

list.Insert(mem, pos);

}

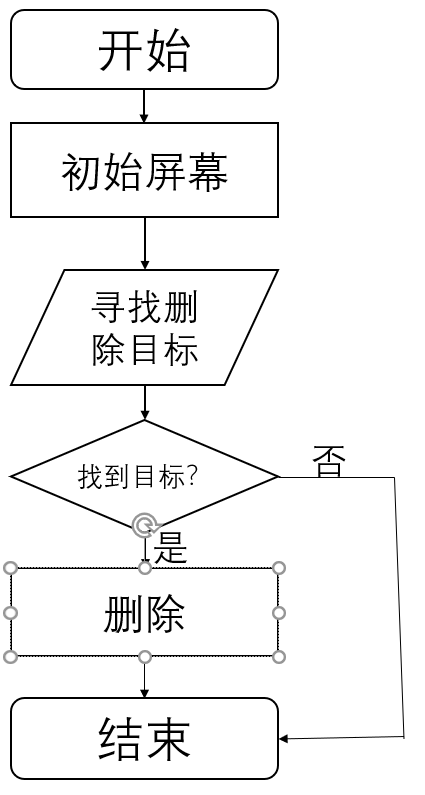
}

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

Candidate\* LinkList::Delete(string id){

Candidate\* p = Search(id);

//删除节点

Candidate\* mem = p->next;

p->next = mem->next;

Candidate\* store = mem;

length--;

return store;

}

void Delete(LinkList& list){

string id;

cout << "请选择要删除的考生的考号： ";

cin >> id;

if (!list.Exist(id)) {

cerr << "不存在考号为" << id << "的考生!" << endl << endl;

}

else {

Candidate\* mem = list.Delete(id);

cout << "你删除的考生的考号信息是： ";

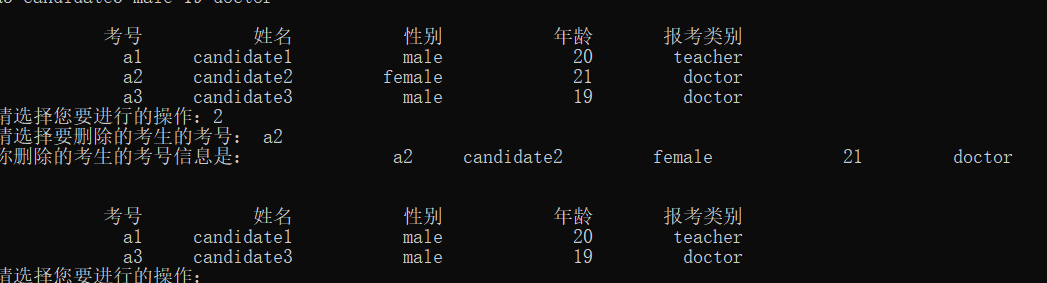
mem->print();

cout << endl;

}

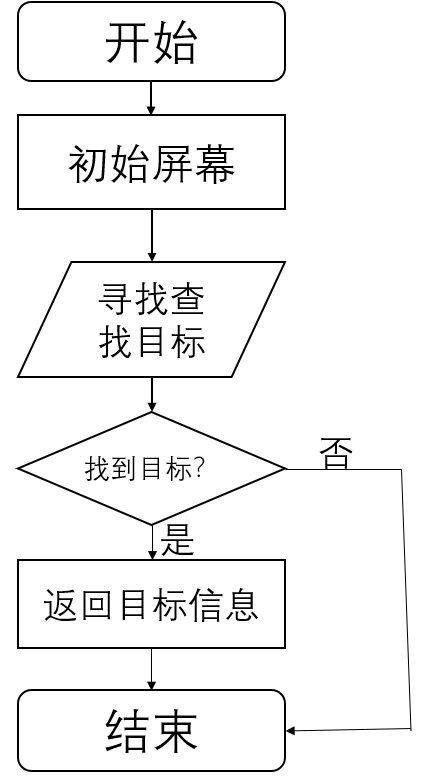
}

### 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

Candidate\* LinkList::Search(string id){

Candidate\* mem = head;

while (mem->next->id != id){

mem = mem->next;

