项目说明文档

数据结构课程设计

**——**考试报名系统

作 者 姓 名： 罗吉皓

学 号： 1652792

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 5

1.1 项目名称 ： 考试报名系统 5

1.2 项目背景 5

1.3 项目功能分析 5

II. 设计 6

2.1 数据结构设计 6

2.2 数据结构类的设计 6

2.3 系统设计 6

2.3.1 考生数据类型类（exam) 6

2.3.2 考生链表类（examinee） 7

2.3.3 考试报名系统类（examineeInformationSystem) 8

2.4 系统设计 8

输入功能截屏示例： 9

III 实现 10

3.1 插入功能的实现 10

3.1.1 插入功能流程图 11

3.1.2 思路分析 12

3.1.3 具体实现如下： 12

3.1.4 插入功能截屏示例 13

3.2 删除功能的实现 13

3.2.1 删除功能流程图 14

3.2.2 思路分析 15

3.2.3 具体实现如下： 15

3.2.4 删除功能截屏示例 16

3.3 搜索功能的实现 16

3.3.1 搜索功能流程图 17

3.3.2 思路分析 18

3.3.3 具体实现如下： 18

3.3.4 搜索功能截屏示例 19

3.4 修改功能的实现 19

3.4.1 修改功能流程图 20

3.4.2 思路分析 21

3.4.3 具体实现如下： 21

3.4.4 修改功能截屏示例 21

3.5 统计功能的实现 23

3.5.1 统计功能流程图 23

3.5.2 思路分析 23

3.5.3 具体实现如下： 24

3.5.4 统计功能截屏示例 24

3.6 总体系统的实现 25

3.6.1 总体系统流程图 25

3.6.2 思路分析 26

3.6.3 具体实现如下： 27

3.6.4 总体系统截屏示例 27

IV 测试 28

4.1 功能测试 28

4.1.0 建立系统测试 28

4.1.1 插入功能测试 28

4.1.2 删除功能测试 29

4.1.3 查找功能测试 29

4.1.4 修改功能测试 30

4.1.5 统计功能测试 30

4.2 边界测试 31

4.2.1 初始化无输入数据 31

4.2.2 删除头结点 31

4.2.3 删除后链表为空 32

4.3 出错测试 32

4.3.1 考生人数错误 33

4.3.2 操作码错误 33

4.3.3 插入位置不存在 34

4.3.4 删除考号不存在 34

4.3.5 查找考号不存在 35

4.3.6 修改考号不存在 35

V 总结 35

VI 参考文献 35

1. 分析

*1.1* 项目名称 ： 考试报名系统

1.2 项目背景

考试报名系统是一个学校不可缺少的部分，它对于学校的管理者和学生来说都至关重要，所以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。考试报名系统对于学校加强考试管理有极其重要的作用。随着学生数量和考试数量的日益庞大，如何管理如此庞大的数据显得极为复杂，传统的手工管理工作量大且容易出错。

随着计算机科学技术的不断成熟，使用计算机对考试报名系统进行管理，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。因此，开发一套考试报名系统具有十分重要的意义。

​

1.3 项目功能分析

作为一个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情况，建立考试报名系统并且可以予以显示。其次，考试报名系统还应该具有插入考生信息、删除考生信息、修改考生信息等功能，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。

II. 设计

2.1 数据结构设计

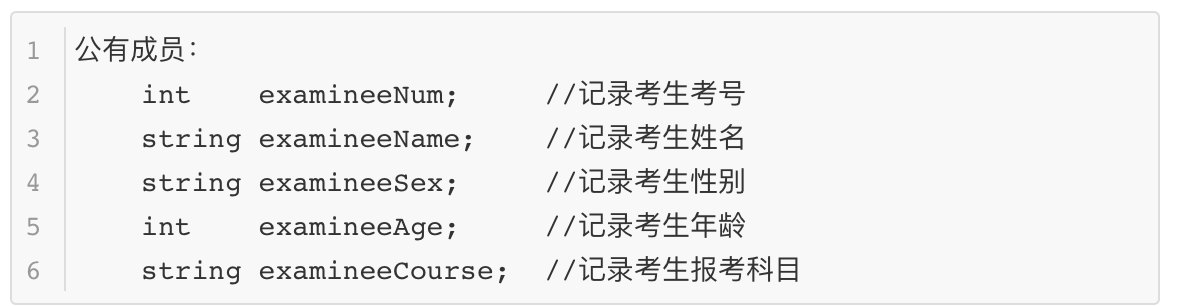
正如以上功能分析所述，该系统要求大量的增加、删除、修改操作，而链表进行增加、删除等操作十分简便，因此我们考虑使用链表数据结构。在链表类的实现中，我们要考虑插入元素，删除元素等操作时，元素位置的不同，我们实现的方法也应该相应有所变化，这在接下来的文档中会有所体现。​

2.2 数据结构类的设计

经典的链表一般包括两个抽象数据类型（ADT）——链表结点类（exam）与链表类（examinee），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（exam），链表结点类（examinee）的具体实现在系统类（examineeInformationSystem）中有所体现。

