考试报名系统

shu'ju'jie'gou

目录

[1、分析： 3](#_Toc57834419)

[1.2项目简介与需求分析： 3](#_Toc57834420)

[1.2、功能分析： 3](#_Toc57834421)

[2、设计： 3](#_Toc57834422)

[2.1、数据结构的选取 4](#_Toc57834423)

[2.2、ADT接口： 4](#_Toc57834424)

[2.2.1、<数据对象>： 4](#_Toc57834425)

[2.2.2、<数据关系>： 4](#_Toc57834426)

[2.2.3、<操作接口>: 5](#_Toc57834427)

[2.3、系统设计： 5](#_Toc57834428)

[3、实现： 6](#_Toc57834429)

[3.1、建立双向链表： 6](#_Toc57834430)

[3.1.1、建立双向链表的流程图： 6](#_Toc57834431)

[3.1.2、建立链表的核心代码： 7](#_Toc57834432)

[3.1.3、建立部分截屏示例： 9](#_Toc57834433)

[3.2、插入数据： 9](#_Toc57834434)

[3.2.1、插入数据的流程图： 9](#_Toc57834435)

[3.2.2、插入部分核心代码： 11](#_Toc57834436)

[3.2.3、插入部分截图展示： 11](#_Toc57834437)

[3.2、删除数据： 14](#_Toc57834438)

[3.2.1、删除数据的流程图： 15](#_Toc57834439)

[3.2.2、删除的核心代码部分： 16](#_Toc57834440)

[3.2.3、删除部分截图展示： 16](#_Toc57834441)

[3.3、查找数据： 19](#_Toc57834442)

[3.3.1、查找数据的流程图： 19](#_Toc57834443)

[3.3.2、查找数据的代码示例： 20](#_Toc57834444)

[3.3.3、查找部分截图展示： 21](#_Toc57834445)

[3.4、修改数据： 22](#_Toc57834446)

[3.4.1、修改数据的流程图： 22](#_Toc57834447)

[3.4.2、修改数据的核心代码展示： 23](#_Toc57834448)

[3.4.3、修改部分截图展示： 25](#_Toc57834449)

[3.5、统计数据： 27](#_Toc57834450)

[3.5.1、统计数据的流程图： 27](#_Toc57834451)

[3.5.2、统计部分的代码： 27](#_Toc57834452)

[3.5.3、统计部分截图展示： 28](#_Toc57834453)

[3.6、总体系统的实现： 29](#_Toc57834454)

[3.6.1、总体部分的流程 29](#_Toc57834455)

[3.6.2、核心部分代码： 29](#_Toc57834456)

[3.6.3、主界面展示： 31](#_Toc57834457)

[4、测试 32](#_Toc57834458)

[4.1、基本测试： 32](#_Toc57834459)

[4.1.1、插入数据测试： 33](#_Toc57834460)

[4.1.2、修改数据测试： 35](#_Toc57834461)

[4.1.3、查询数据测试： 36](#_Toc57834462)

[4.1.4、删除数据测试： 37](#_Toc57834463)

[4.1.5、统计功能测试： 39](#_Toc57834464)

[4.2、边界情况测试： 40](#_Toc57834465)

[4.2.1、删除后链表为空： 40](#_Toc57834466)

# 1、分析：

## 1.2项目简介与需求分析：

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。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。

根据要求，本项目在数据结构上采用链表，通过链表操作实现建、增、删、查、改这些功能。其中考生信息包括准考证号，姓名，性别，年龄和报考类别这些信息。本程序使用C++中的类来实现。

## 1.2、功能分析：

作为考试报名系统，首先要能记录同学们的考试成绩；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。最后，考试报名系统软件还应该确保软件可以正常关闭。

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、退出的功能。

# 2、设计：

## 2.1、数据结构的选取

本项目采用双向链表。根据需求，需要进行大量的增加、删除、修改操作，而链表在增加、删除和修改时的操作均十分便捷；而相比于单向链表，双向链表的相关操作会更加便捷。

## 2.2、ADT接口：

### 2.2.1、<数据对象>：

struct student:

|  |
| --- |
| num |
| name |
| age |
| sex |
| subject |
| last |
| next |

### 2.2.2、<数据关系>：

双向链表中的一个节点：

|  |  |  |
| --- | --- | --- |
| last | student | next |

### 2.2.3、<操作接口>:

info\_system():构造考试报名系统；

void insert()：插入数据；

void del()：删除数据；

void revise()：修改数据

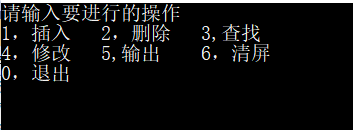
void display()：展示数据；

int getSize()：获取系统中学生总数

## 2.3、系统设计：

通过操作码来获取用户操作。

截图：



# 3、实现：

## 3.1、建立双向链表：

### 3.1.1、建立双向链表的流程图：

开始

输入系统人数

合法否？

头指针指向尾指针

尾指针指向头指针

las指针指向头指针

输入学生信息

在las后一个位置插入信息

cur指向后一个位置

结束

是

否

### 3.1.2、建立链表的核心代码：

info\_system()