}

if (mem->next == NULL) {

cerr << "没找到考号为" << id << "的考生！";

}

else

return mem;

### }

void Search(LinkList& list){

cout << "请选择要查找的考生的考号： ";

string id;

cin >> id;

if (!list.Exist(id)) {

cout << "该考生不存在!\n";

}

else

{

Candidate\* mem = list.Search(id);

mem = mem->next;

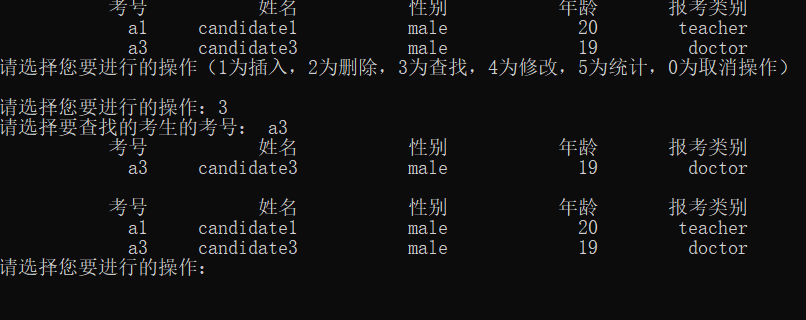
cout << setw(15) << "考号" << setw(15) << "姓名" << setw(15) << "性别" << setw(15) << "年龄" << setw(15) << "报考类别" << endl;

mem->print();

}

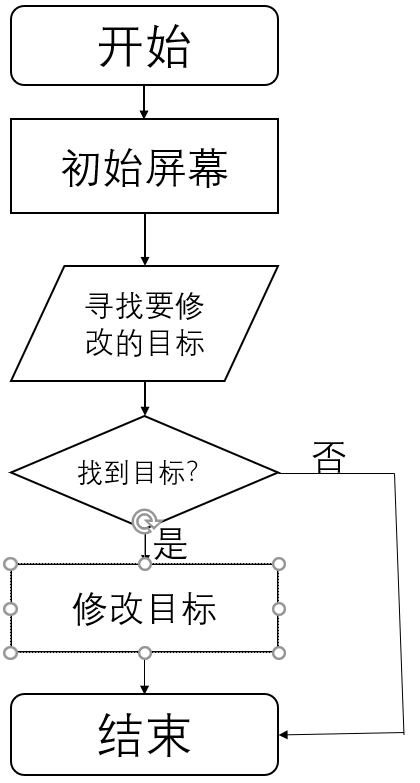
}

### 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 Change(LinkList& list){

cout << "请选择要修改的考生的考号： ";

string id;

cin >> id;

if (!list.Exist(id)) {

cerr << "该考号不存在！" << endl;

}

else {

Candidate\* mem = list.Search(id);

mem = mem->next;

cout << "你要修改的考生的考号信息是： ";

mem->print();

cout << "请依次输入新的考号， 姓名， 性别， 年龄及报考类别！" << endl;

string unit;

getchar();

getline(cin, unit);

Candidate New = DistributeUnit(unit);

mem->id = New.id;

mem->name = New.name;

mem->gender = New.gender;

mem->age = New.age;

mem->major = New.major;

