
山东大学（威海）
SHANDONG UNIVERSITY , WEIHAI

# 大作业报告（设计）

**设计(论文)题目** 学生信息管理系统的综合设计实现

# 摘　　要

在信息技术不断推陈出新的背景下,针对传统人工管理学生信息方式效率低,提出一种基于 C++ 语言的学生信息管理系统。

本设计主要通过使用 C++ 程序设计语言，按照大作业的相关要求在实现增删改查功能时，数据结构均采用链表实现，同时程序均采用文件将信息储存，也设计了具有分角色使用或管理系统功能。

设计程序时首先完成了 win32 位窗口程序的设计，在 win32 位窗口程序的代码基础之上，利用 QT5 框架完成了图形化 (GUI) 的改造，使之更加符合用户的使用习惯，并且支持在不同系统环境下的使用。本程序设计是在单机情况下的较为完善的学生信息管理系统。



**关　键　词:** 学生信息管理系统，链表，C++，图形化程序设计，QT5 开发

# ABSTRACT

Under the background of continuos innovation of information technology, aiming at the low efficiency of manual management of student information, a student information management system based on C++ programming language is proposed.

This design uses the C++ programming language, according to the relevant requirements of major works. When implementing the addition, deletion, modification and inspection functions, the data structure is implemented by a linked list. At the same time, the program uses files to store the information. A role-based use or management system is also designed.

When designing the program, the design of win32-bit windows program was first completed. On the basis of the code of the win32-bit windows program, the QT5 framework was used to complete the graphical (GUI) transformation to make it more in line with the user's habits and support different Use in system environment. This program design is a relatively complete student information management system under the stand-alone situation.

**Key words:** student information management system,linked list,C++,GUI,QT5

# 目　　录

# 一、项目分工情况

根据大作业的相关要求，项目的分工合作情况如下表1.1所示：

表 1.1: 项目分工合作情况表

## 二、系统需求分析

对于常见的学生信息系统而言，我们需要存储学生的许多相关信息，包括学生的学号、姓名、年龄、性别、生日、地址、电话、成绩等，这些数据要求能够从规范的存储学生信息的文件中读入，也要能够在程序运行时及时保存修改删除的学生信息到文件之中，并且根据大作业的相关要求，我们将这些数据在程序中以链表的形式存储，并通过链表操作。

同时进入程序后用户需要以菜单的形式方便用户进行相关的操作，故需要设计菜单界面，用户可以根据不同的指令来进行操作。

对于学生信息管理系统，可能需要进行的常见操作有如下几种：

- 从文件读入学生信息
- 输出学生的相关信息
- 在程序中增加学生学生信息
- 删除单个学生信息
- 修改已存储的某个学生的信息
- 查找单个学生的信息
- 按学号或成绩对学生信息进行排序

因此，我们需要设计相关的功能，而这些功能的实现均需根据大作业的相关要求，对增删改查功能必须使用链表进行操作。故我们在设计实现相关功能操作的时候，都采用链表来进行相关功能函数的实现。

同时由于数据量可能较大，采用链表实现，可以在程序运行时根据实际的数据量来分配内存，并在一定程度上提高了程序的运行效率，可以面对较大规模数据，从而实现可能出现的较大规模信息数据的处理的需求。

对于使用程序的不同用户，我们需要进行验证，对经过验证确认为管理员的，我们可以让管理员进行上述所有操作，而非管理员无需认证，但仅能够查找学生信息和按学号或成绩对学生信息进行排序。因此在进入程序前我们需设计登录界面，来确认是否为管理员。从而实现不同用户角色对学生信息的操作，保障了所存储信息的安全。

在管理员使用程序输入学生信息或者修改学生信息时，可能会输入部分比较明显的错误信息，如学生的成绩超过 100 等，这时需要提醒管理员进行修改，

同时增加或修改学生信息时可能会漏填相关信息时，也需要提醒管理员补充好相关信息。

而由于我们的程序面向实际应用，所以需要对我们的程序界面进行美化，使用户在使用程序时能够感到比较舒服。故我们首先设计 win32 位程序时需要对窗口的界面和文字进行调整以更美观，而在 GUI 程序设计时也需要对窗口进行美化。

