项目说明文档

数据结构课程设计

——考试报名系统

作 者 姓 名： 谢宇翔

学 号： 1951708

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](#_Toc21861)

[1.1 背景分析 1](#_Toc9915)

[1.2 功能分析 1](#_Toc21334)

[2 设计 2](#_Toc24802)

[2.1 数据结构设计 2](#_Toc24214)

[2.2 类结构设计 2](#_Toc4386)

[2.3 成员与操作设计 2](#_Toc23156)

[2.4 系统设计 3](#_Toc28616)

[3 实现 4](#_Toc24415)

[3.1 插入功能的实现 4](#_Toc14229)

[3.1.1 插入功能流程图 4](#_Toc22771)

[3.1.2 插入功能核心代码 5](#_Toc23050)

[3.1.3 插入功能截屏示例 6](#_Toc15772)

[3.2 删除功能的实现 7](#_Toc23312)

[3.2.1 删除功能流程图 7](#_Toc29737)

[3.2.2 删除功能核心代码 7](#_Toc14005)

[3.2.3 删除功能截屏示例 8](#_Toc24313)

[3.3 查找功能的实现 10](#_Toc2596)

[3.3.1 查找功能流程图 10](#_Toc437)

[3.3.2 查找功能核心代码 10](#_Toc16634)

[3.3.3 查找功能截图示例 11](#_Toc7601)

[3.4 修改功能的实现 13](#_Toc21159)

[3.4.1 修改功能流程图 13](#_Toc16996)

[3.4.2 修改功能核心代码 13](#_Toc14209)

[3.4.3 修改功能截屏示例 14](#_Toc31388)

[3.5 统计功能的实现 15](#_Toc15274)

[3.5.1 统计功能流程图 15](#_Toc17494)

[3.5.2 统计功能核心代码 15](#_Toc17920)

[3.5.3 统计功能截屏示例 16](#_Toc12392)

[3.6 总体系统的实现 17](#_Toc7083)

[3.6.1 总体系统流程图 17](#_Toc12380)

[3.6.2 总体系统核心代码 17](#_Toc11531)

[3.6.3 总体系统截屏示例 19](#_Toc3110)

[4 测试 20](#_Toc7948)

[4.1 功能测试 20](#_Toc1701)

[4.1.1 插入功能测试 20](#_Toc7214)

[4.1.2 删除功能测试 21](#_Toc12041)

[4.1.3 查找功能测试 22](#_Toc4804)

[4.1.4 修改功能测试 22](#_Toc3738)

[4.1.5 统计功能测试 22](#_Toc29241)

[4.2 边界测试 23](#_Toc7531)

[4.2.1 初始化无输入数据 23](#_Toc18261)

[4.2.2 删除头结点 23](#_Toc32262)

[4.2.3 删除后链表为空 24](#_Toc22491)

[4.3 出错测试 24](#_Toc16145)

[4.3.1 考生人数错误 24](#_Toc8190)

[4.3.2 操作码错误 25](#_Toc7034)

[4.3.3 插入位置不存在 25](#_Toc26738)

[4.3.4 删除考号不存在 26](#_Toc13488)

[4.3.5 查找考号不存在 26](#_Toc8753)

[4.3.6 修改考号不存在 27](#_Toc24560)

# 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，在对学生信息的录入和修改，保存等方面会存在疏漏。所以本项目是为了给学校校务处有效便捷处理大量学生而设计的，使用计算机处理学生信息的方法会对比人工记录处理有许多方便之处。

## 1.2 功能分析

本项目主要为了实现对学生信息的读取，并包含对学生信息的插入，删除，查找，修改，统计等功能，并且对错误的输入进行提示然后继续进行正确的操作。

# 2 设计

## 2.1 数据结构设计

由于本项目涉及众多学生的信息，并且会有增删改查等众多功能需要实现，所以本项目适合使用动态链表写，就可以很好地适应各种功能的要求

## 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（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struct描述链表结点类（node），这样使得链表结点类（list）可以访问链表结点。

## 2.3 成员与操作设计

**链表结点结构体（node）**

struct node

{

int NO = 0;

string name;

string sex;

int age = 0;

string job;

node\* next = NULL;

};

**链表类（list）**

class list

{

public:

node\* head, \* current;//head为表头空节点

int studentnum = 0;//学生总人数

list()