{

//建立头尾指针

head = new ElemType(1);

tail = new ElemType(1);

cout << "请输入考生人数： ";

//考生人数输入与检查

while ((\_size = getNum()) == -1 || \_size < 1)

{

cout << "输入有误，请重新输入！ ";

}

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

ElemType\* las = head;

head->next = tail;

head->last = NULL;

tail->last = head;

tail->next = NULL;

//输入考生数据

for (int i = 0; i < \_size; ++i)

{

ElemType\* cur = new ElemType;

//在las之后插入数据

insertAfter(las, cur);

las = cur;

}

其中，在结构体student（ElemType）中使用构造函数：

//构造函数，输入学生信息

student()

{

num = new char[100];

name = new char[100];

sex = new char[5];

subject = new char[100];

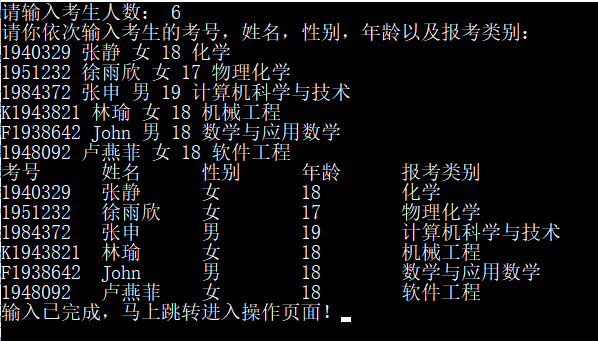
cin >> num >> name >> sex >> age >> subject;

}

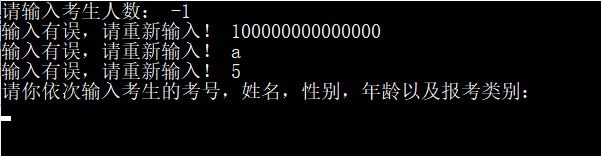
//不做任何处理的构造函数

student(int num){}

### 3.1.3、建立部分截屏示例：



输入错误示例：



## 3.2、插入数据：

3.2.1、插入数据的流程图：

开始

读取位置

合法否？

按照位置寻找节点

用p指向插入位置之前的节点

输入考生信息

newE->last = p

newE->next = p->next

p->next->last = newE,

p->next = newE

结束

是

否

### 3.2.2、插入部分核心代码：

void insert()

{

//输入位置

cout << "请输入你要插入考生的位置：";

int pos;

if ((pos = getNum()) == -1 || pos<1 || pos>\_size + 1)

{

cout << "输入位置错误" << endl;

return;

}

//寻找要插入位置的前一个位置，用指针p指向它

for(ElemType \*p=head;p!=tail;p=p->next)

{

if (!(--pos))

{

cout << "请输入你要插入考生的学号、名字、性别、年龄以及报考专业： " << endl;

ElemType\* newE = new ElemType;

newE->last = p, newE->next = p->next;

p->next->last = newE, p->next = newE;

cout << "插入操作已经完成！" << endl;

++\_size;

return;