## 三、系统概要分析

对于我们设计的学生信息管理系统，主要采用文件和程序交互设计，用户通过程序操作文件内的存储信息。以下是对于我们设计的两种不同的程序的系统概要分析。

### （一）对于 win32 位程序

对于我们设计的 win32 位程序的学生信息管理系统，管理员的相关信息存储于 *admin.txt*中, 学生信息的相关数据存储于 *score.dat*中，为便于分工合作采用面向对象的思维进行程序设计。

#### 1.程序运行架构

程序运行架构由下图3.1所示

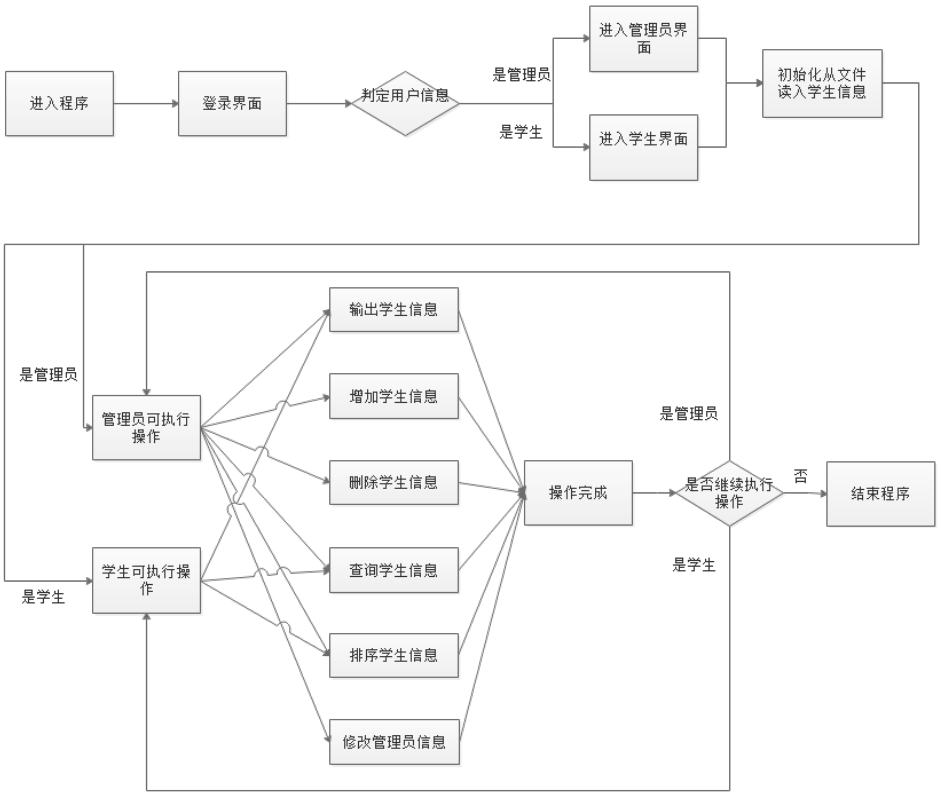


图 3.1: win32 位程序整体架构图

2.代码文件组成

整个程序的代码分为以下三个文件

- *Student.hpp*
- *Node.hpp*
- *Main.cpp*

在 *Student.hpp*中，包含类 Student，类中的 private 利用不同的基本数据类型保存着每一项学生信息，public 中主要是设置信息和获取信息的相关函数内容以及构造函数。以下表3.1是 private 中各个变量的基本信息：

表 3.1: Student 类中 private 的变量

| 变量名称 | 变量类型 | 变量含义 |
|---|---|---|
| stu_number | long long | 学号 |
| stu_name | string | 姓名 |
| stu_age | string | 年龄 |
| stu_gender | string | 性别 |
| stu_bir | int | 生日 |
| stu_address | string | 地址 |
| stu_tel | long long | 电话 |
| stu_email | string | 邮件 |
| stu_score | float | 成绩 |
| sort | int | 排名 |

在 *Node.hpp*中，包含类 Node，类中的 private 存储数据域、指针域以及头节点。在类中的 public 中主要为各个功能函数的声明以及函数具体的功能实现的代码。