}

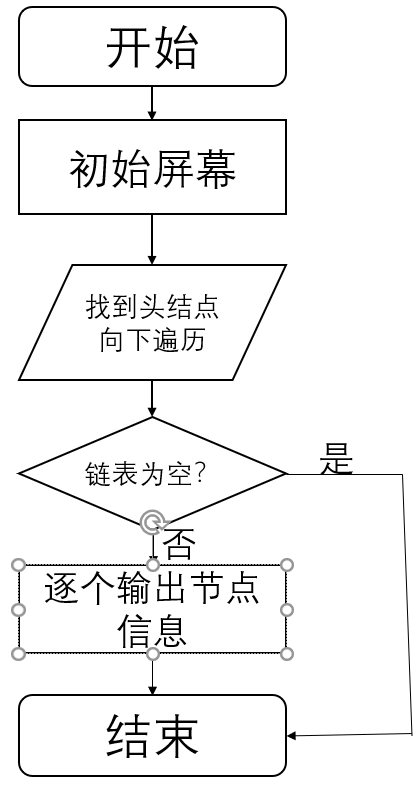
}

### 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 LinkList::Statistic(){

cout << endl;

if (head->next == NULL) {//如果链表为空

cout << "(空)" << endl;

}

else {//否则

cout << setw(15) << "考号" << setw(15) << "姓名" << setw(15) << "性别" << setw(15) << "年龄" << setw(15) << "报考类别" << endl;

Candidate\* mem = head;

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

mem = mem->next;

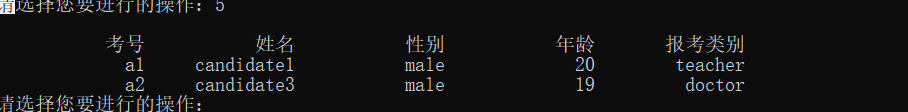
cout << setw(15) << mem->id << setw(15) << mem->name << setw(15) << mem->gender << setw(15) << mem->age << setw(15) << mem->major << endl;

}

}

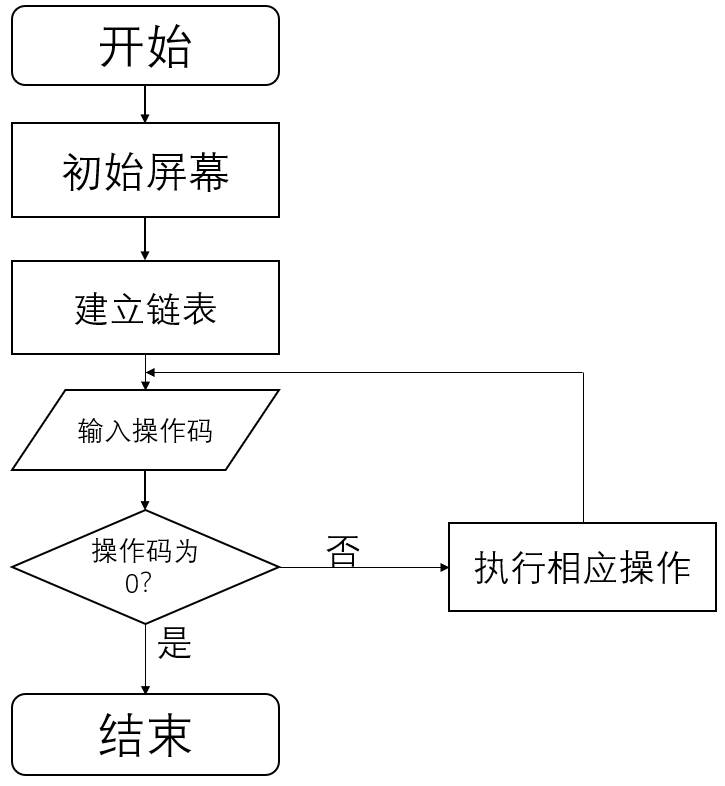
### }

### 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

LinkList list = BuildList();

list.Statistic();

int opt;

bool n = true;

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 << endl;

while (n){

cout << "请选择您要进行的操作：";

cin >> opt;

switch (opt){

case(0):

n = false;

break;

case(1):

Insert(list);

list.Statistic();

break;

case(2):

Delete(list);

list.Statistic();

break;

case(3):

Search(list);

list.Statistic();

break;

case(4):

