项目说明文档

数据结构课程设计

——考试报名系统

作 者 姓 名： 杨晶

学 号： 1854025

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](#_Toc495668153)

[1.1 背景分析](#_Toc495668154)

[1.2 功能分析](#_Toc495668155)

[2 设计](#_Toc495668156)

[2.1 数据结构设计](#_Toc495668157)

[2.2 类结构设计](#_Toc495668158)

[2.3 成员与操作设计](#_Toc495668159)

[2.4 系统设计](#_Toc495668160)

[3 实现](#_Toc495668161)

[3.1 插入功能的实现](#_Toc495668162)

[3.1.1 插入功能流程图](#_Toc495668163)

[3.1.2 插入功能核心代码](#_Toc495668164)

[3.1.3 插入功能截屏示例](#_Toc495668165)

[3.2 删除功能的实现](#_Toc495668166)

[3.2.1 删除功能流程图](#_Toc495668167)

[3.2.2 删除功能核心代码](#_Toc495668168)

[3.2.3 删除功能截屏示例](#_Toc495668169)

[3.3 查找功能的实现](#_Toc495668170)

[3.3.1 查找功能流程图](#_Toc495668171)

[3.3.2 查找功能核心代码](#_Toc495668172)

[3.3.3 查找功能截图示例](#_Toc495668173)

[3.4 修改功能的实现](#_Toc495668174)

[3.4.1 修改功能流程图](#_Toc495668175)

[3.4.2 修改功能核心代码](#_Toc495668176)

[3.4.3 修改功能截屏示例](#_Toc495668177)

[3.5 统计功能的实现](#_Toc495668178)

[3.5.1 统计功能流程图](#_Toc495668179)

[3.5.2 统计功能核心代码](#_Toc495668180)

[3.5.3 统计功能截屏示例](#_Toc495668181)

[3.6 总体系统的实现](#_Toc495668182)

[3.6.1 总体系统流程图](#_Toc495668183)

[3.6.2 总体系统核心代码](#_Toc495668184)

[3.6.3 总体系统截屏示例](#_Toc495668185)

[4 测试](#_Toc495668186)

[4.1 功能测试](#_Toc495668187)

[4.1.1 插入功能测试](#_Toc495668188)

[4.1.2 删除功能测试](#_Toc495668189)

[4.1.3 查找功能测试](#_Toc495668190)

[4.1.4 修改功能测试](#_Toc495668191)

[4.1.5 统计功能测试](#_Toc495668192)

[4.2 边界测试](#_Toc495668193)

[4.2.1 初始化无输入数据](#_Toc495668194)

[4.2.2 删除头结点](#_Toc495668195)

[4.2.3 删除后链表为空](#_Toc495668196)

[4.3 出错测试](#_Toc495668197)

[4.3.1 考生人数错误](#_Toc495668198)

[4.3.2 操作码错误](#_Toc495668199)

[4.3.3 插入考号已存在](#_Toc495668200)

[4.3.4 删除考号不存在](#_Toc495668201)

[4.3.5 查找考号不存在](#_Toc495668202)

[4.3.6 修改考号不存在](#_Toc495668203)

# 1 分析

## 1.1 背景分析

随着计算机技术和互联网的快速发展，人类已经进入了信息时代，

## 1.2 功能分析

在

# 2 设计

## 2.1 数据结构设计

根据上面的功能分析，我们的考试报名系统需要有很多的增加节点，删除节点，修改节点的1操作。容易想到链表这一种基本的数据结构来完成我们的操作。

## 2.2 类结构设计

链表结构一般包括两个抽象数据类型——链表结点类与链表类。本考试报名系统采用class描述链表结点类以及链表类。

## 2.3 成员与操作设计

**链表结点类（Node）**

1. **class** Node
2. {
3. **public**:
4. **int** id;//考生号
5. string name;//姓名
6. string sex;//性别
7. **int** age;//年龄
8. string exam;//报考类别
9. Node\* next;
10. Node() { next = nullptr; }
12. **void** Display()
13. {
14. //输出考生节点信息
15. cout << id<< "\t" << name << "\t" << sex
16. << "\t" << age << "\t" << exam << endl;
17. **return**;
18. }
19. };

**链表类（LinkList）**

1. //链表类定义
2. **class** Link
3. {
4. **public**:
5. Node\* head;
6. **int** len;
7. Link()
8. {
9. len = 0;
10. head = **new** Node();
11. }
13. **void** AddNode(Node\* s);//增加考生节点
14. **void** DeleteNode(**int** id);//删除考生节点
15. Node\* FindNode(**int** id);//寻找考生并返回这个节点
16. **void** ModifyNode(**int** id);//修改考生信息
17. **void** Traverse();//按考号排序并输出考生信息
18. };

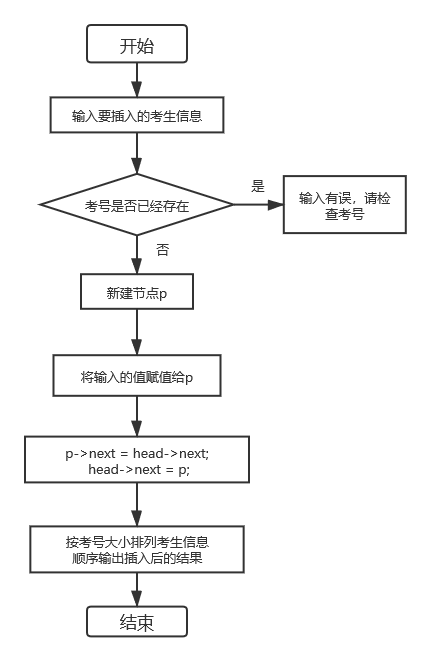
## 2.4 系统设计

系统首先调用InitScreen()函数实现对屏幕的初始化，完成对链表L的创建和输入数据工作，然后根据用户所输入的操作码（operatorCode）执行链表L对应的成员函数。