2.3 系统设计

2.3.1 考生数据类型类（exam)





2.3.2 考生链表类（examinee）



2.3.3 考试报名系统类（examineeInformationSystem)



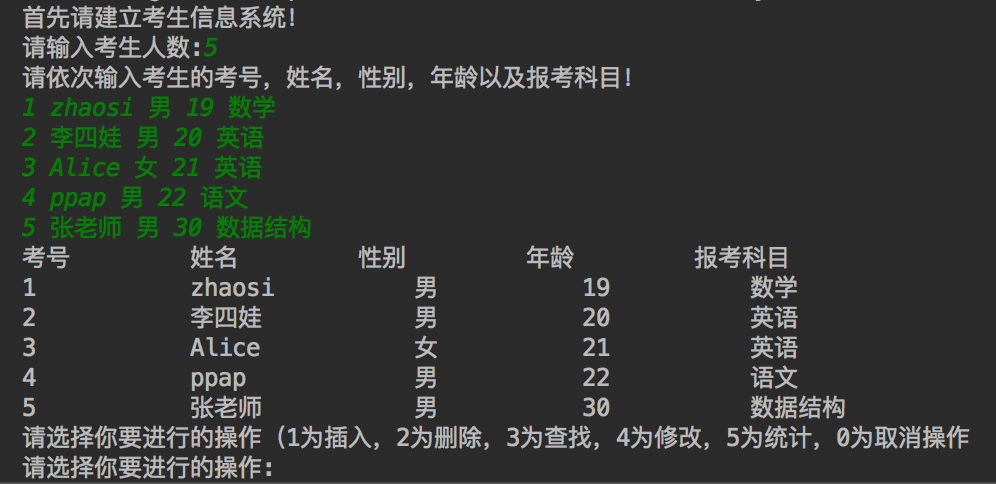


2.4 系统设计

系统首先调用Init()函数实现对屏幕的初始化，完成对链表的创建和输入数据工作，然后根据用户所输入的操作码（choice）执行考试报名系统对应的成员函数,在系统构建完毕后，要将创建完毕的系统输出显示，具体实现如下：