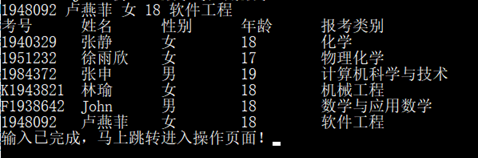
}

}

}

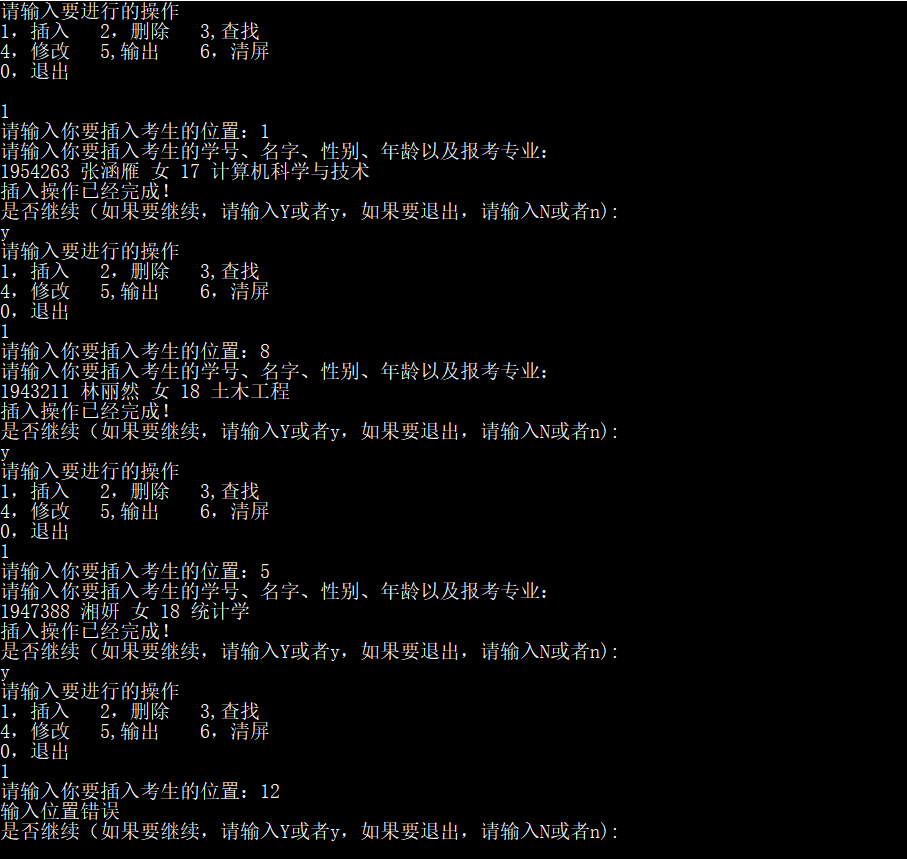
### 3.2.3、插入部分截图展示：

#### <1>、数据初始化：

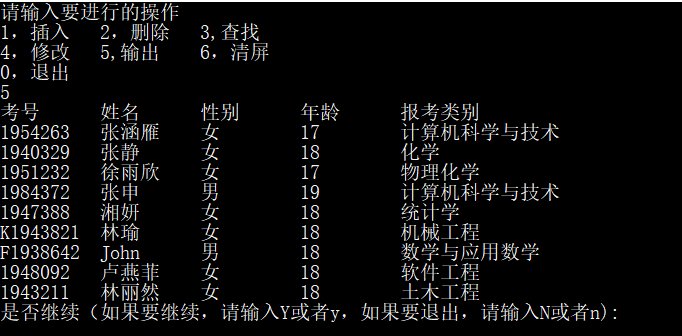


#### <2>、插入数据截图

分别在开头、中间部分、结尾部分以及超出数据范围部分插入数据，考虑到可能会连续进行大量操作，为防止控制台的混乱，在每一次插入操作结束后不予输出。如果用户想要输出数据，可以输入5。



#### <3>、插入结果展示：



可以看出，张涵燕、林丽然和湘妍的数据都已经插入正确位置。

## 3.2、删除数据：

### 3.2.1、删除数据的流程图：

开始

输入考号

是否找到？

p指向尾节点？

p指向下一个节点

输出要删除考生的数据

p->last->next = p->next

p->next->last = p->last

delete p;

输出错误提示

结束

是

否

是

否

### 3.2.2、删除的核心代码部分：

void del()

{

//输入删除考生的考号

cout << "请输入你要删除的考生的考号";

char\* num = new char[100];

cin >> num;

bool find = 0;

//寻找要删除的考生信息

Search

{

if (!strcmp(num, p->num))

{

find = 1;

cout << "你要删除的考生的信息是：" << endl;

//展示该信息：

display(p);

//删除内容

p->last->next = p->next;

p->next->last = p->last;

delete p;

--\_size;

break;

}

}

if (!find)

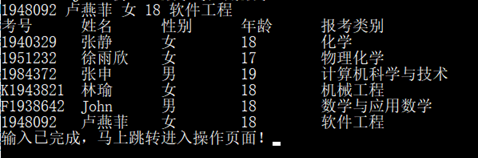
cout << "没找到该考生的信息！" << endl;

delete[]num;