# 3 实现

## 3.1 插入功能的实现

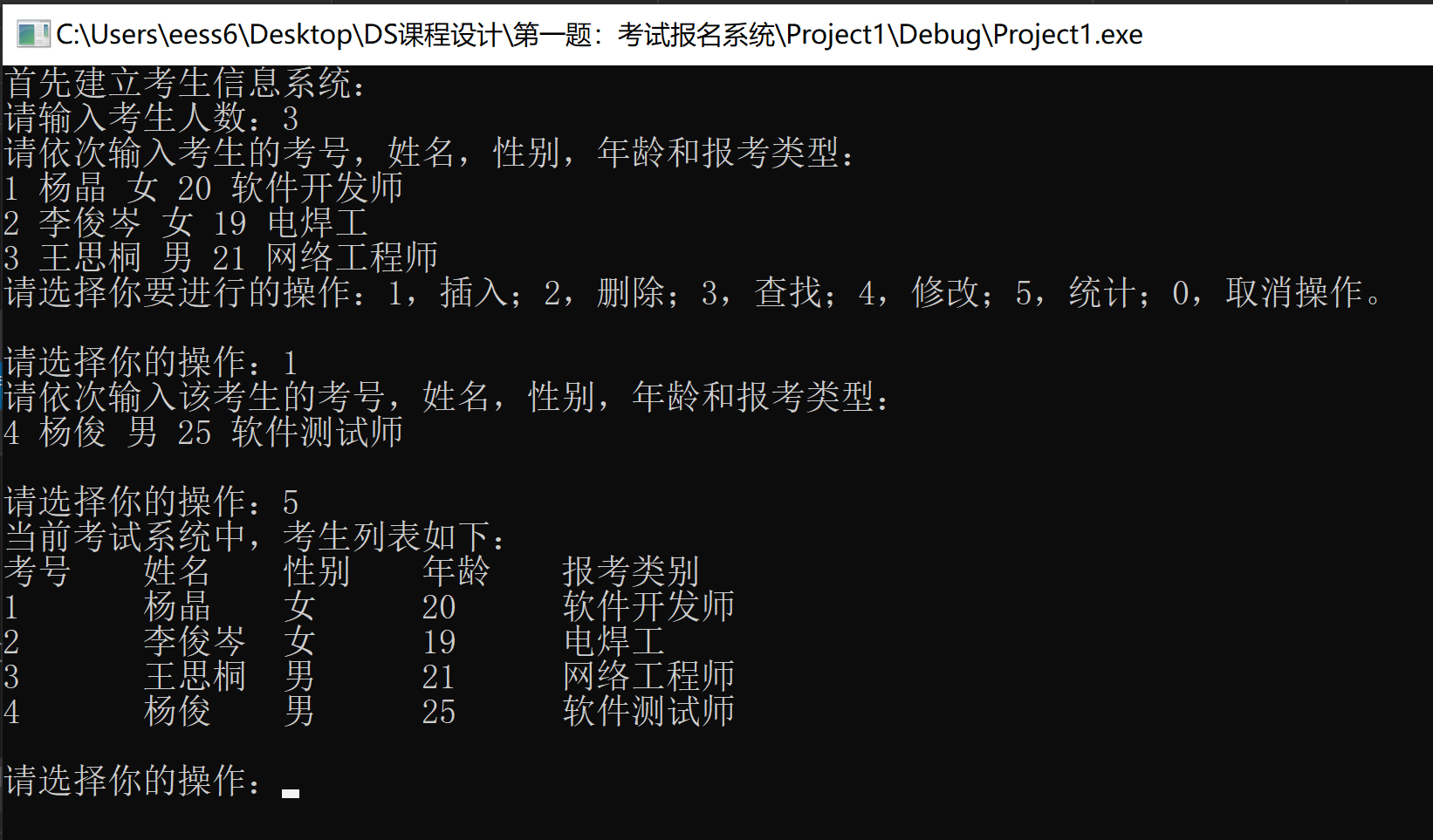
### 3.1.1 插入功能流程图



### 3.1.2 插入功能核心代码

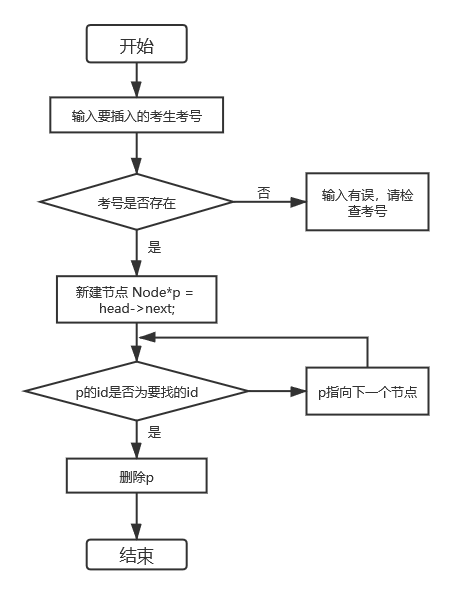
1. //增加考生节点
2. **void** Link::AddNode(Node\* s)
3. {
4. Node\*p = **new** Node();
5. p = s;
6. p->next = head->next;
7. head->next = p;
8. len++;
9. }
10. **case** 1://插入考生
11. cout << "请依次输入该考生的考号，姓名，性别，年龄和报考类型：" << endl;
12. cin >> s->id >> s->name >> s->sex >> s->age >> s->exam;
13. **if**(StudentList.FindNode(s->id)==nullptr)
14. StudentList.AddNode(s);
15. **else** **if**(StudentList.FindNode(s->id) != nullptr)
16. cout << "插入失败，该考号已存在学生，请检查输入考号是否正确。" << endl;
17. **break**;

### 3.1.3 插入功能截屏示例



## 3.2 删除功能的实现

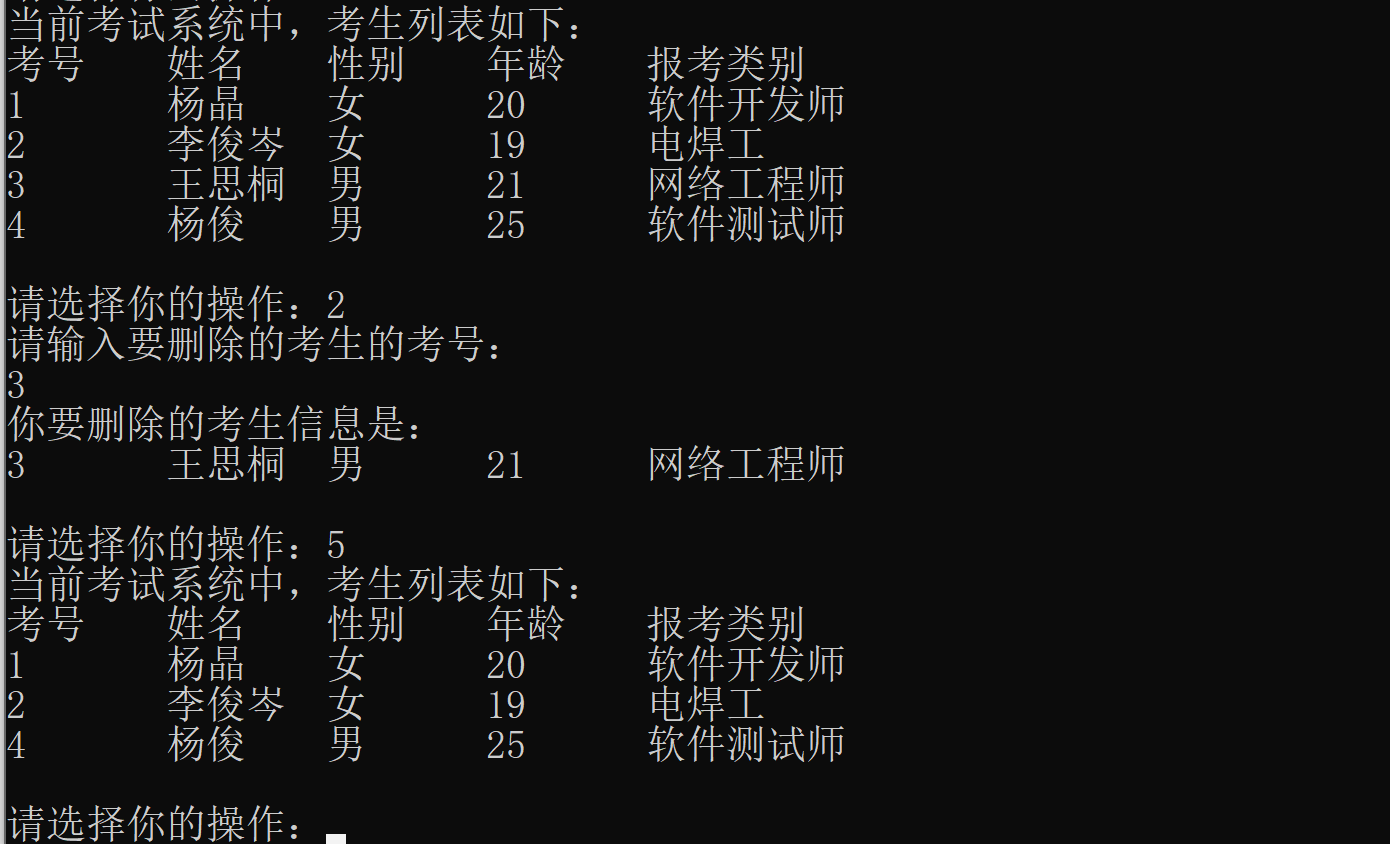
### 3.2.1 删除功能流程图

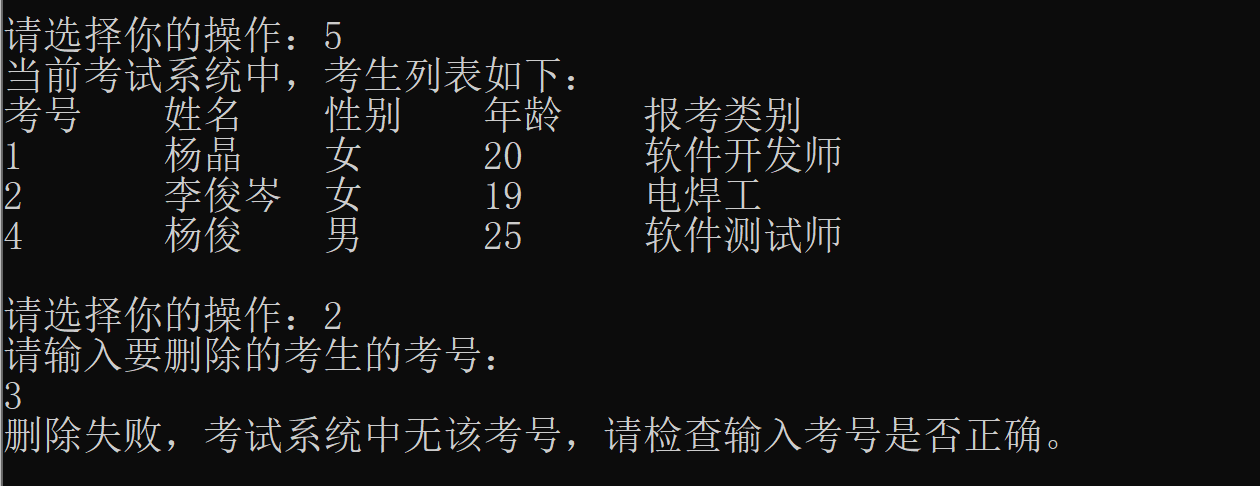


### 3.2.2 删除功能核心代码

1. //删除考生节点
2. **void** Link::DeleteNode(**int** id)
3. {
4. Node\*q = head;
5. Node\*p = head->next;
6. **while** (p != nullptr)
7. {
8. **if** (p->id == id)
9. {
10. q->next = p->next;
11. cout << "你要删除的考生信息是：" << endl;
12. p->Display();
13. **delete** p;
14. len--;
15. **break**;
16. }
17. **else**
18. {
19. p = p->next;
20. q = q->next;
21. }
22. }
23. }
24. **case** 2://删除一个考生
25. cout << "请输入要删除的考生的考号：" << endl;
26. cin >> temp;
27. **if** (StudentList.FindNode(temp) != nullptr)
28. StudentList.DeleteNode(temp);
29. **else** cout << "删除失败，考试系统中无该考号，请检查输入考号是否正确。" << endl;
30. **break**;