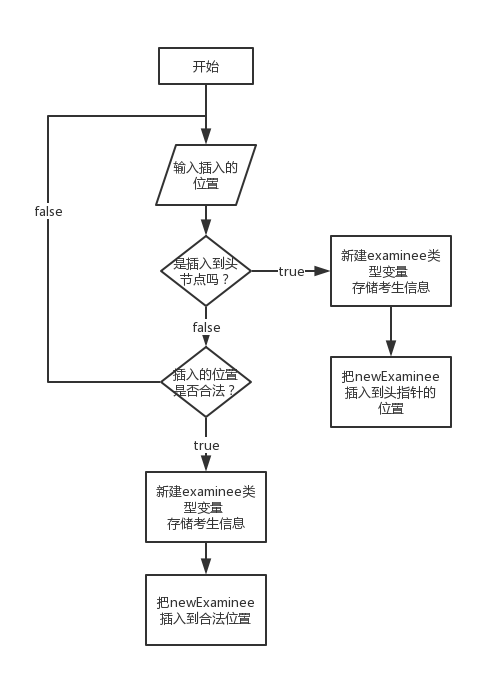
输入功能截屏示例：



III 实现

3.1 插入功能的实现

3.1.1 插入功能流程图



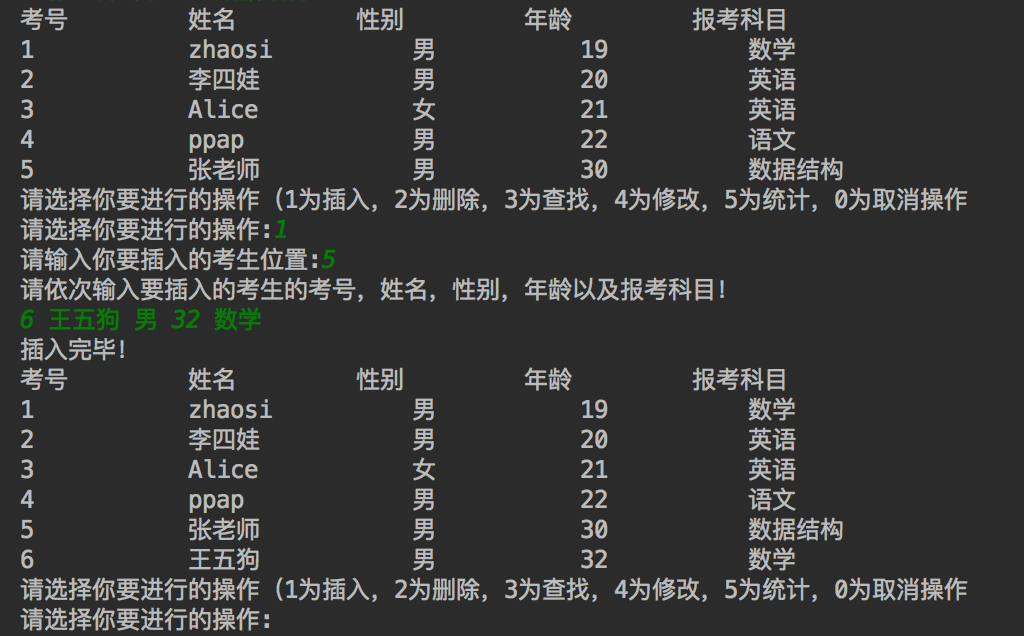
3.1.2 思路分析

由于采用的是单链表结构，我们在插入元素的时候要从头节点开始，不断去寻找他的下一个元素，直到找到该插入的位置。

3.1.3 具体实现如下：

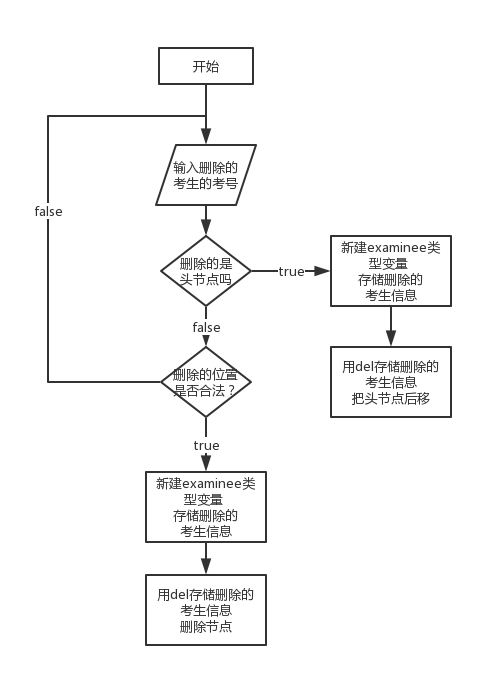


3.1.4 插入功能截屏示例



3.2 删除功能的实现

3.2.1 删除功能流程图



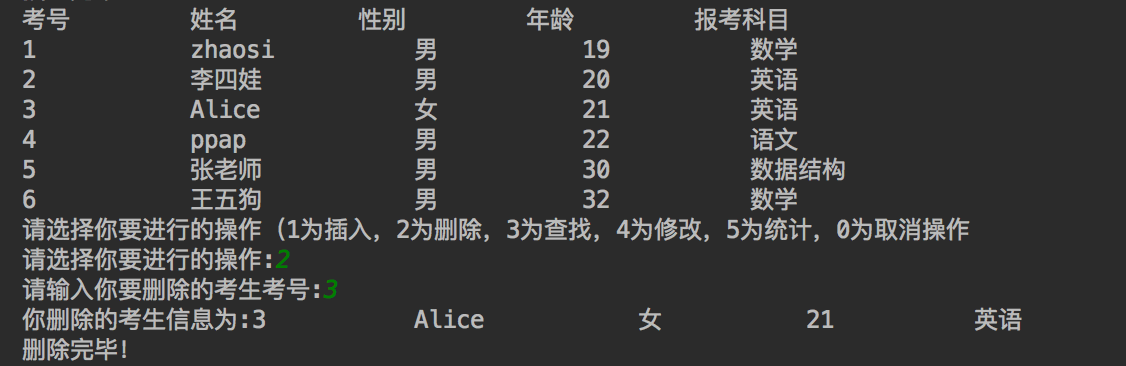
3.2.2 思路分析

由于采用的是单链表结构，我们在删除元素的时候也要从头节点开始，不断去寻找他的下一个元素，直到找到删除的位置进行操作。

3.2.3 具体实现如下：

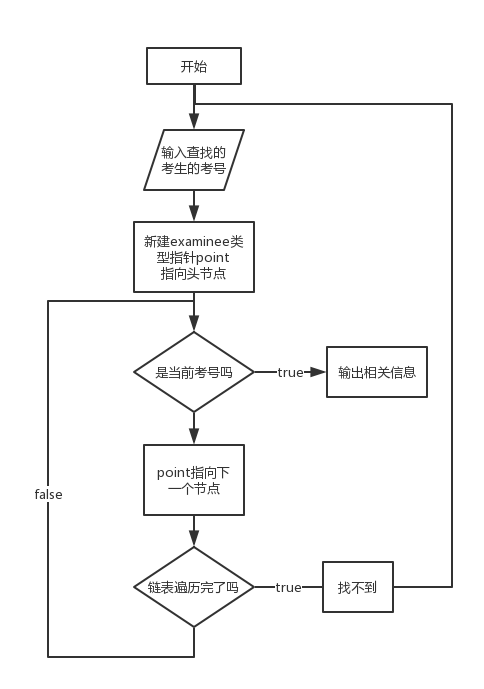


3.2.4 删除功能截屏示例



3.3 搜索功能的实现

3.3.1 搜索功能流程图