Change(list);

list.Statistic();

break;

case(5):

list.Statistic();

break;

default:

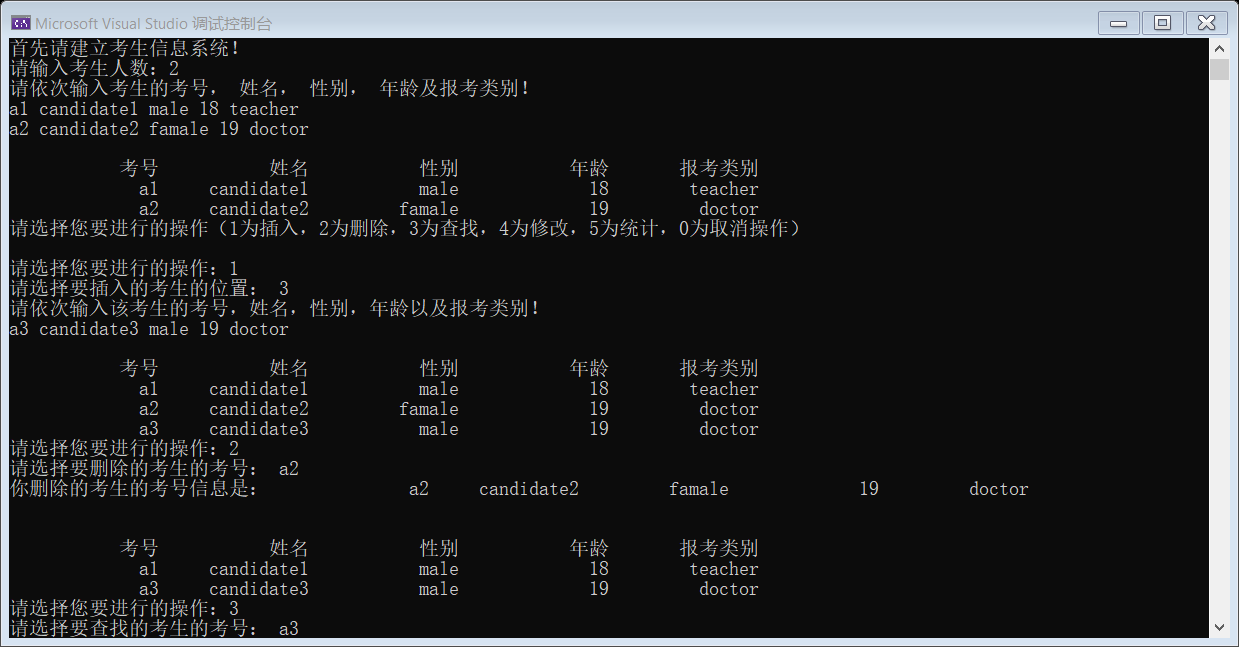
cout << "操作码错误!请重新输入!" << endl << endl;