}

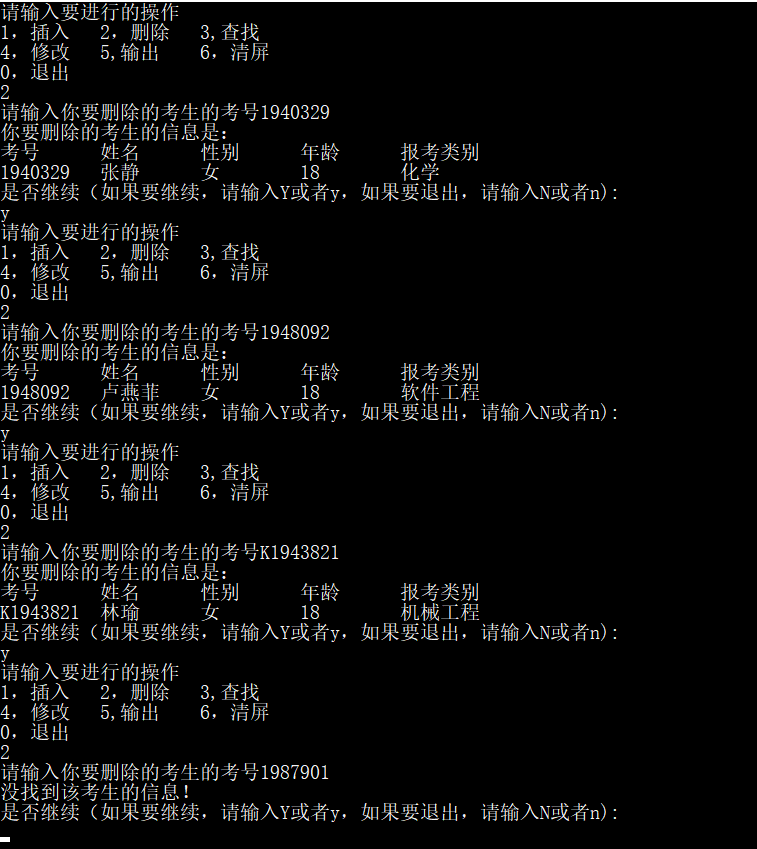
### 3.2.3、删除部分截图展示：

#### <1>、数据初始化：

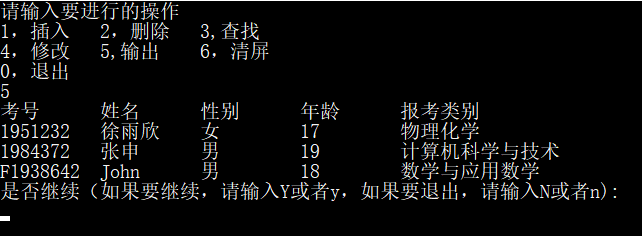


#### <2>、删除数据截图

分别删除开头，中间、结尾和不存在的数据。



#### <3>、展示删除后的数据：



可以看出，张静，卢燕菲和林榆的数据已经被删除。

## 3.3、查找数据：

### 3.3.1、查找数据的流程图：

开始

输入考号

是否找到？

p指向尾节点？

p指向下一个节点

展示该考生的信息

输出错误提示

结束

是

否

是

否

### 3.3.2、查找数据的代码示例：

//根据考生的考号来查找考生

void search()

{

char\* num = new char[100];

cout << "请输入考生的考号：";

cin >> num;

//遍历所有信息

for (ElemType\* p = head->next; p != tail; p = p->next)

{

//如果找到该考号，输出该考生的信息

if (!strcmp(num, p->num))

{

display(p);

delete[]num;

return;

}

};

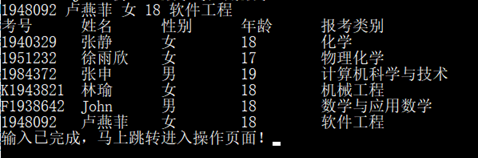
cout << "Not Found!" << endl;

delete[]num;

}

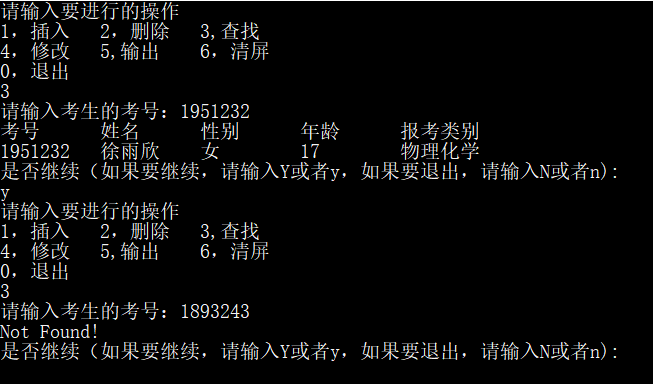
### 3.3.3、查找部分截图展示：

#### <1>、数据初始化：



#### <2>、查找数据展示：

查找1951232和1893243



## 3.4、修改数据：

### 3.4.1、修改数据的流程图：

开始

输入考号

是否找到？

p指向尾节点？

p指向下一个节点

newE->last = p->last newE->next = p->next

输出错误提示

结束

输入考生信息

将newE中的数据深拷贝到p中

删除newE

是

否

是

否

### 3.4.2、修改数据的核心代码展示：

//修改数据

void revise()