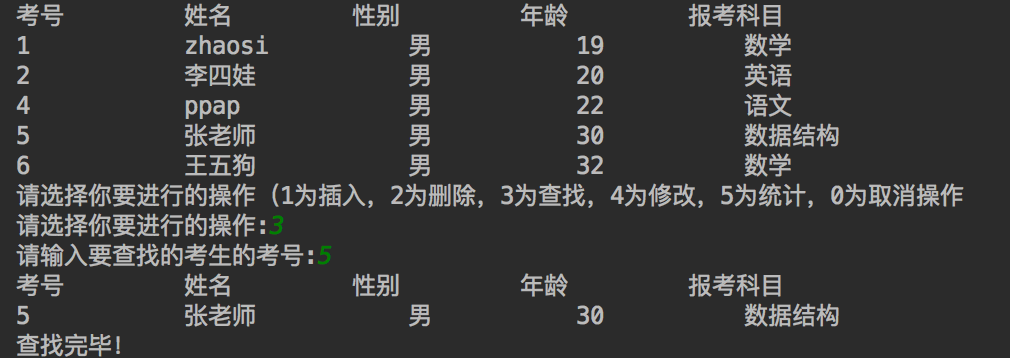
3.3.2 思路分析

由于采用的是单链表结构，我们在搜索元素的时候也要从头节点开始，不断去寻找他的下一个元素，直到找到所找的考号进行操作。

3.3.3 具体实现如下：

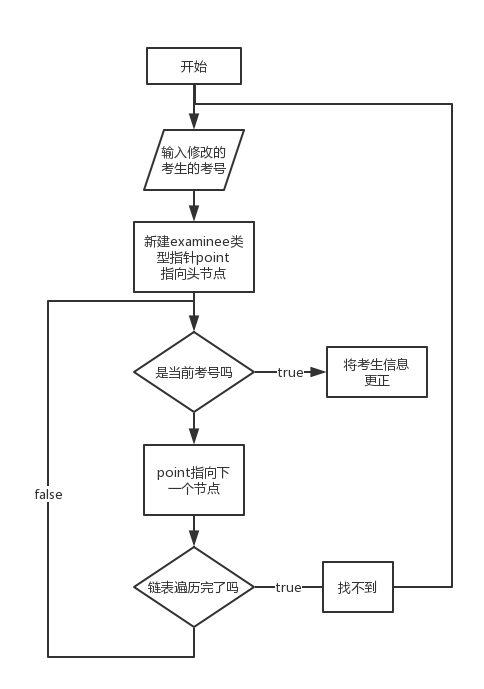


3.3.4 搜索功能截屏示例



3.4 修改功能的实现

3.4.1 修改功能流程图



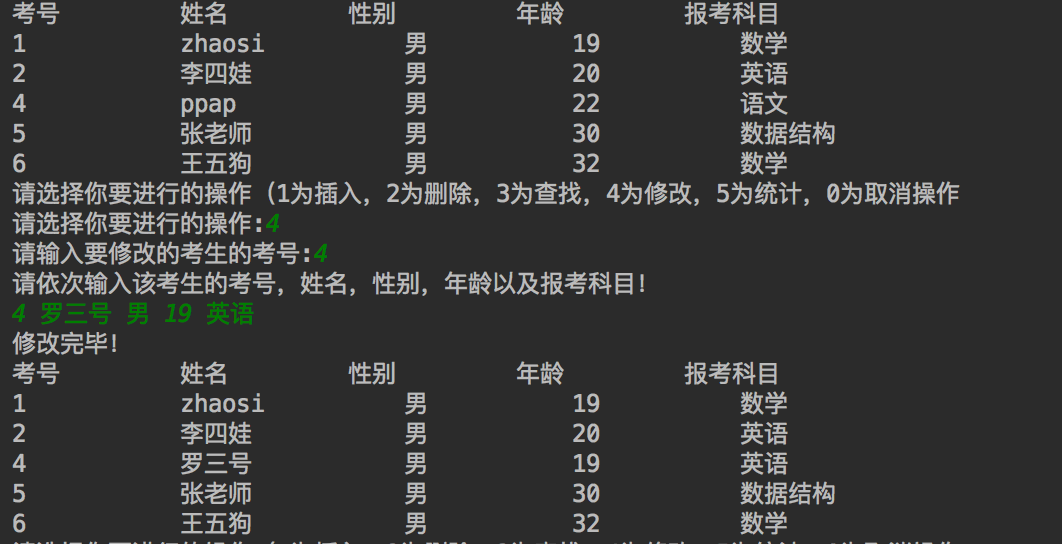
3.4.2 思路分析

由于采用的是单链表结构，我们在搜索元素的时候也要从头节点开始，不断去寻找他的下一个元素，直到找到所找的考号进行操作。

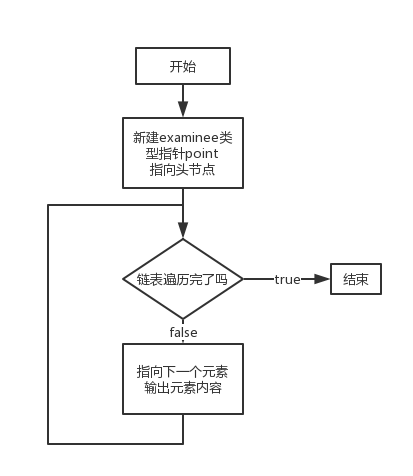
3.4.3 具体实现如下：



3.4.4 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

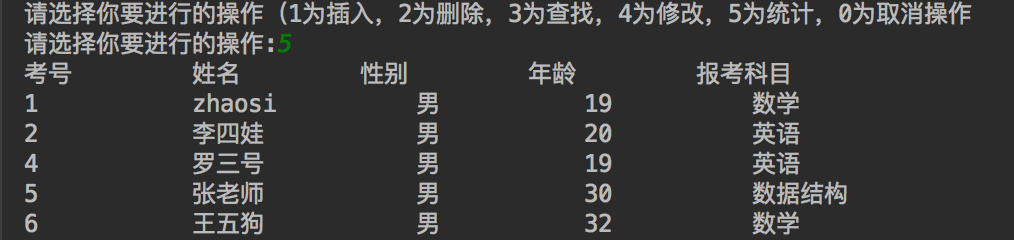
3.5.2 思路分析