{

head = new node, current = NULL;//构造函数，为head开辟节点，current等于null

}

void build();//建立链表

void menu();//菜单函数

void insert();//插入函数

void show(node\* p);//展示函数

void tjdelete();//删除函数

node\* find(int want);//搜索函数};

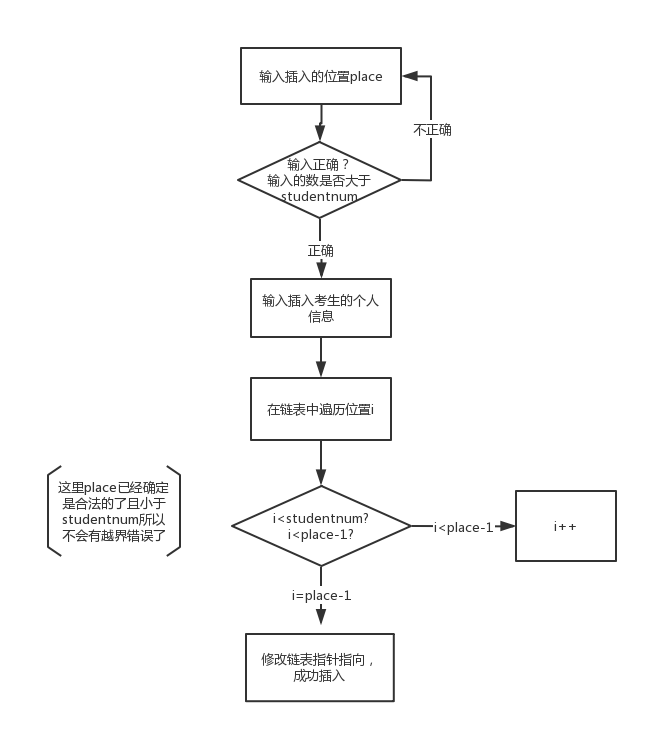
## 2.4 系统设计

首先建立list对象L1，然后调用build函数建立链表，然后调用menu函数输出菜单，选择不同的选项之后分别进入不同的函数进行操作，insert，find，delete，修改名字等操作。结束函数后，循环menu菜单即可。

# 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

int place;

cout << "请输入要插入的位置:";

while (1)

{

cin >> place;

if (!(place > 0 && place <= studentnum + 1))//place从0到最大值减一

{

cout << "输入错误，请重新输入!" << endl;

}

else

break;

}

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

current = head;

for (int i = 0; i < place - 1; i++)

{

current = current->next;

}

node\* p = new node;

cin >> p->NO >> p->name >> p->sex >> p->age >> p->job;

p->next = current->next;

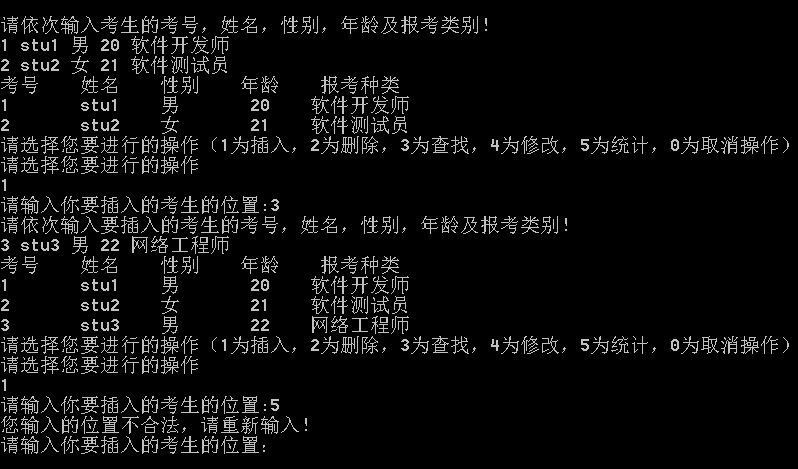
current->next = p;

studentnum++;

show();