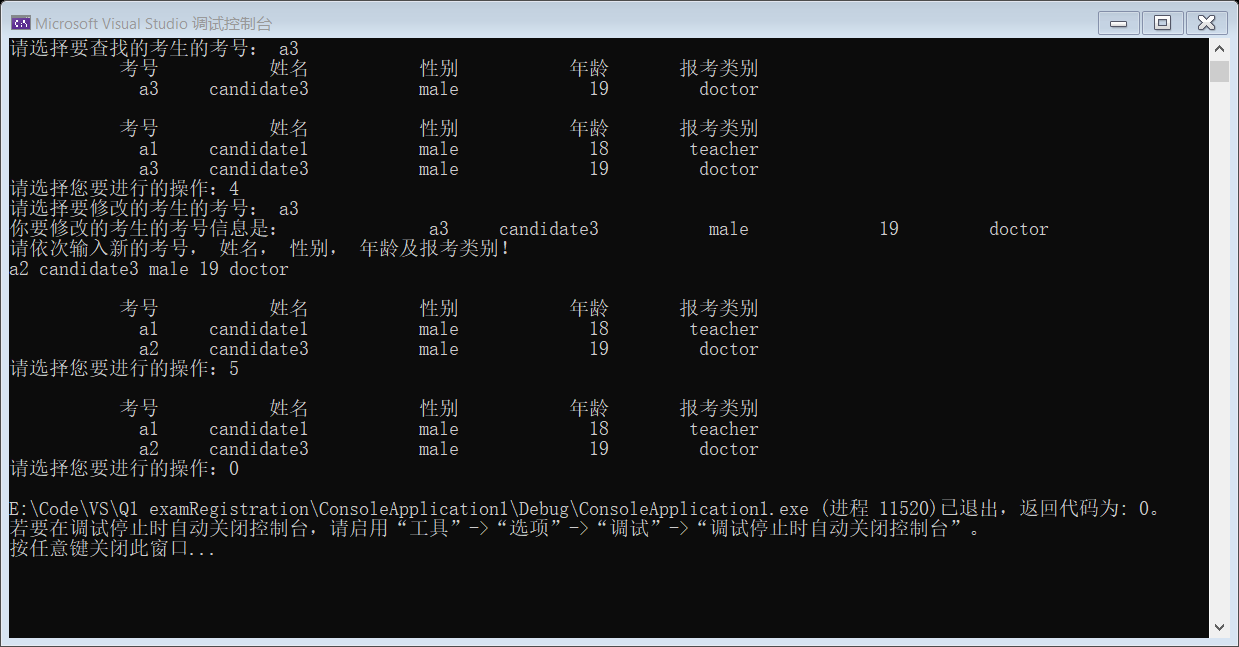
break;