由于采用的是单链表结构，我们在搜索元素的时候也要从头节点开始，不断去寻找他的下一个元素输出，直到所有的节点都已经被遍历完毕。

3.5.3 具体实现如下：

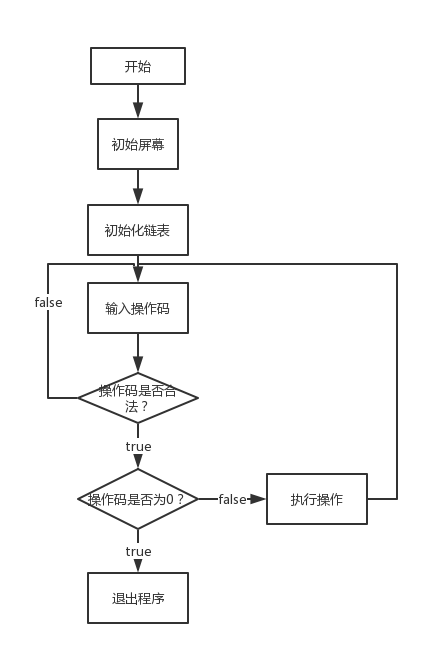


3.5.4 统计功能截屏示例



3.6 总体系统的实现

3.6.1 总体系统流程图



3.6.2 思路分析

用switch……case……default语句实现。

3.6.3 具体实现如下：



3.6.4 总体系统截屏示例



IV 测试

4.1 功能测试

4.1.0 建立系统测试

测试用例：

1 stu1 男 20 软件开发师

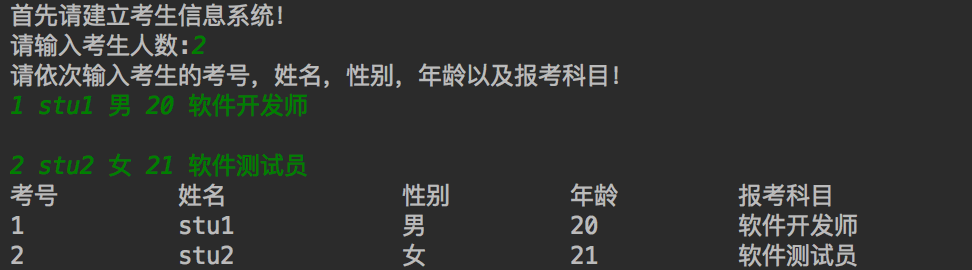
2 stu2 女 21 软件测试员

预期结果：

1 stu1 男 20 软件开发师

2 stu2 女 21 软件测试员

实验结果：



4.1.1 插入功能测试

测试用例：

插入位置：2

3 stu3 男 22 网络工程师

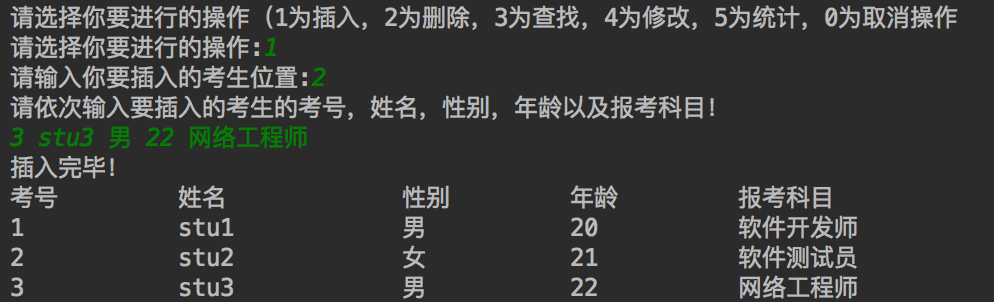
预期结果：