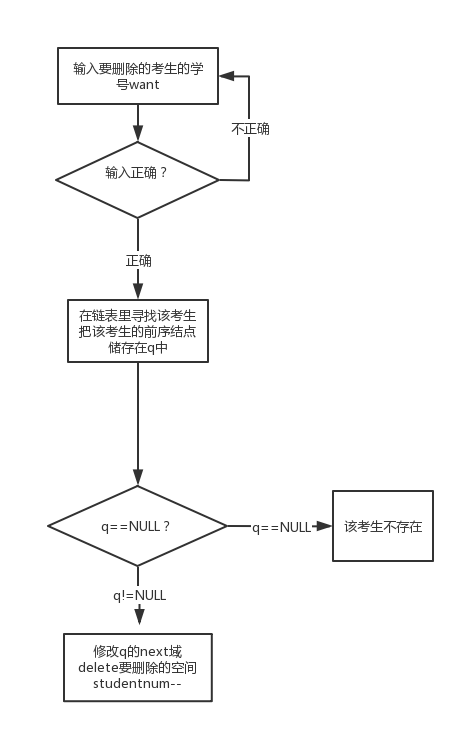
### 3.1.3 插入功能截屏示例





## 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::tjdelete()

{

current = head->next;

if (current == NULL)

{

cout << "已无考生！" << endl;

return;

}

int want;

while (1)

{

cout << "请输入要删除的考生的考号:";

cin >> want;

if (!cin.good())

{

cout << "输入错误！请重新输入" << endl;

}

else

break;

}

node\* q = find(want);

if (q == NULL)

{

cout << "无该考号的考生！" << endl;

return;

}

else

{

cout << "删除的考生信息是:";

cout << setiosflags(ios::left) << setw(8) << q->next->NO << setw(8) << q->next->name << setw(8) << q->next->sex << setw(8) << q->next->age << setw(8) << q->next->job << endl;

node\* p = q->next;

q->next = p->next;

delete p;

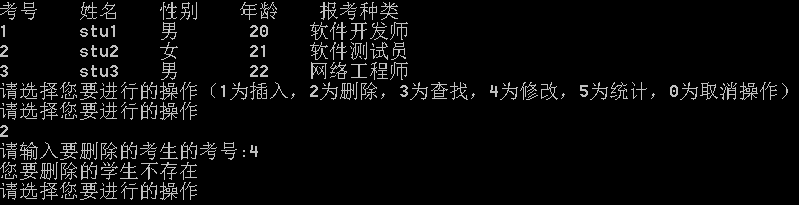
studentnum--;

}

}

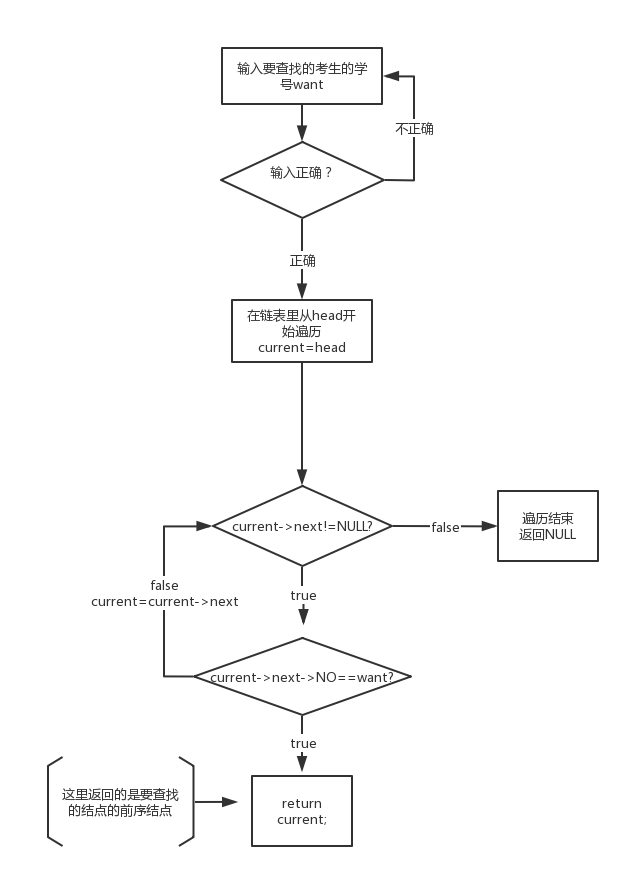
### 3.2.3 删除功能截屏示例





## 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

node\* list::find(int want)