{

//输入考号与正误判断

cout << "请输入你要修改的考生的考号：";

char\* num = new char[100];

cin >> num;

for (ElemType\* p = head->next; p != tail; p = p->next)

{

//判断考号是否匹配

if (!strcmp(num, p->num))

{

cout << "你要修改的考生的信息是：" << endl;

display(p);

cout << "请输入该考生的考号、姓名、性别、年龄以及报考类别：";

//建立新的节点newE

ElemType\* newE = new ElemType();

newE->last = p->last;

newE->next = p->next;

//调用student中重载的operator=，将newE的值赋值给p

\*p = \*newE;

//删除newE(newE里的值已经被拷贝赋值给p）

delete newE;

return;

}

}

}

结构体student中重载等号部分的代码：

student& operator=(const student& stu)

{

//防止自赋值

if (this == &stu)return \*this;

//分别拷贝数据内容

strcpy(this->num, stu.num);

strcpy(this->name, stu.name);

strcpy(this->sex, stu.sex);

strcpy(this->subject, stu.subject);

age = stu.age;

last = stu.last;

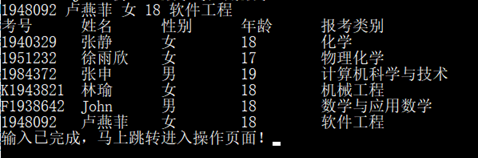
next = stu.next;

return \*this;

}

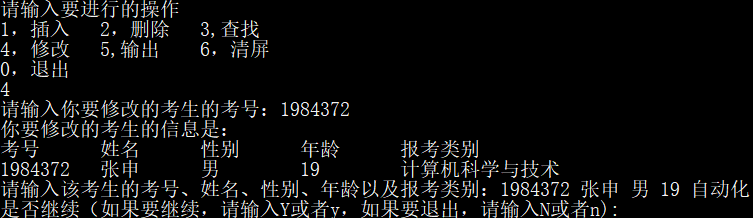
### 3.4.3、修改部分截图展示：

#### <1>、数据初始化：

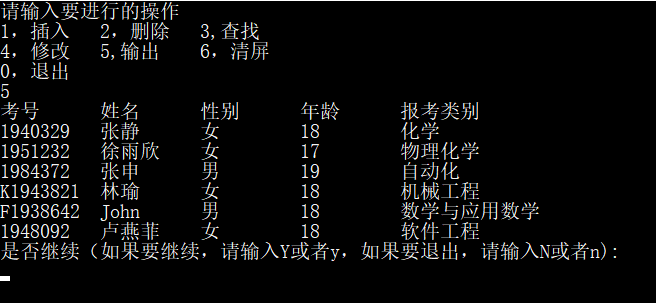


#### <2>、修改数据

将张申的报考类别修改为自动化：



#### <3>、修改后的数据展示：



可见，张申的专业修改成功

## 3.5、统计数据：

### 3.5.1、统计数据的流程图：

开始

p为尾节点？

显示节点

移动p

结束

是

否

### 3.5.2、统计部分的代码：

在public中：  
//展示链表中所有数据

void display()

{

//判断是否是第一次，如果是，需要输出表头

bool fir = 1;

for (ElemType\* p = head->next; p != tail; p = p->next)

{

if (fir)

{

fir = 0;

display(p);

}

else

display(p,1);

}

}

在private中，重载带有默认变量的display函数：

void display(ElemType\* e, bool consistent = 0)

{

//consistent为1输出表头，否则不输出

if(!consistent)

cout << left << setw(10) << "考号" << setw(10) << "姓名" << setw(10)

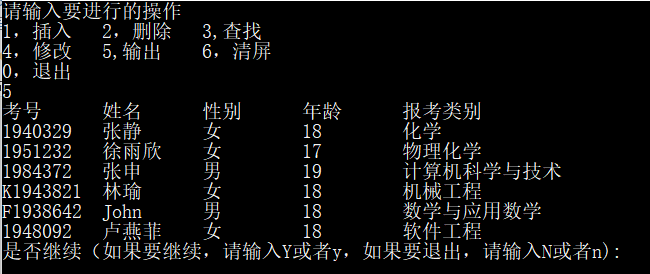
<< "性别" << setw(10) << "年龄" << setw(10) << "报考类别" << endl;

cout << left << setw(10) << e->num << setw(10) << e->name << setw(10)