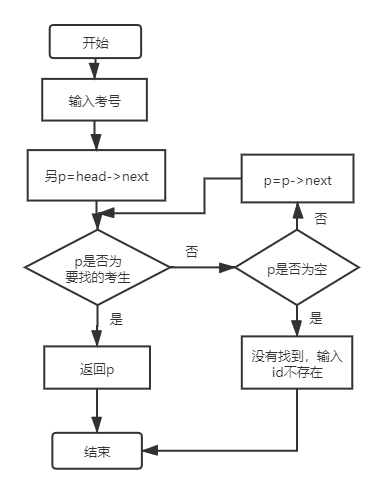
### 3.2.3 删除功能截屏示例





## 3.3 查找功能的实现

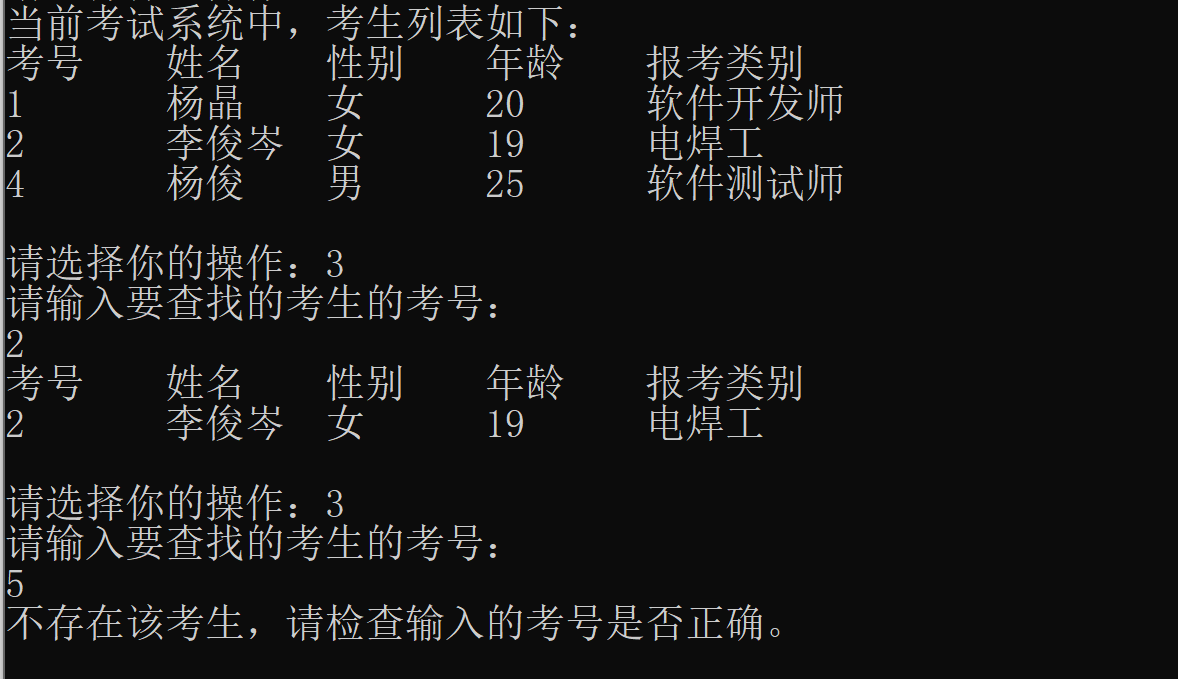
### 3.3.1 查找功能流程图



### 3.3.2 查找功能核心代码

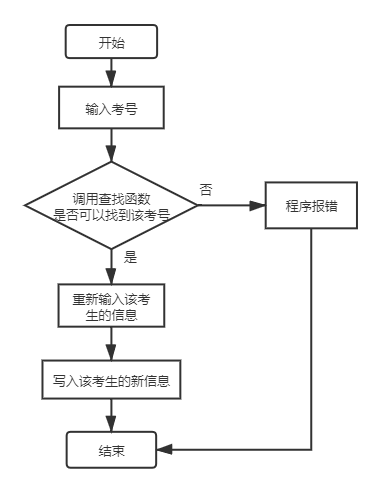
1. //寻找考生并返回这个节点
2. Node\* Link::FindNode(**int** id)
3. {
4. Node\*p = head->next;
5. **while** (p != nullptr)
6. {
7. **if** (p->id == id)
8. {
9. **return** p;
10. **break**;
11. }
12. **else**
13. {
14. p = p->next;
15. }
16. }
17. **return** nullptr;
18. }
19. **case** 3://查找一个考生
20. cout << "请输入要查找的考生的考号：" << endl;
21. cin >> temp;
22. **if** (StudentList.FindNode(temp) != nullptr)
23. {
24. cout << "考号" << "\t" << "姓名" << "\t" << "性别"
25. << "\t" << "年龄" << "\t" << "报考类别" << endl;
26. StudentList.FindNode(temp)->Display();
27. }
28. **else** cout << "不存在该考生，请检查输入的考号是否正确。" << endl;
29. **break**;