{

current = head;

while (current->next != NULL)

{

if (current->next->NO == want)

{

return current;

}

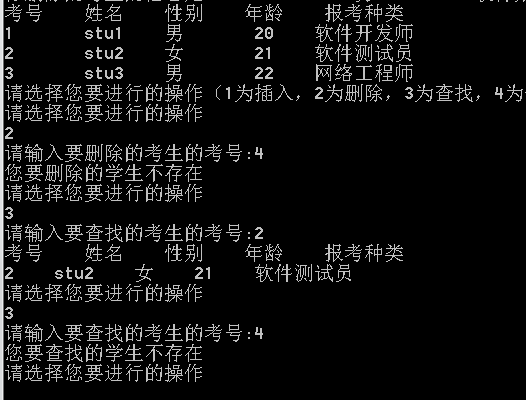
current = current->next;

}

return NULL;}

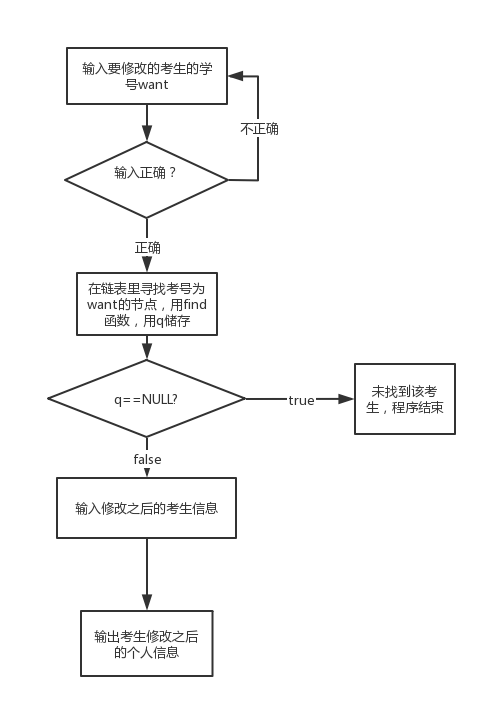
### 3.3.3 查找功能截图示例





## 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

cout << "请输入要修改的考生的当前的考号：";

cin >> want;

q = find(want);

if (q == NULL)

{

cout << "无该考号的考生！" << endl;

break;

}

else

{

cout << "请输入修改后的信息:" << endl;

cin >> q->next->NO >> q->next->name >> q->next->sex >> q->next->age >> q->next->job;

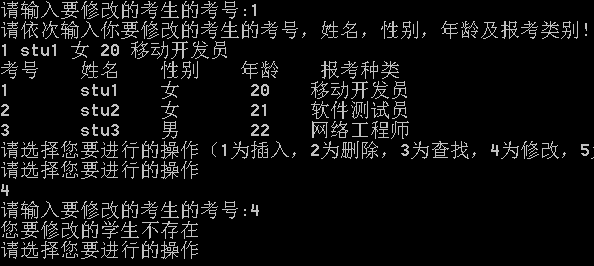
cout << "修改后的考生信息为:" << endl;

cout << setiosflags(ios::left) << setw(8) << q->next->NO << setw(8) << q->next->name << setw(8) << q->next->sex << setw(8) << q->next->age << setw(8) << q->next->job << endl;

break;}

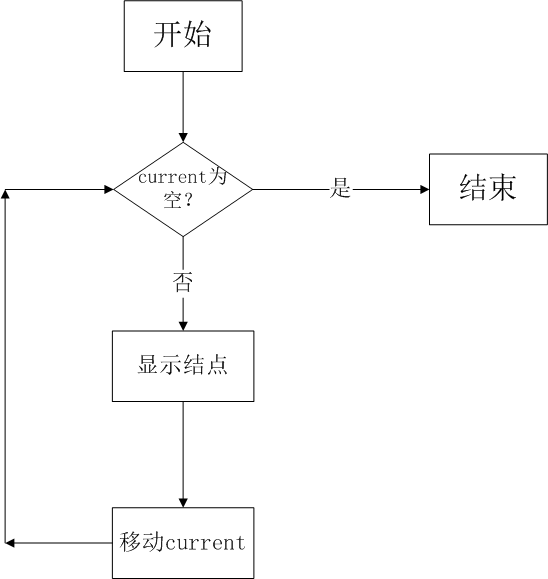
### 3.4.3 修改功能截屏示例





## 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 list::show(node\* p = NULL)

{

cout << "考号 姓名 性别 年龄 报考类别" << endl;

if (p == NULL)