<< e->sex << setw(10) << e->age << setw(10) << e->subject << endl;

}

### 3.5.3、统计部分截图展示：



## 3.6、总体系统的实现：

### 3.6.1、总体部分的流程

开始

输入数据

清屏

显示提示信息

输入操作

执行操作

是否继续？

结束

是

否

### 3.6.2、核心部分代码：

<1>、构造info\_system类，输入数据并且进入操作界面：

info\_system is;

is.display();

int ope;

cout << "输入已完成，马上跳转进入操作页面！";

//调用Sleep，暂停三秒

Sleep(3000);

system("cls");

<2>、进入循环：

while (1)

{

//输入表格：

cout << "请输入要进行的操作\n";

cout << setw(10) << left << "1，插入" << setw(10) << left << "2，删除"

<< setw(10) << left << "3,查找" << endl;

cout << setw(10) << left << "4，修改" << setw(10) << left << "5,输出" << setw(10)

<< left << "6，清屏" << endl << setw(10) << left << "0，退出" << endl;

//判断操作数是否正确

ope = getNum();

bool next\_loop = 0;

while (ope == -1)

{

cout << "输入格式有误，请重新输入！" << endl;

ope = getNum();

}

if (!ope)

break;

switch (ope)

{

case 1:

//插入

is.insert();

break;

case 2:

//删除

is.del();

break;

case 3:

//查找

is.search();

break;

case 4:

//修改

is.revise();

break;

case 5:

//统计

is.display();

break;

case 6:

//清屏

system("cls");

next\_loop = 1;

break;

default:

//输入有误提示

cout << "输入有误，请重新输入！" << endl;

}

if (next\_loop)continue;//如果是清屏操作，直接进入下一轮循环

//判断是否要继续

cout << "是否继续（如果要继续，请输入Y或者y，如果要退出，请输入N或者n): " << endl;

bool ex = 0;

while (1)

{

string sig;

cin >> sig;

if (sig == "Y" || sig == "y")

break;

else if (sig == "N" || sig == "n")

{

ex = 1;

break;

}

else

cout << "错误输入" << endl;

}

if (ex)

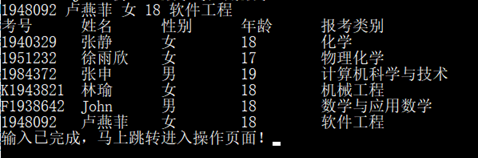
break;

}

<3>、退出，并输入报名系统中考生人数：

cout << "已退出，考试报名系统中共有 " << is.getSize() << " 个数据" << endl;