}

}

### 3.6.3 总体系统截屏示例





# 4 测试

## 4.1 常规功能测试

### 4.1.1 插入功能测试

**测试用例**：4 stu4 女 21 软件测试师

**预期结果**：

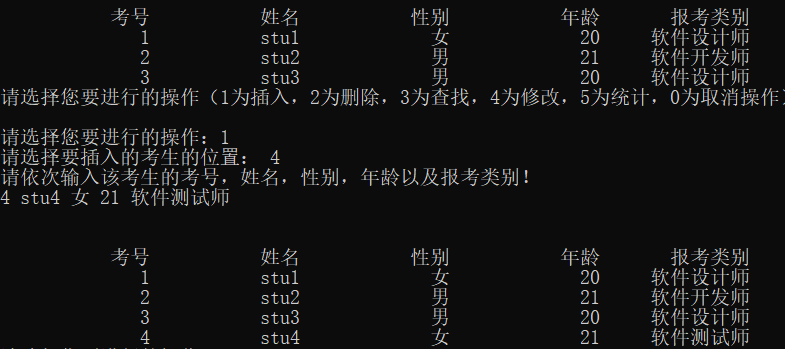
1 stu1女 20 软件设计师

2 stu2男 21 软件开发师

3 stu3男 20 软件设计师

4 stu4女 21 软件测试师

**实验结果**



### 4.1.2 删除功能测试

**测试用例：**删除考号为2的考生

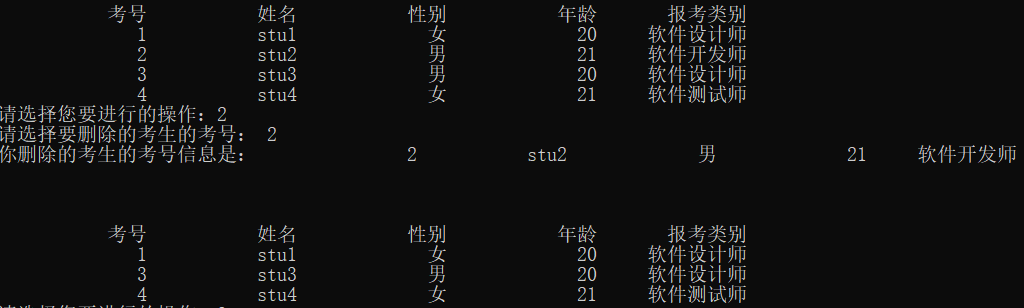
**预期结果：**

1 stu1 女 20 软件设计师

3 stu3 男 20 软件设计师

4 stu4 女 21 软件测试师

**实验结果：**



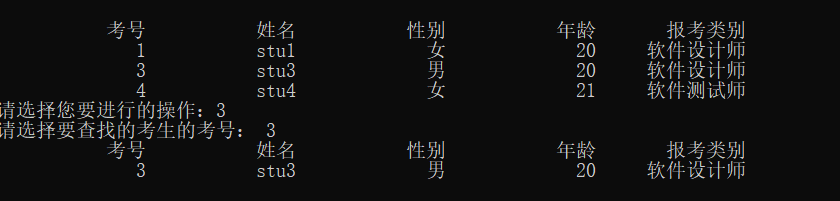
### 4.1.3 查找功能测试

**测试用例：**查找考号为3的考生

**预期结果：**

3 stu3 男 20 软件测试师

**实验结果：**



### 

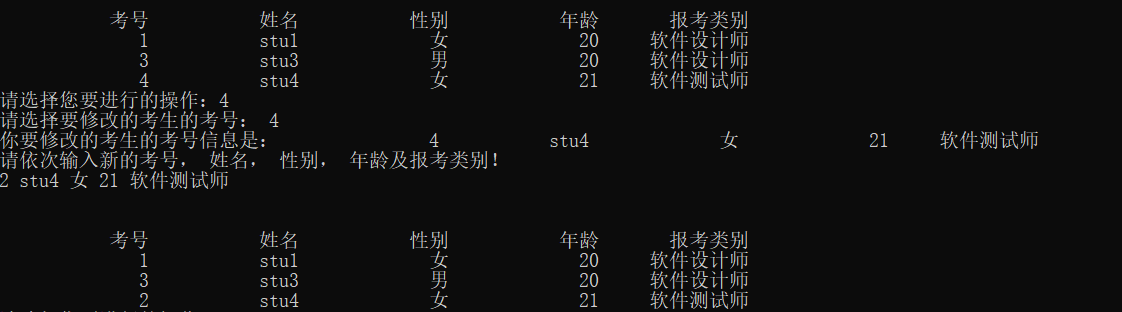
### 4.1.4 修改功能测试

**测试用例：**将考号4修改为考号2，其他项保持原状。

**预期结果：**

2 stu4 女 21 软件测试师

**实验结果：**



### 4.1.5 统计功能测试

**测试用例：**统计当前数据

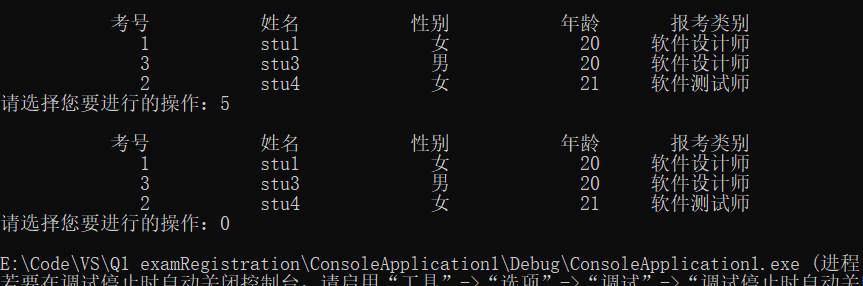