### 3.3.3 查找功能截图示例



## 3.4 修改功能的实现

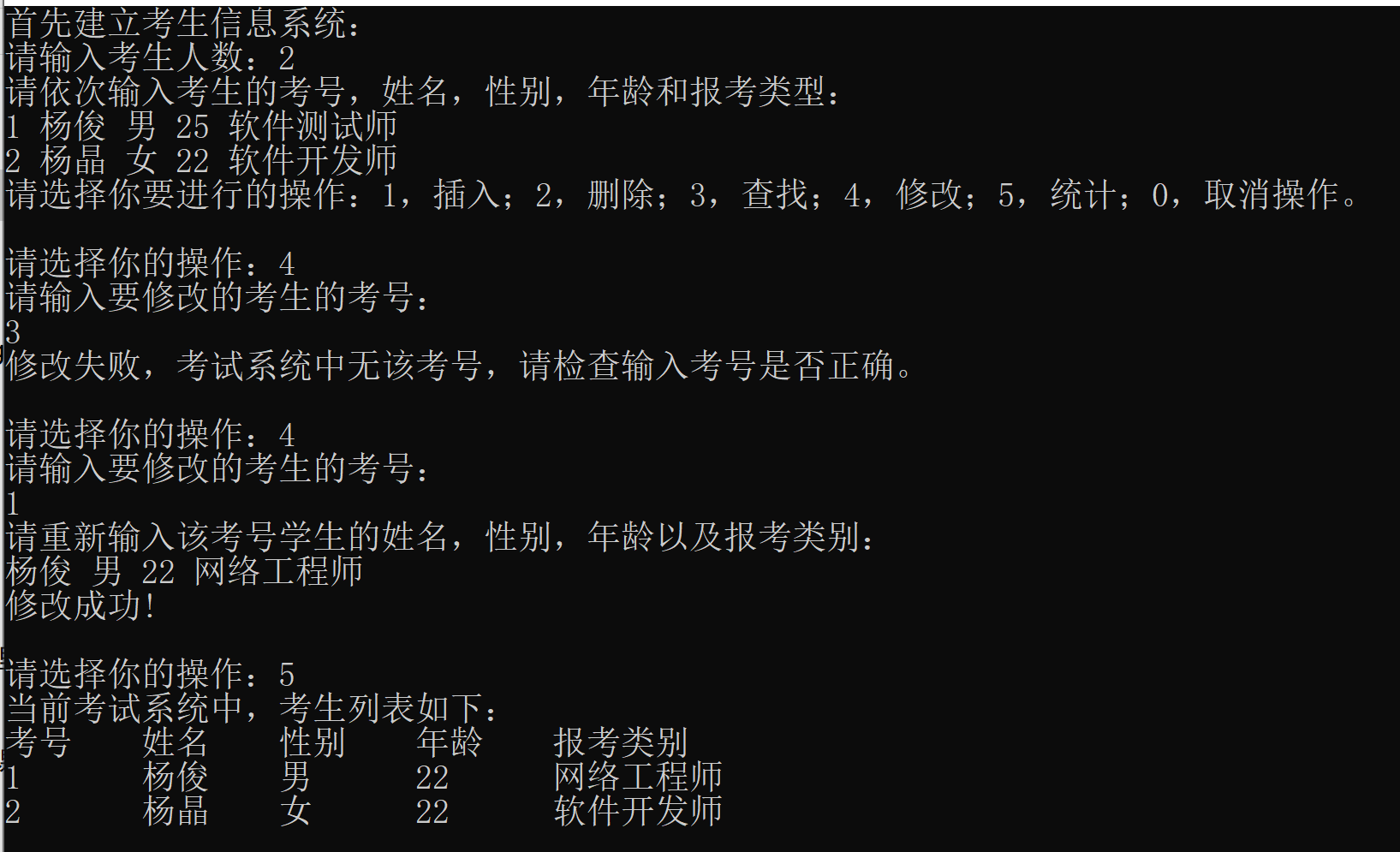
### 3.4.1 修改功能流程图



### 3.4.2 修改功能核心代码

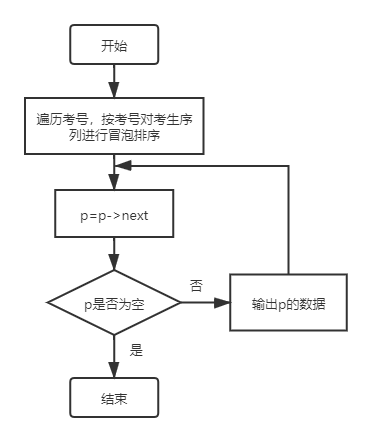
1. //修改考生信息
2. **void** Link::ModifyNode(**int** id)
3. {
4. Node\*p = FindNode(id);
5. cout << "请重新输入该考号学生的姓名，性别，年龄以及报考类别：" << endl;
6. cin >> p->name >> p->sex >> p->age >> p->exam;
7. cout << "修改成功!" << endl;
8. }
9. **case** 4://修改其中一个考生信息
10. cout << "请输入要修改的考生的考号：" << endl;
11. cin >> temp;
12. **if** (StudentList.FindNode(temp) != nullptr)
13. StudentList.ModifyNode(temp);
14. **else** cout << "修改失败，考试系统中无该考号，请检查输入考号是否正确。 " << endl;
15. **break**;

### 3.4.3 修改功能截屏示例



## 3.5 统计功能的实现

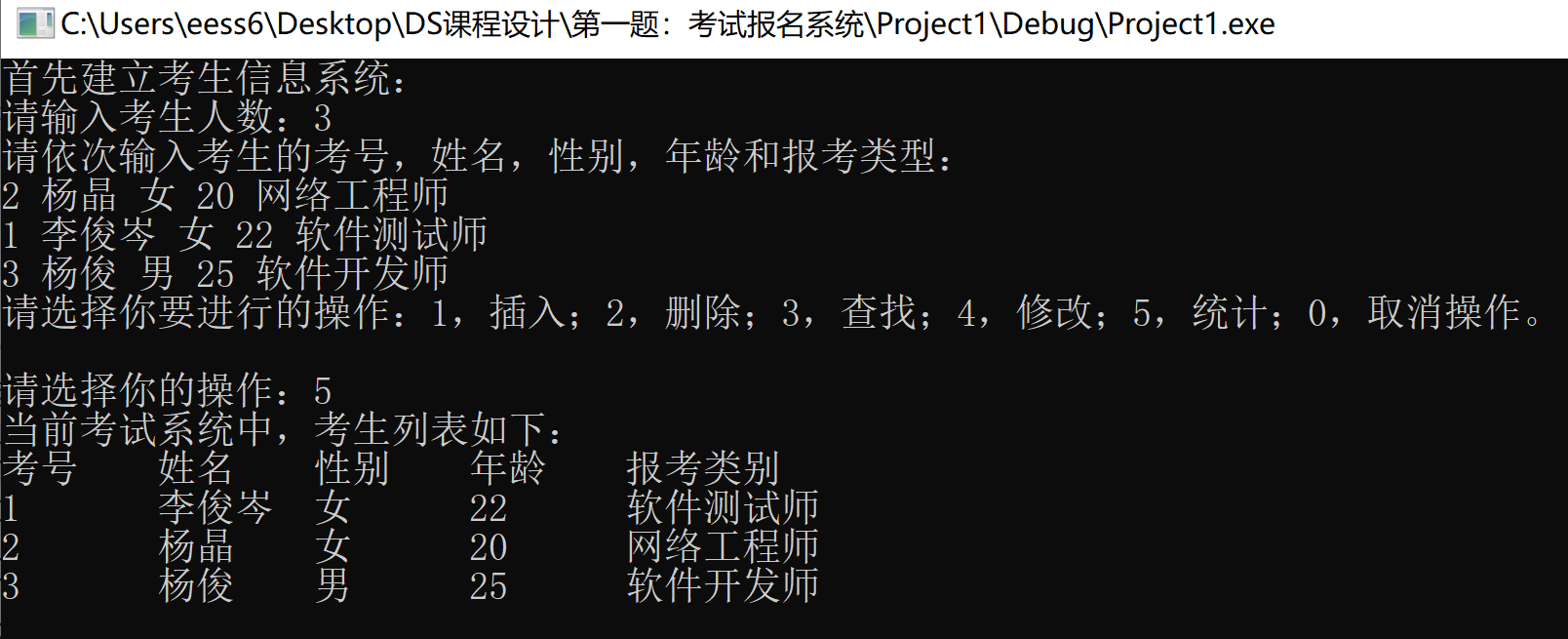
### 3.5.1 统计功能流程图



### 3.5.2 统计功能核心代码

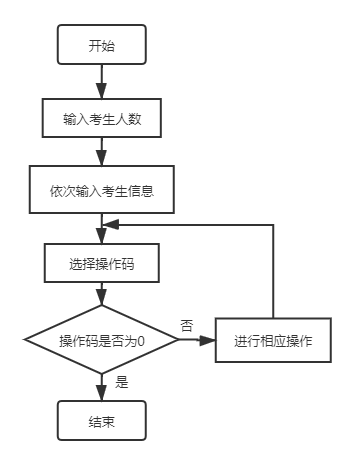
1. //按考号排序并输出考生信息
2. **void** Link::Traverse()
3. {
4. Node\* p = **new** Node();
6. //冒泡排序
7. **for** (**int** i = 0; i < len; i++)
8. {
9. **for** (p = head->next; p->next != nullptr; p = p->next)
10. {
11. **if** (p->id > p->next->id)
12. {
13. swap(p->id, p->next->id);
14. swap(p->name, p->next->name);
15. swap(p->age, p->next->age);
16. swap(p->sex, p->next->sex);
17. swap(p->exam, p->next->exam);
18. }
19. }
20. }
22. cout << "考号" <<"\t"<< "姓名" << "\t" << "性别"
23. << "\t" << "年龄" << "\t" << "报考类别" << endl;
25. p = head->next;
26. **while** (p != nullptr) {
27. p->Display();
28. p = p->next;
29. }
30. }