### 3.6.3、主界面展示：



# 4、测试

## 4.1、基本测试：

测试用例：

1940329 张静 女 18 化学

1951232 徐雨欣 女 17 物理化学

1984372 张申 男 19 计算机科学与技术

K1943821 林瑜 女 18 机械工程

F1938642 John 男 18 数学与应用数学

1948092 卢燕菲 女 18 软件工程

预期结果：

考号 姓名 性别 年龄 报考类别

1940329 张静 女 18 化学

1951232 徐雨欣 女 17 物理化学

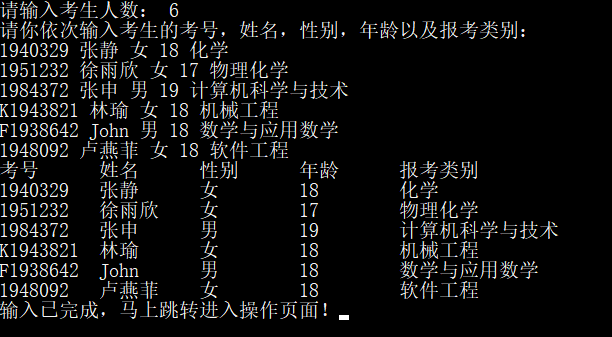
1984372 张申 男 19 计算机科学与技术

K1943821 林瑜 女 18 机械工程

F1938642 John 男 18 数学与应用数学

1948092 卢燕菲 女 18 软件工程

实验结果：



### 4.1.1、插入数据测试：

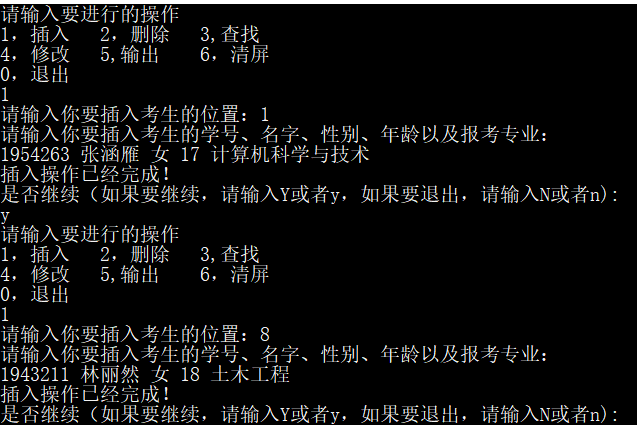
测试用例：

在1号为插入1954263 张涵雁 女 17 计算机科学与技术，

在8号位插入1943211 林丽然 女 18 土木工程，

在5号位插入1947388 湘妍 女 18 统计学

插入过程截图如下：



预期结果：

考号 姓名 性别 年龄 报考类别

1954263 张涵雁 女 17 计算机科学与技术

1940329 张静 女 18 化学

1951232 徐雨欣 女 17 物理化学

1984372 张申 男 19 计算机科学与技术

1947388 湘妍 女 18 统计学

K1943821 林瑜 女 18 机械工程

F1938642 John 男 18 数学与应用数学

1948092 卢燕菲 女 18 软件工程

1943211 林丽然 女 18 土木工程

实验结果：



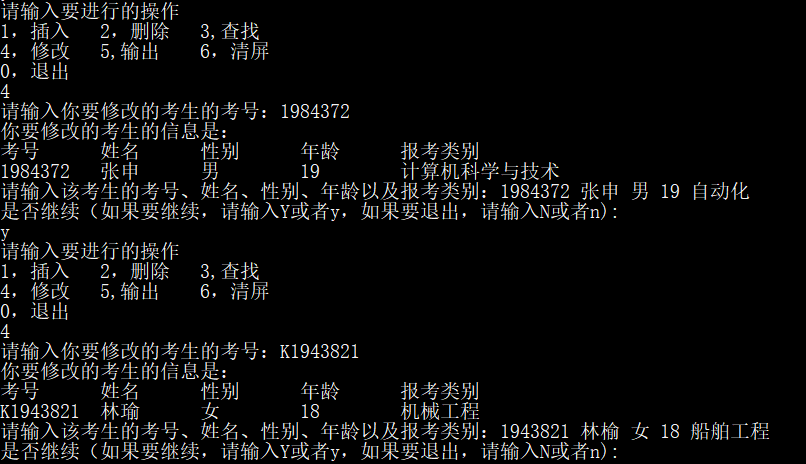
### 4.1.2、修改数据测试：

测试用例：

将考号位1984372的信息改为1984372 张申 男 19 自动化

将考号为K1943821的信息改为1943821 林榆 女 18 船舶工程

修改过程截图如下：



预期结果：

考号 姓名 性别 年龄 报考类别

1954263 张涵雁 女 17 计算机科学与技术

1940329 张静 女 18 化学

1951232 徐雨欣 女 17 物理化学

1984372 张申 男 19 自动化

1947388 湘妍 女 18 统计学