{

current = head->next;

while (current != NULL)

{

cout << setiosflags(ios::left) << setw(8) << current->NO << setw(8) << current->name << setw(8) << current->sex << setw(8) << current->age << setw(8) << current->job << endl;

current = current->next;

}

}

else

{

current = head->next;

while (current->NO != p->NO)

{

current = current->next;

}

//cout << current->NO << current->name << current->sex << current->age << current->job << endl;

cout << setiosflags(ios::left) << setw(8) << current->NO << setw(8) << current->name << setw(8) << current->sex << setw(8) << current->age << setw(8) << current->job << endl;

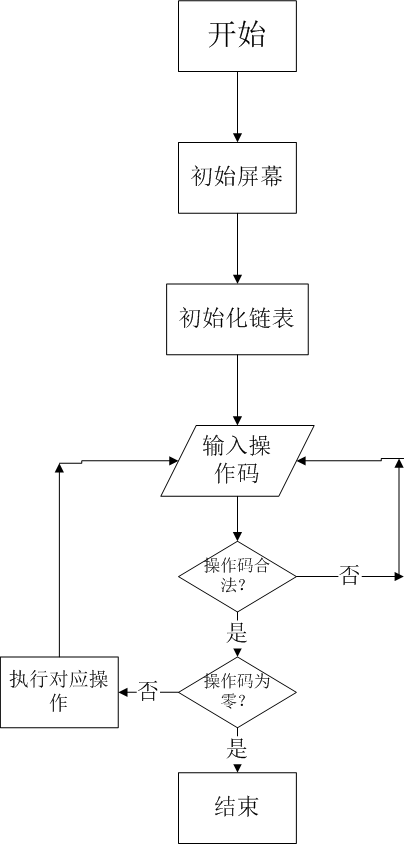
}}

### 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

while (1)

{

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;

cin >> choice;

if (!(choice >= 0 && choice < 6))

{

cout << "输入错误，请重新输入!" << endl;

}

else

switch (choice)

{

case 1:

insert();

break;

case 2:

tjdelete();

break;

case 3:

cout << "请输入要查找考生的考号:";

cin >> want;

q = find(want);

if (q == NULL)

{

cout << "无该考号的考生！" << endl;

break;

}

else

{

cout << "查找的考生信息是:";

cout << setiosflags(ios::left) << setw(8) << q->next->NO << setw(8) << q->next->name << setw(8) << q->next->sex << setw(8) << q->next->age << setw(8) << q->next->job << endl;

break;

}

case 4:

cout << "请输入要修改的考生的当前的考号：";

cin >> want;

q = find(want);

if (q == NULL)

{

cout << "无该考号的考生！" << endl;

break;

}

else

{

cout << "请输入修改后的信息:" << endl;

cin >> q->next->NO >> q->next->name >> q->next->sex >> q->next->age >> q->next->job;

cout << "修改后的考生信息为:" << endl;

cout << setiosflags(ios::left) << setw(8) << q->next->NO << setw(8) << q->next->name << setw(8) << q->next->sex << setw(8) << q->next->age << setw(8) << q->next->job << endl;

break;

}

case 5:

show();

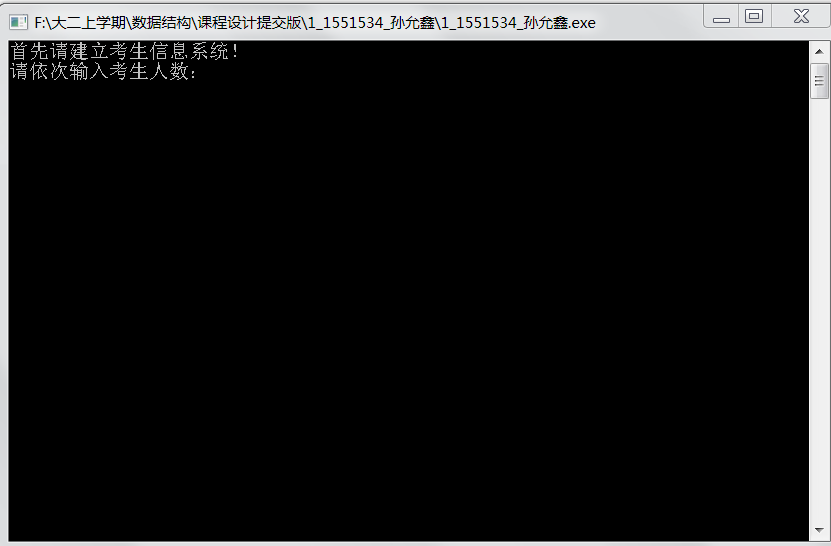
case 0:

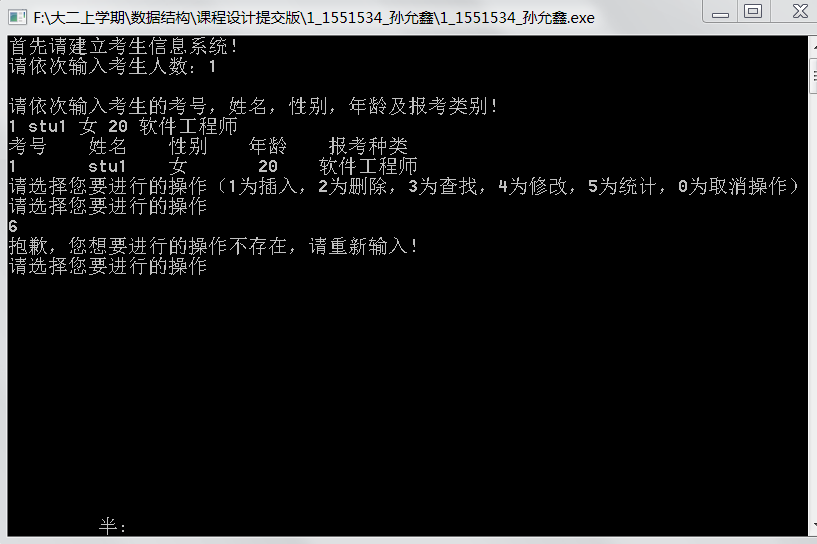
return;