### 3.5.3 统计功能截屏示例



## 3.6 总体系统的实现

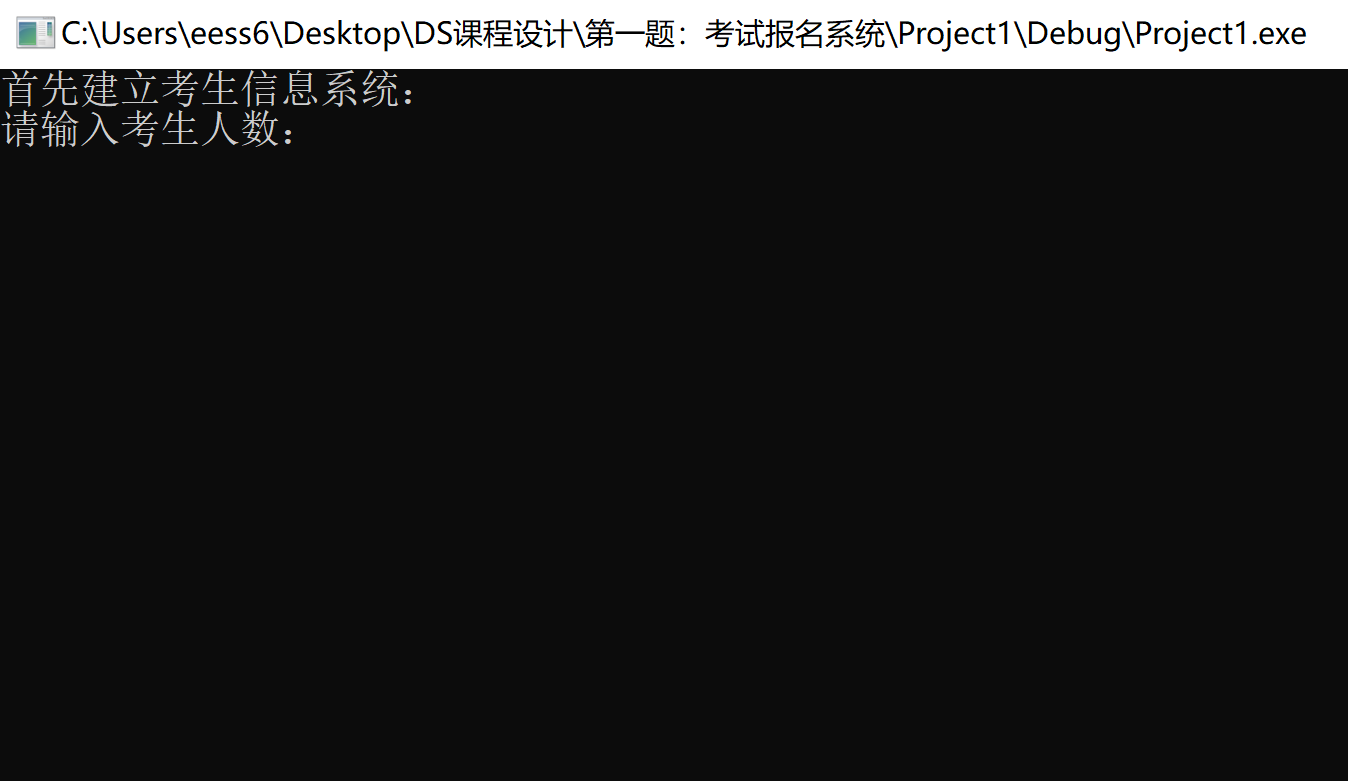
### 3.6.1 总体系统流程图

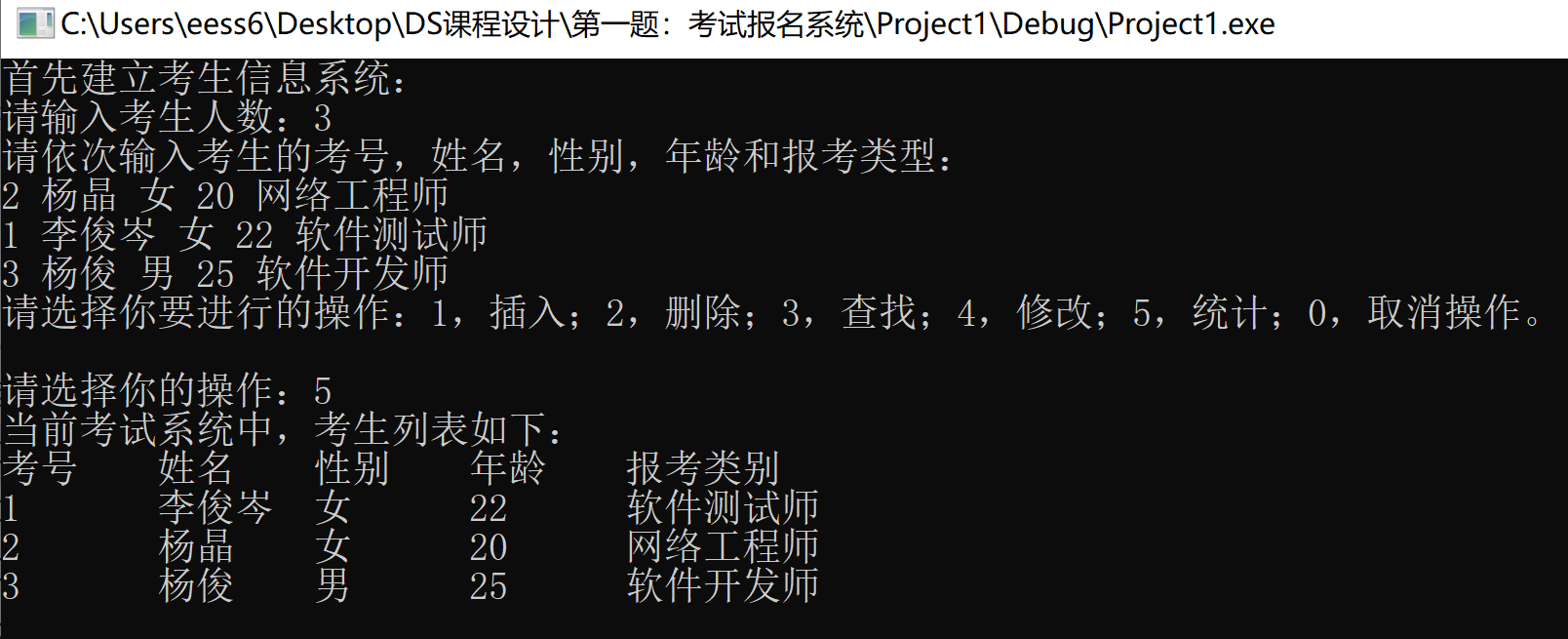


### 3.6.2 总体系统核心代码

1. //测试循环
2. **void** TestLoop()
3. {
4. **int** num;
5. cout << "首先建立考生信息系统：" << endl << "请输入考生人数：";
6. cin >> num;
7. cout << "请依次输入考生的考号，姓名，性别，年龄和报考类型：" << endl;
9. Link StudentList;
10. //初始化考生信息
11. **for** (**int** i = 0; i < num; i++)
12. {
13. Node\* p=**new** Node();
14. cin >> p->id >> p->name >> p->sex >> p->age >> p->exam;
15. StudentList.AddNode(p);
16. }
18. cout << "请选择你要进行的操作：1，插入；2，删除；3，查找；4，修改；5，统计；0，取消操作。" << endl;
20. **while** (1)
21. {
22. cout << endl << "请选择你的操作：";
24. **int** operate;
25. cin >> operate;
26. **int** temp;
27. Node\* s = **new** Node();
28. **switch** (operate)
29. {
30. **case** 1://插入考生
31. cout << "请依次输入该考生的考号，姓名，性别，年龄和报考类型：" << endl;
32. cin >> s->id >> s->name >> s->sex >> s->age >> s->exam;
33. **if**(StudentList.FindNode(s->id)!=nullptr)
34. StudentList.AddNode(s);
35. **else** cout << "插入失败，该考号已存在学生，请检查输入考号是否正确。" << endl;
36. **break**;
37. **case** 2://删除一个考生
38. cout << "请输入要删除的考生的考号：" << endl;
39. cin >> temp;
40. **if** (StudentList.FindNode(temp) != nullptr)
41. StudentList.DeleteNode(temp);
42. **else** cout << "删除失败，考试系统中无该考号，请检查输入考号是否正确。" << endl;
43. **break**;
44. **case** 3://查找一个考生
45. cout << "请输入要查找的考生的考号：" << endl;
46. cin >> temp;
47. **if** (StudentList.FindNode(temp) != nullptr)
48. {
49. cout << "考号" << "\t" << "姓名" << "\t" << "性别"
50. << "\t" << "年龄" << "\t" << "报考类别" << endl;
51. StudentList.FindNode(temp)->Display();
52. }
53. **else** cout << "不存在该考生，请检查输入的考号是否正确。" << endl;
54. **break**;
55. **case** 4://修改其中一个考生信息
56. cout << "请输入要修改的考生的考号：" << endl;
57. cin >> temp;
58. **if** (StudentList.FindNode(temp) != nullptr)
59. StudentList.ModifyNode(temp);
60. **else** cout << "修改失败，考试系统中无该考号，请检查输入考号是否正确。" << endl;
61. **break**;
62. **case** 5://按序打印当前考生列表
63. cout << "当前考试系统中，考生列表如下：" << endl;
64. StudentList.Traverse();
65. **break**;
66. **case** 0:
67. **return**;
68. }
69. }
70. **return**;
71. }
73. //主函数
74. **int** main()
75. {
76. TestLoop();
77. **return** 0;
78. }