**预期结果：**

1 stu1 女 20 软件设计师

3 stu3 男 20 软件设计师

4 stu2 女 21 软件测试师

**实验结果：**



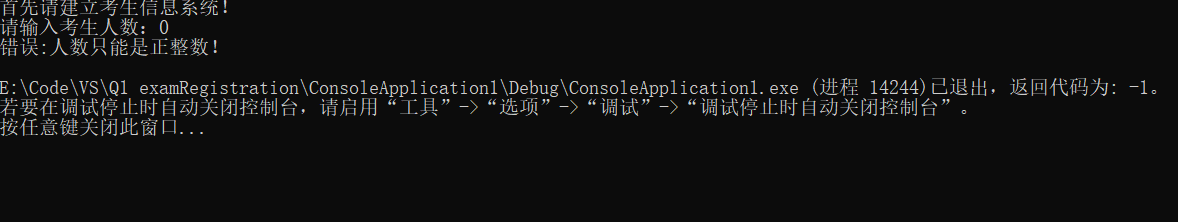
## 4.2 不良输入测试

### 4.2.1 初始化考生人数输入不大于0

**测试用例：**0

**预期结果：**给出错误提示，程序直接关闭。

**实验结果：**

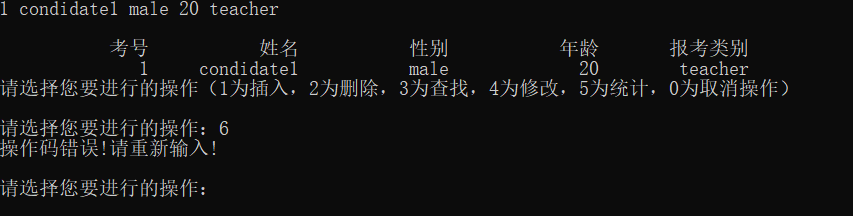


### 4.2.2 操作码错误

**测试用例：**输入操作码：6

**预期结果：**程序给出提示信息，并要求用户重新输入，程序正常运行不崩溃。

**实验结果：**

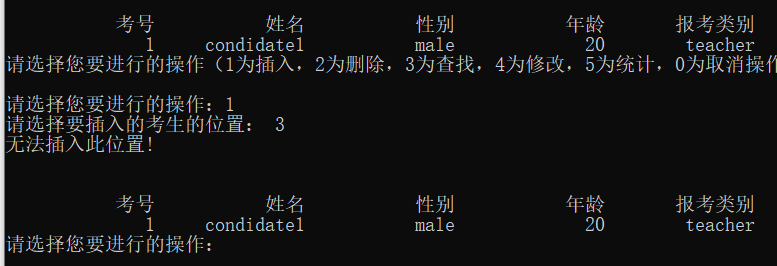
****

### 4.3.3 插入位置不存在

**测试用例：**链表里有一条记录，向链表的第三个位置插入结点

**预期结果：**程序给出错误信息，并跳过本次插入操作，程序正常运行不崩溃。

**实验结果：**

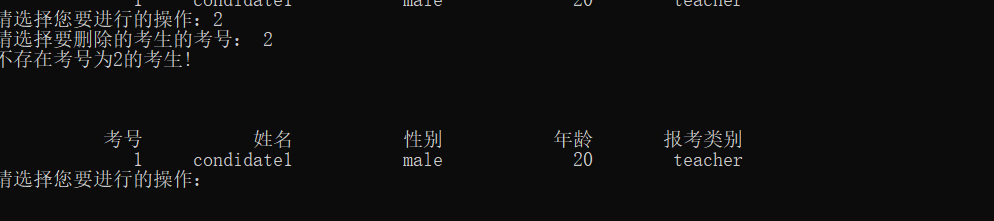


### 4.3.4 删除考号不存在

**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：**

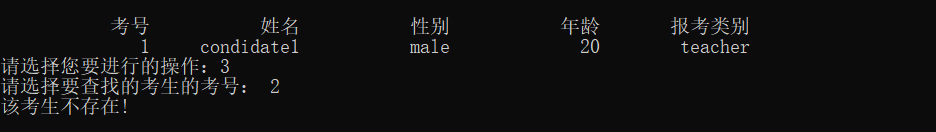
****

### 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：**