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



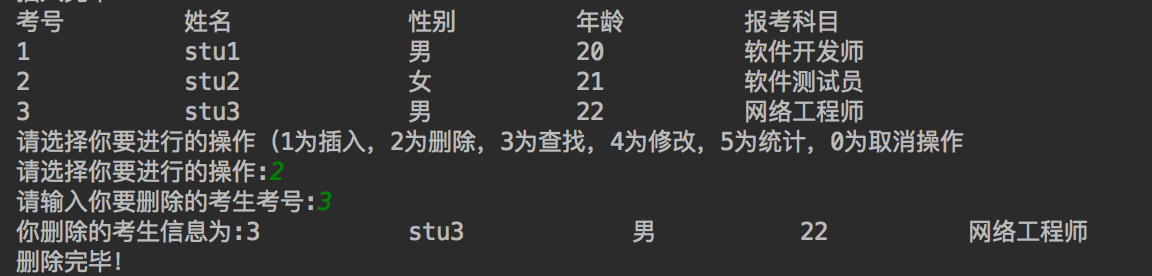
4.1.2 删除功能测试

测试用例：删除考号为3的考生

预期结果：

删除的考生信息为 3 stu3 男 22 网络工程师

实验结果：



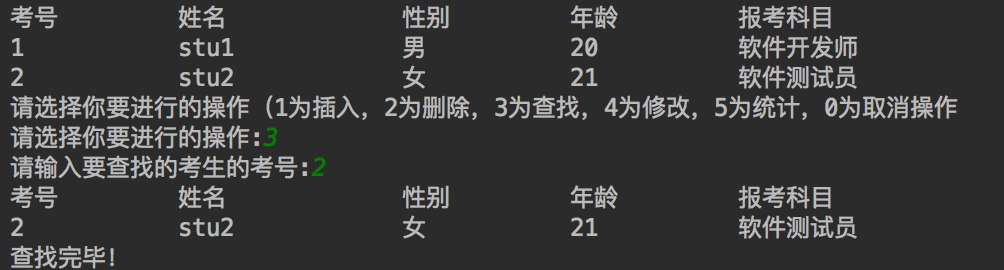
4.1.3 查找功能测试

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：



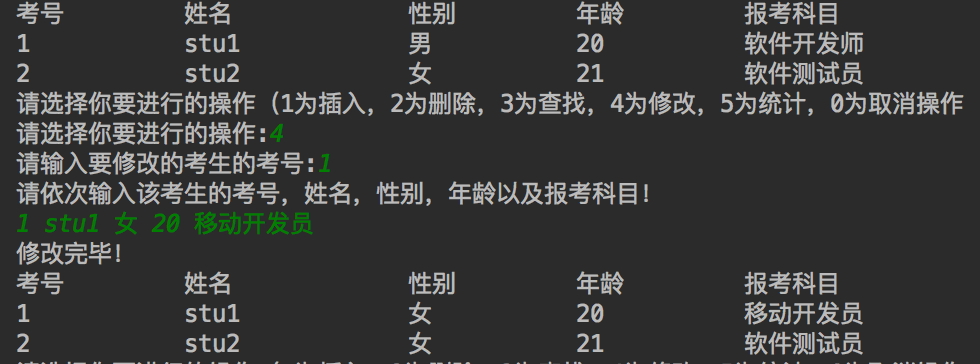
4.1.4 修改功能测试

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

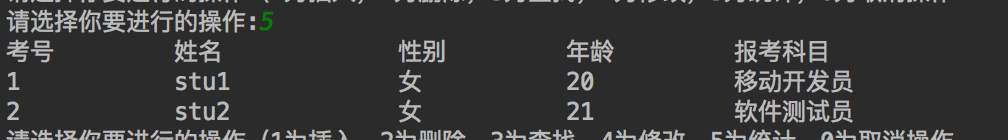
测试用例：统计当前数据

预期结果：

1 stu1 女 20 移动开发员

2 stu2 女 21 软件测试员

实验结果：



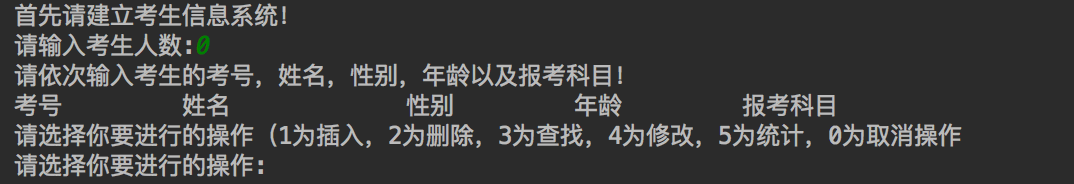
4.2 边界测试