### 3.6.3 总体系统截屏示例





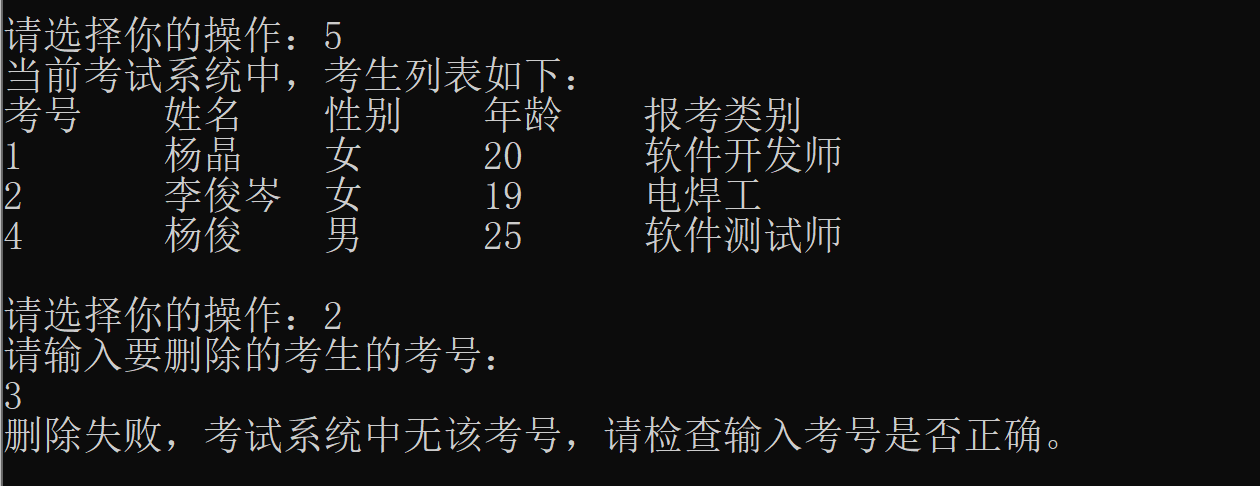
# 4 测试

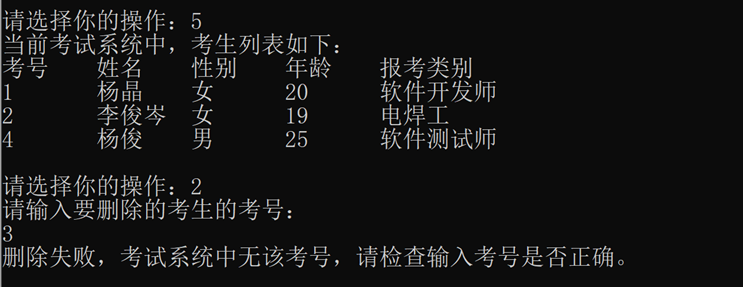
## 4.1 功能测试

### 4.1.1 插入功能测试

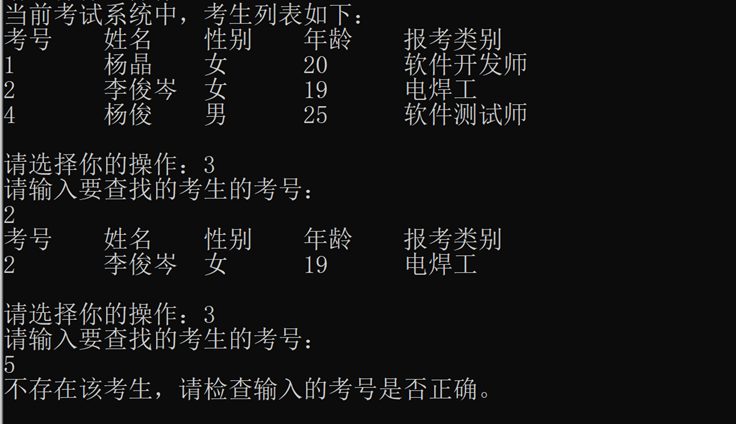


### 4.1.2 删除功能测试

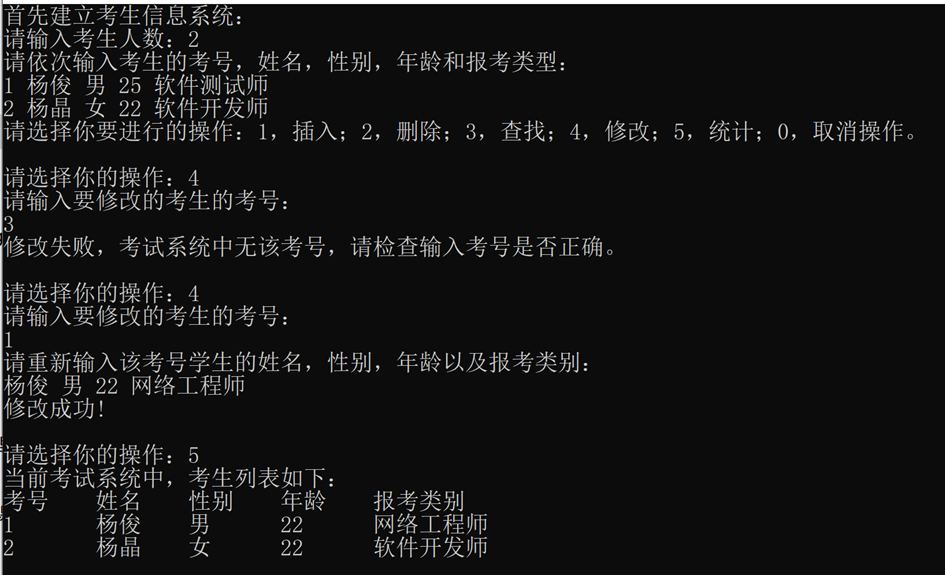




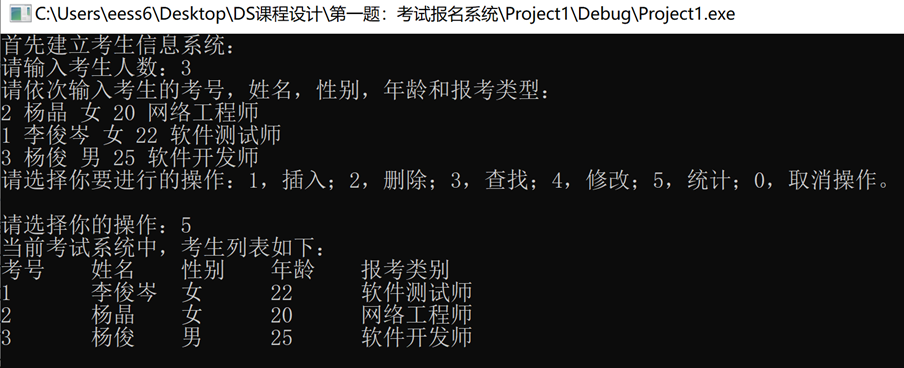
### 4.1.3 查找功能测试



### 4.1.4 修改功能测试



### 4.1.5 统计功能测试



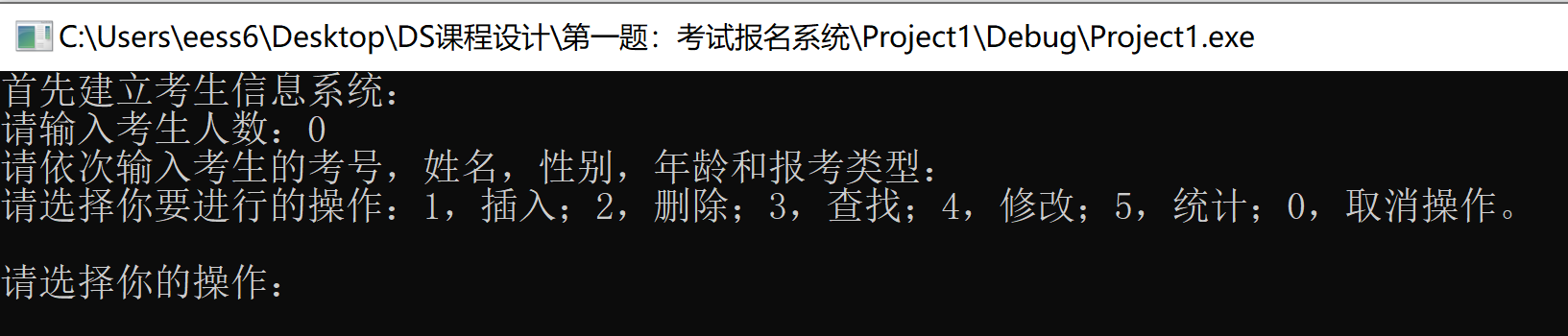