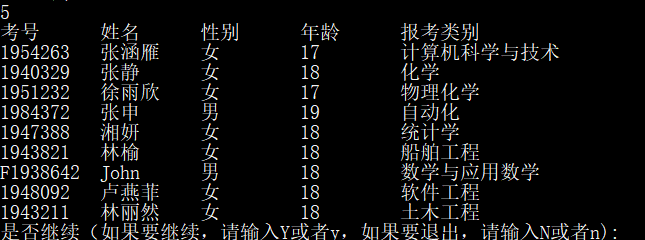
1943821 林榆 女 18 船舶工程

F1938642 John 男 18 数学与应用数学

1948092 卢燕菲 女 18 软件工程

1943211 林丽然 女 18 土木工程

实验结果：



### 4.1.3、查询数据测试：

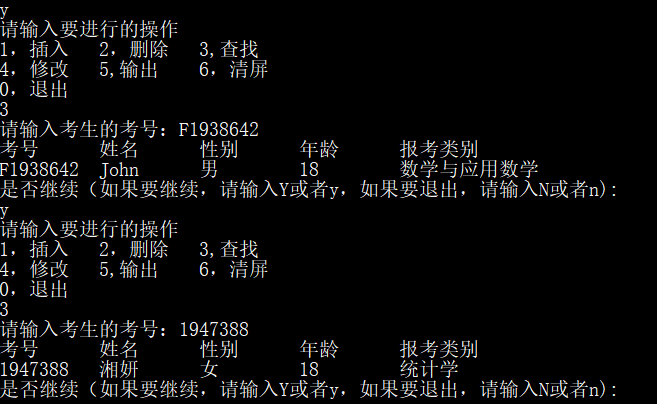
查询的考号为：F1938642和1947388

预期结果：  
考号 姓名 性别 年龄 报考类别

F1938642 John 男 18 数学与应用数学

考号 姓名 性别 年龄 报考类别

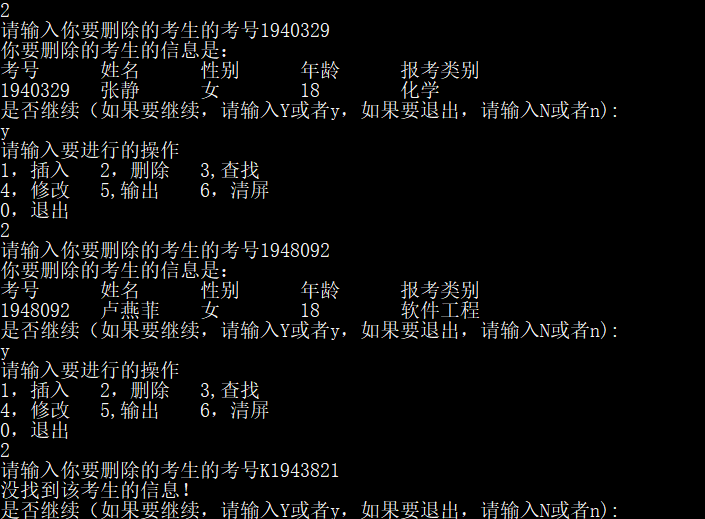
1947388 湘妍 女 18 统计学

实验结果：  


### 4.1.4、删除数据测试：

测试用例：要删除的数据：1940329、1948092和K1943821

删除操作截图如下：



预期结果：

考号 姓名 性别 年龄 报考类别

1954263 张涵雁 女 17 计算机科学与技术

1951232 徐雨欣 女 17 物理化学

1984372 张申 男 19 自动化

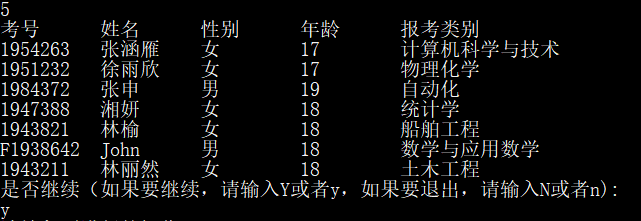
1947388 湘妍 女 18 统计学

1943821 林榆 女 18 船舶工程

F1938642 John 男 18 数学与应用数学

1943211 林丽然 女 18 土木工程

实验结果：



### 4.1.5统计功能测试：

预期结果：

1954263 张涵雁 女 17 计算机科学与技术

1951232 徐雨欣 女 17 物理化学

1984372 张申 男 19 自动化

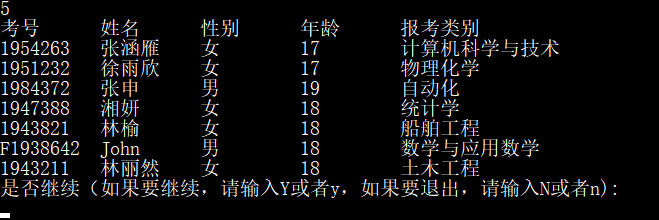
1947388 湘妍 女 18 统计学

1943821 林榆 女 18 船舶工程

F1938642 John 男 18 数学与应用数学

1943211 林丽然 女 18 土木工程

实验结果：



## 4.2、边界情况测试：

### 4.2.1、删除后链表为空：

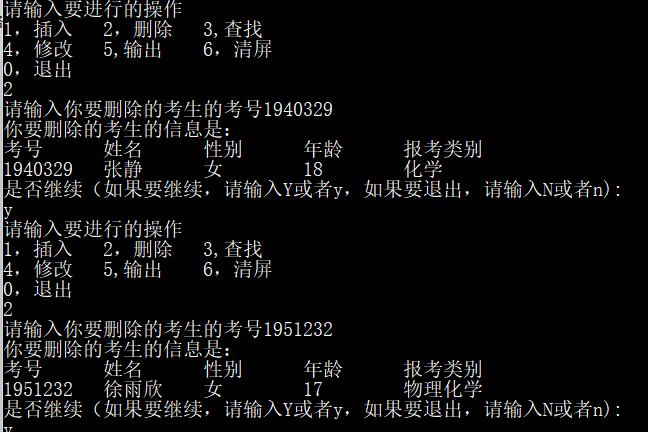
测试用例：

1940329 张静 女 18 化学

1951232 徐雨欣 女 17 物理化学

根据学号删除这两个节点

测试过程截图：



结果预期：

告知用户系统中没有数据

实验结果：