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：程序运行正常不崩溃。

实验结果：

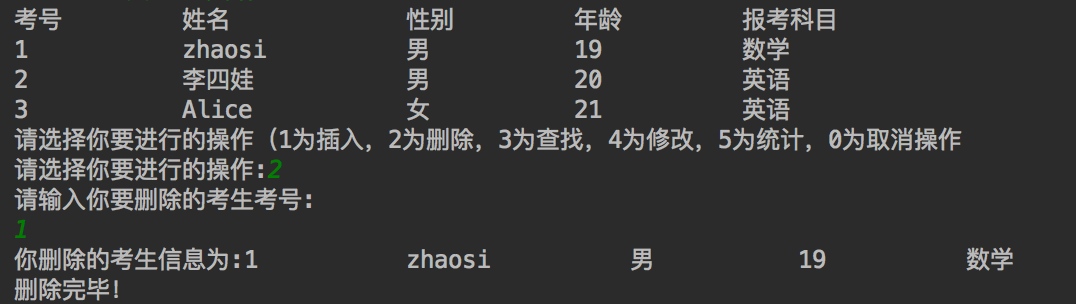


4.2.2 删除头结点

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：

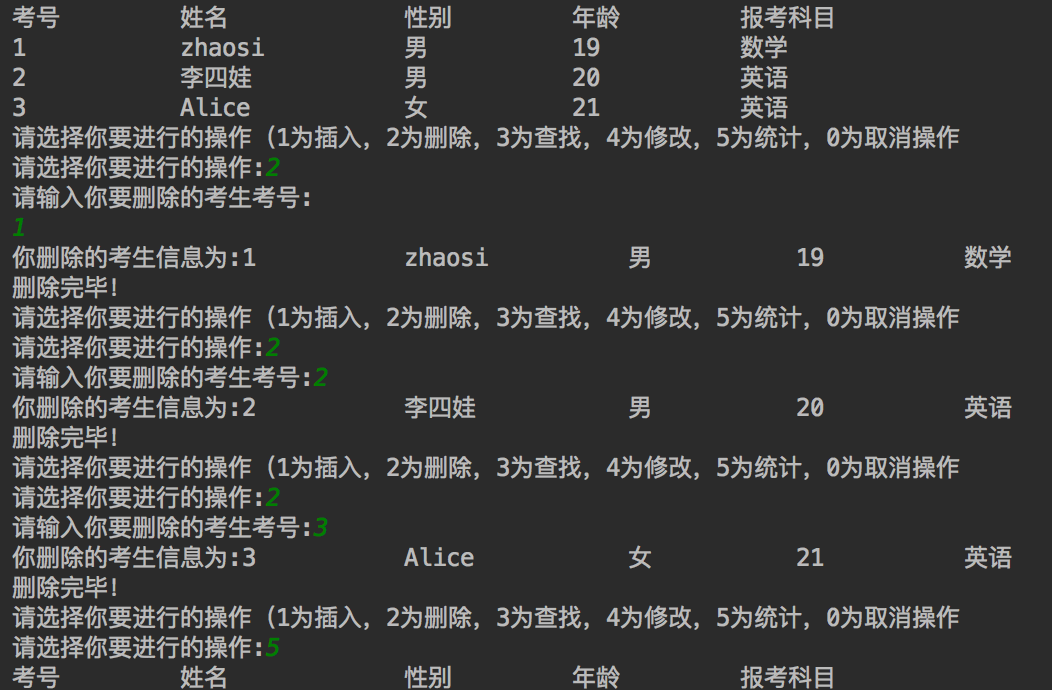


4.2.3 删除后链表为空

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：



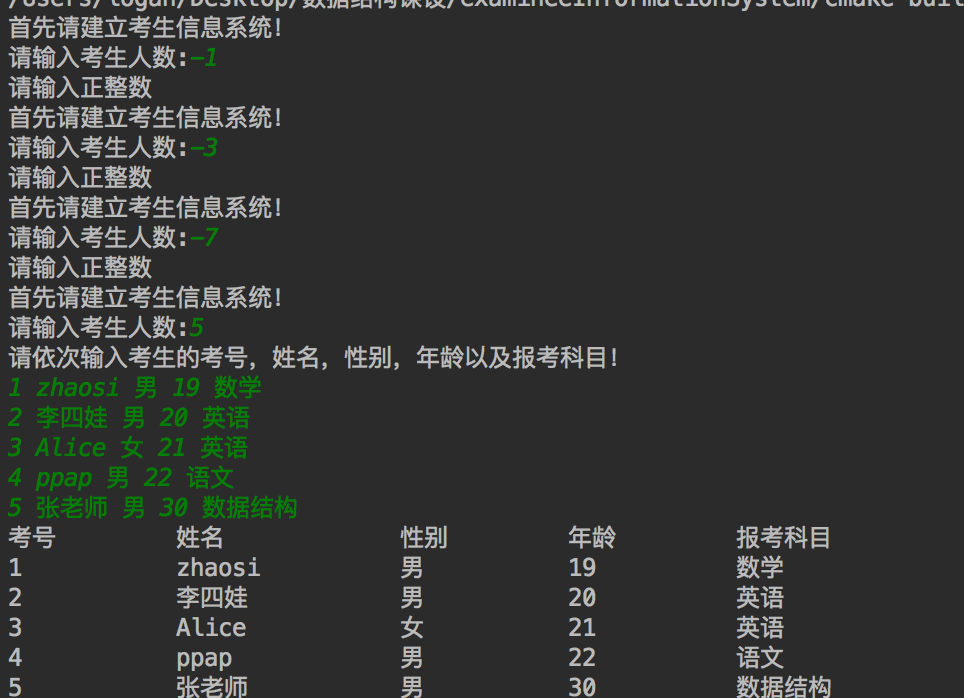
4.3 出错测试

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：

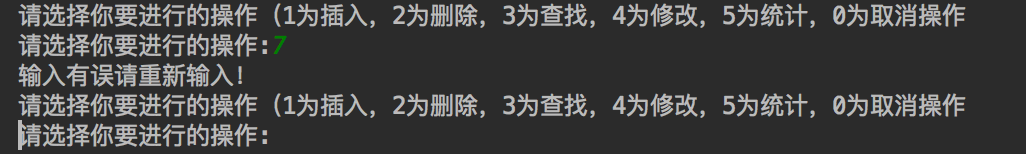


4.3.2 操作码错误

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：