## 4.2 边界测试

### 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：**程序运行正常不崩溃。

**实验结果：**

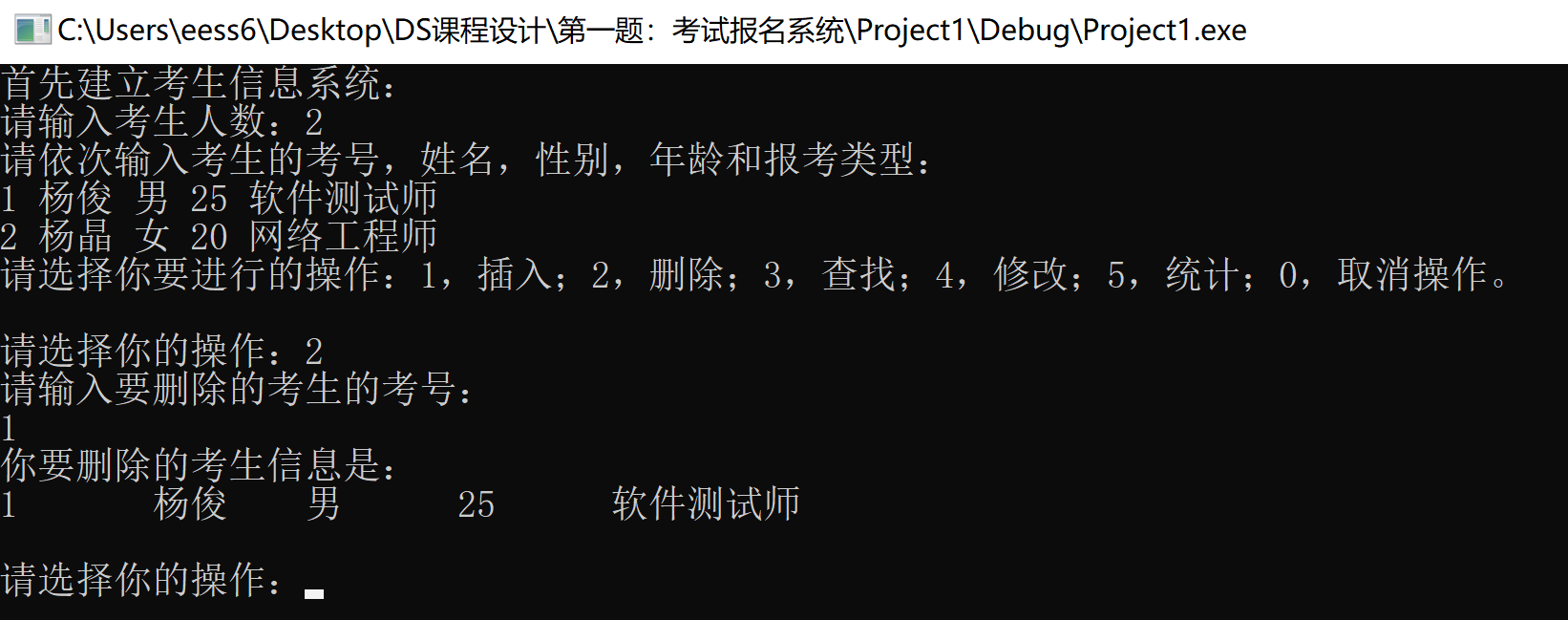


### 4.2.2 删除头结点

**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：**

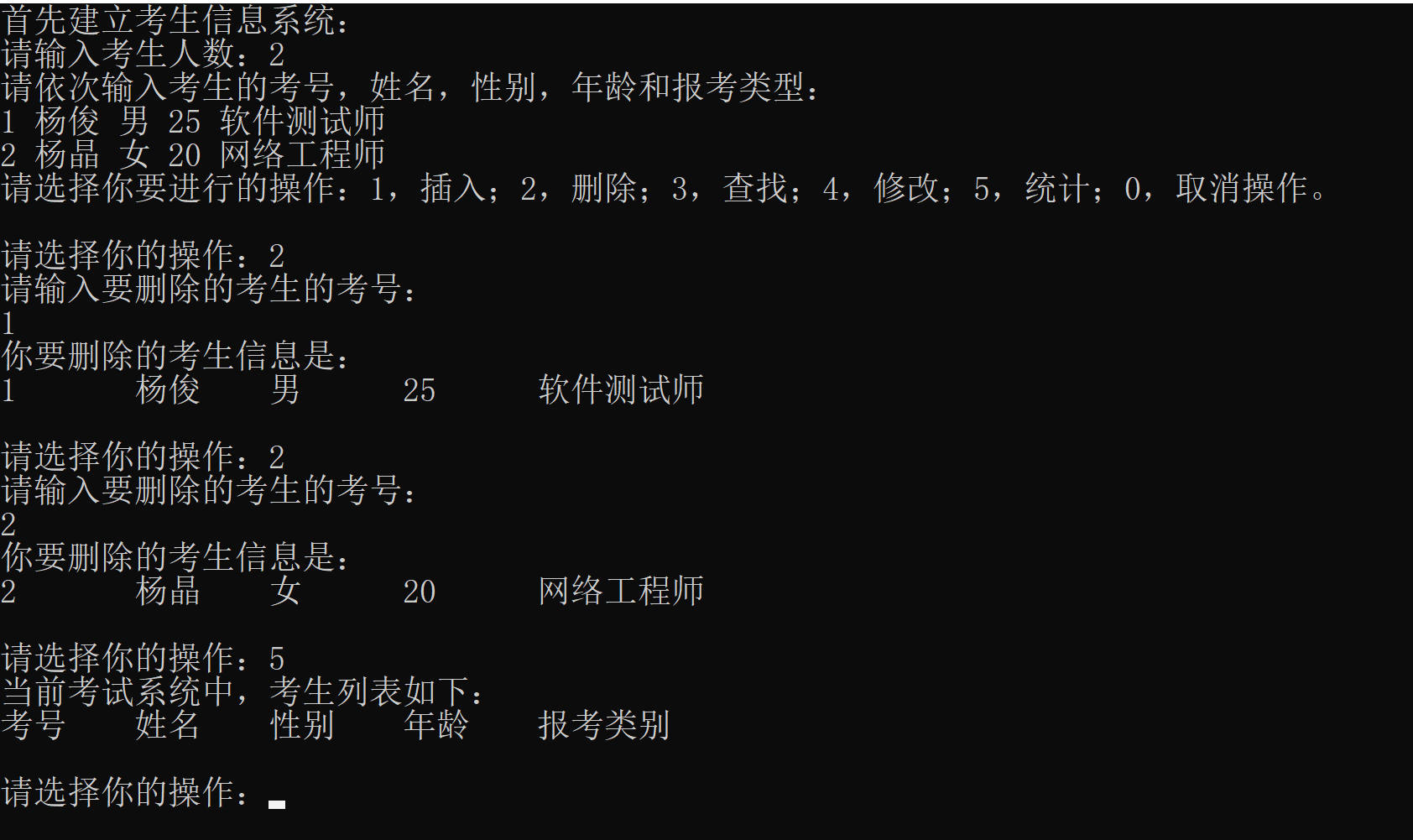


### 4.2.3 删除后链表为空

**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：**

****

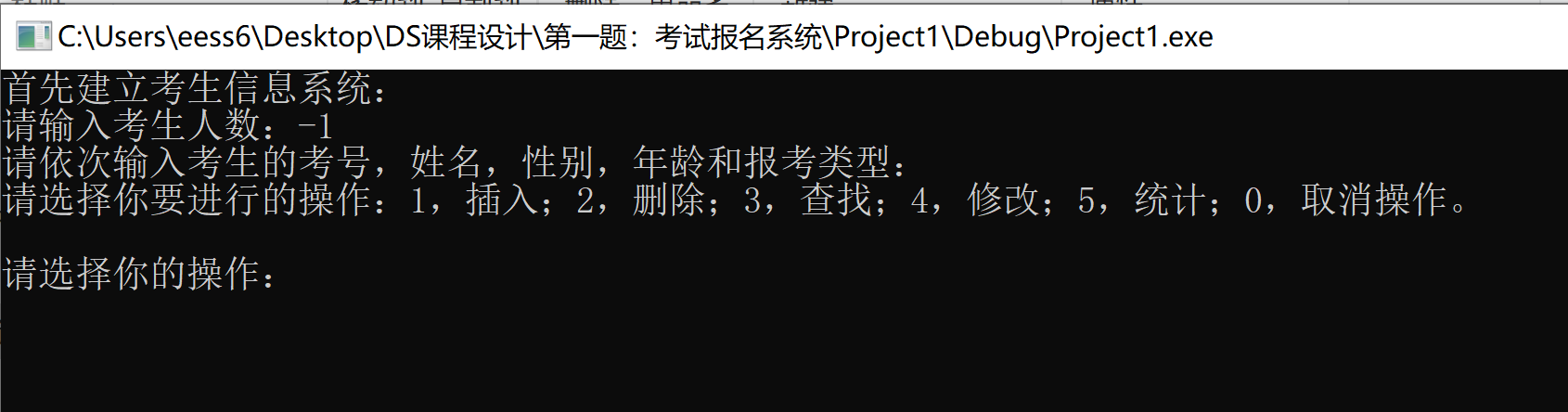
## 4.3 出错测试

### 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：**程序正常运行不崩溃，只是在接下来的操作无效。

**实验结果：**

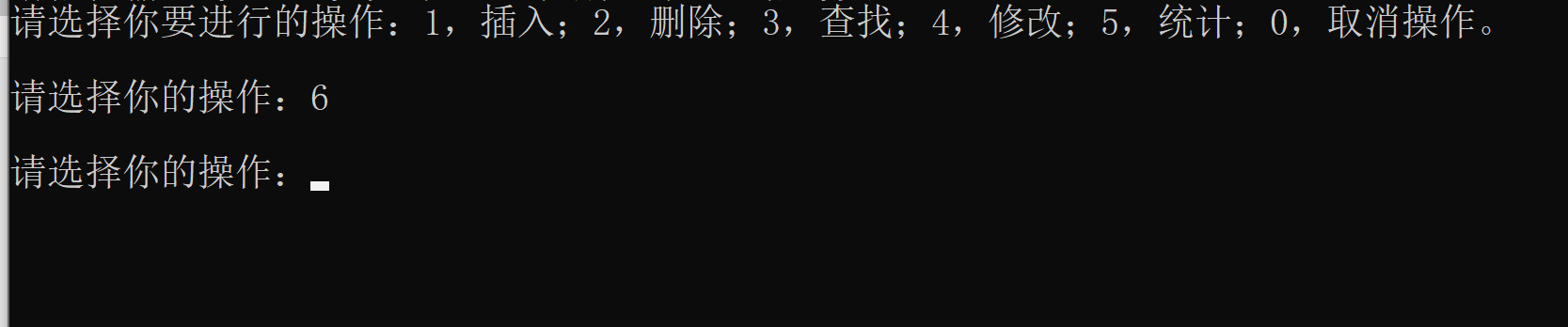