}}

### 

### 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：**删除考号为4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

3 stu3 男 22 网络工程师

**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

3 stu3 男 22 网络工程师

**实验结果：**



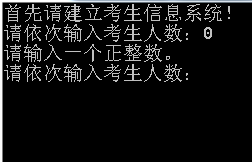
## 4.2 边界测试

### 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：**

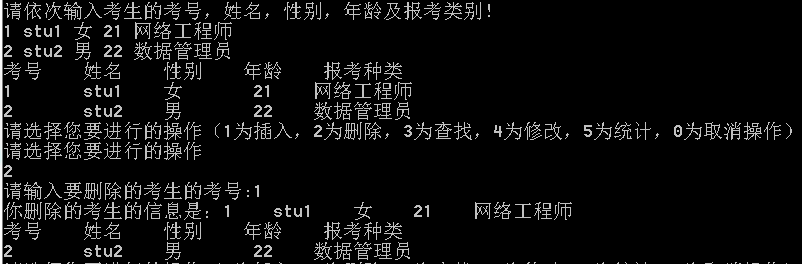


### 4.2.2 删除头结点

**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：**

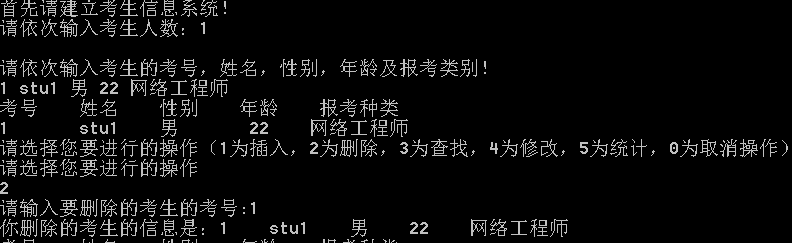


### 4.2.3 删除后链表为空

**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：**



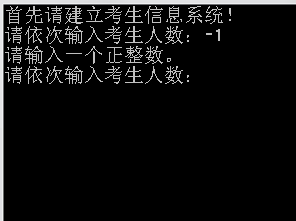
## 4.3 出错测试

### 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：**

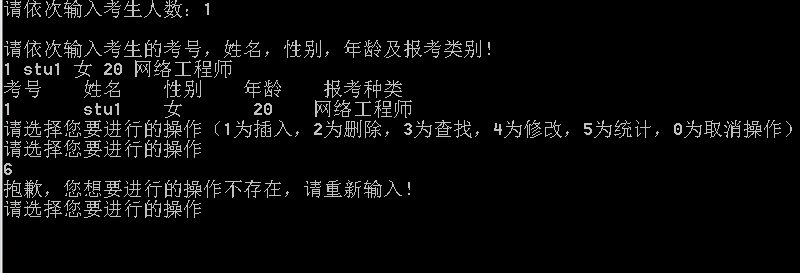
****

### 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：**

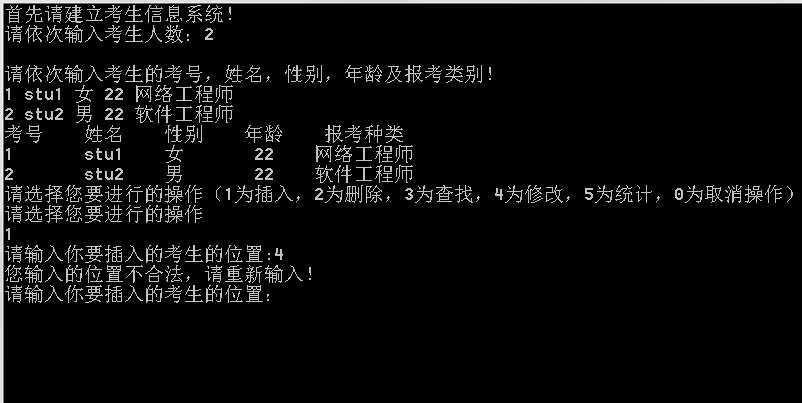
****

### 4.3.3 插入位置不存在

**测试用例：**链表里有两条记录，向链表的第四个位置插入结点

**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：**

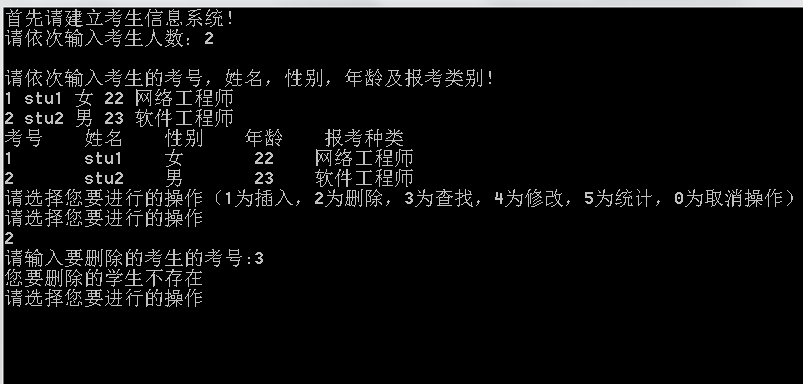


### 4.3.4 删除考号不存在

**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：**

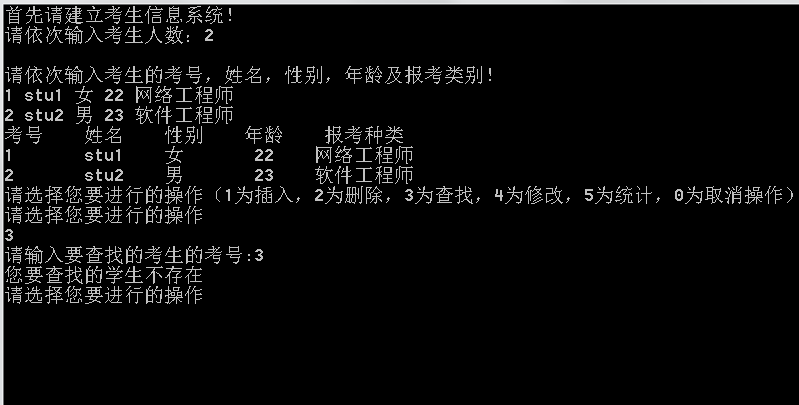
****

### 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：**