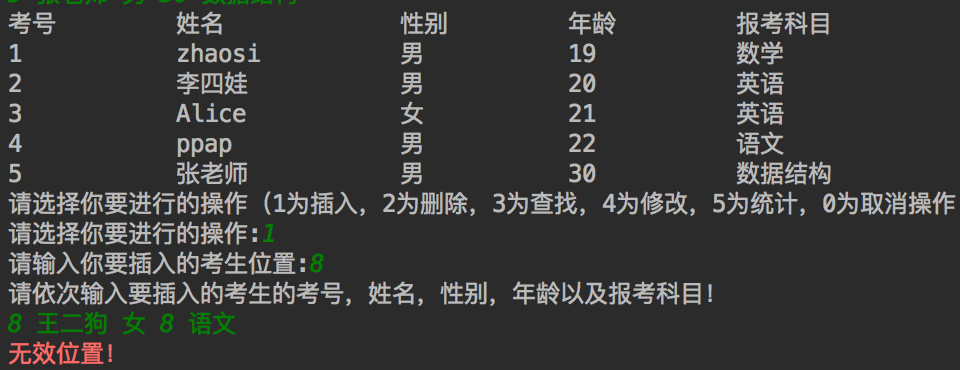


4.3.3 插入位置不存在

测试用例：链表里有5条记录，向链表的第8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：

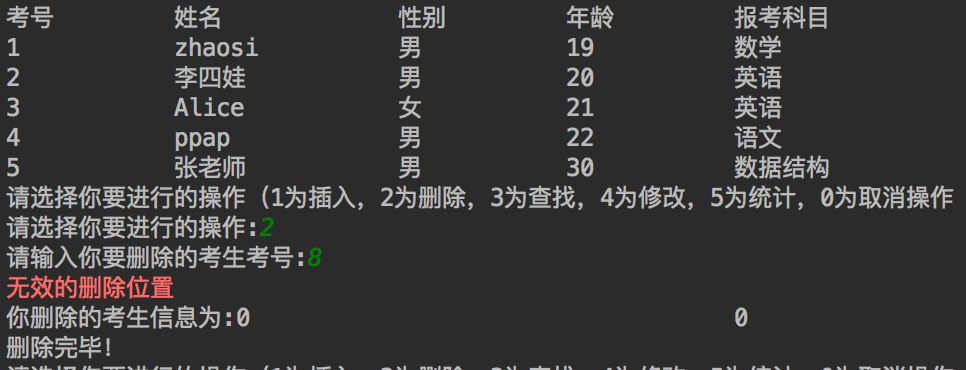


4.3.4 删除考号不存在

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：

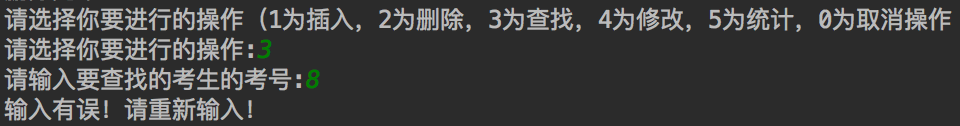


4.3.5 查找考号不存在

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：

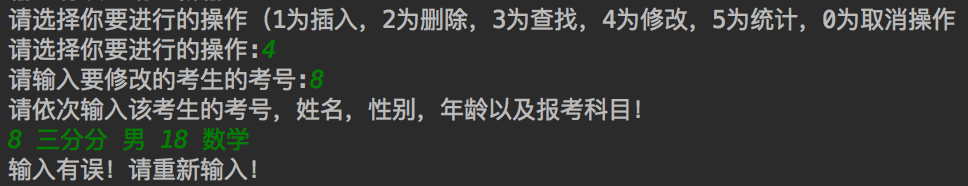


4.3.6 修改考号不存在

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：



V 总结

本实验让我进一步理解了链表类的使用方法。

VI 参考文献

1.数据结构课本