****

### 4.3.2 操作码错误

**测试用例：**输入操作码错误

**预期结果：**程序重新给出操作提示信息，程序正常运行不崩溃。

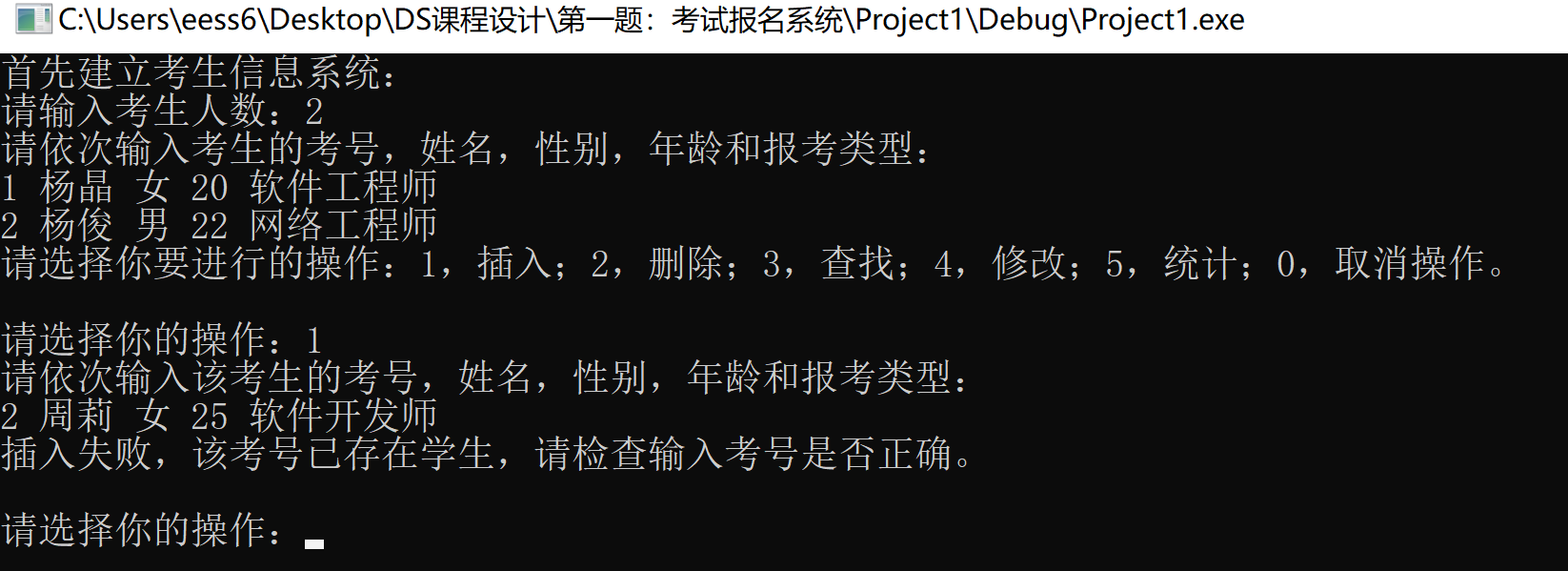
**实验结果：**

****

### 4.3.3 插入考号已存在

**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：**

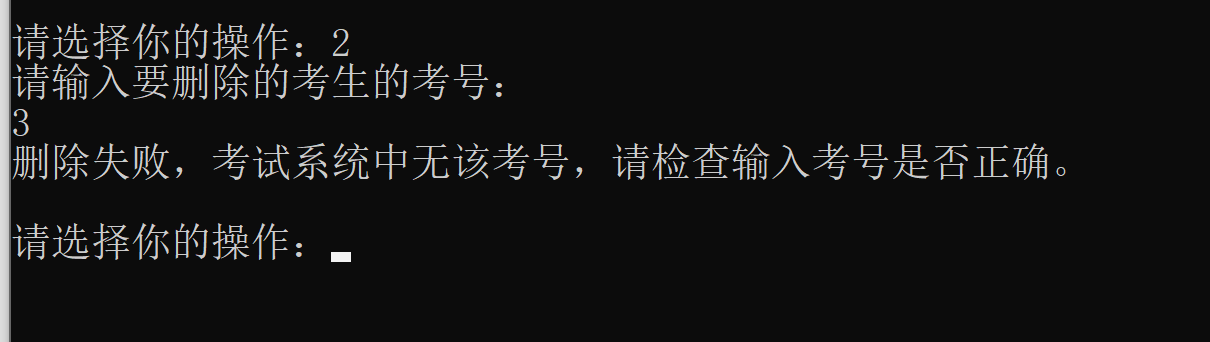


### 4.3.4 删除考号不存在

**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：**

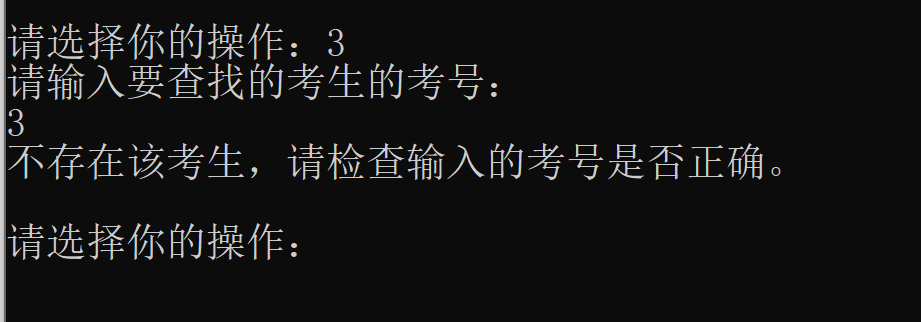
****

### 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：**