在 *Main.cpp*中，主要是实现登录的界面以及不同用户的界面菜单的设计和函数的调用，主要功能是实现用户交互功能，同时也包含了修改管理员信息的功能。

3.函数列表

在 *Student.hpp*中，主要是面向对象中常见的函数以及传递数据的相关函数，在此不表。

在 *Node.hpp*中，具体的函数和其功能如下表3.2：

表 3.2: Node.hpp 中所有的函数

| 函数返回类型 | 函数名 | 函数功能 |
|---|---|---|
| void | InputStudent | 输入学生信息 |
| void | OutputStudent | 输出学生信息 |
| void | FileOutputStudent | 输出学生信息到文件 |
| void | AddStudent | 增加学生信息 |
| void | DeleteStudent | 删除学生信息 |
| void | ChangeStudent | 修改学生信息 |
| void | SearchStudent | 查找学生信息 |
| void | SortScore | 将成绩按照从大到小排序 |
| void | SortNum | 将学号按照从小到大排序 |
| void | SetScore | 设置排名 |

注：以上函数均没有形参

在 *Main.cpp*中，具体的函数及其功能如下表3.3：

表 3.3: Main.cpp 中所有的函数

| 函数返回类型 | 函数名 | 函数功能 |
|---|---|---|
| void | menu_admin | 管理员菜单界面 |
| void | menu_stu | 学生用户菜单界面 |
| void | menu_login | 用户登录界面 |
| void | modify_admin | 管理员信息修改界面 |
| int | main | 主函数 |

注：以上函数均没有形参

## （二）对于 GUI 程序

对于我们设计的基于 QT5 框架下的 GUI 程序，我们将学生的相关信息存储在 *stuinfo.txt*文件中，由于没有使用数据库相关技术，在登录界面上只能使用提前设置好的账号密码验证管理员身份 (**在本程序设计中, 管理员账号为 *admin*, 密码为 *123456***)，对于学生用户，在登录界面可以直接选择学生界面，同样可以查找和排序 (查看) 的操作，从而实现不同用户的对程序的使用。**注意：本程序在 QT 5.12.8 环境下设计,Qt Creator 版本为 4.11.2 (Community)**[1]

[1]安装包来源https://mirrors.tuna.tsinghua.edu.cn/qt/official_releases/qt/5.12/5.12.8/

1.程序运行架构

程序运行架构由下图3.2所示

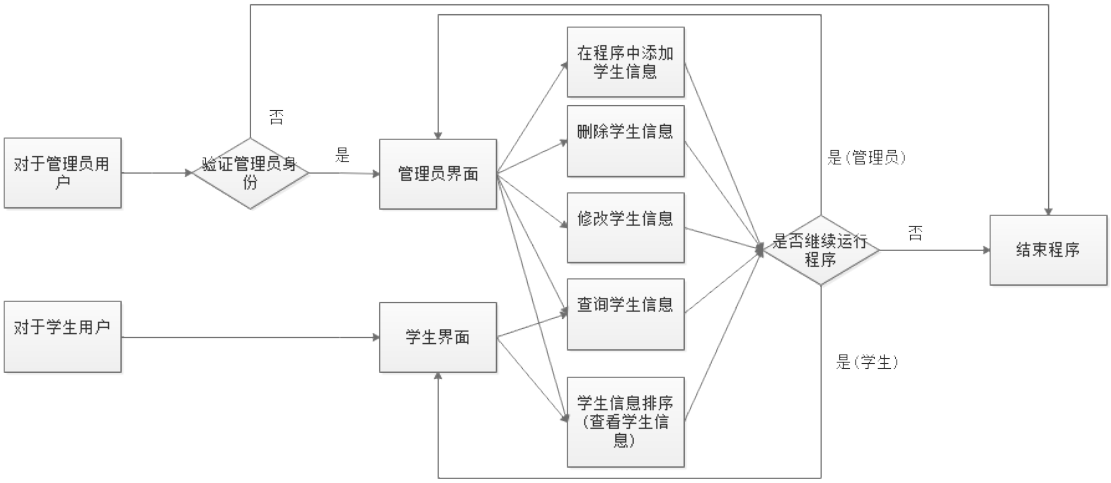


图 3.2: GUI 程序整体架构图

2.代码文件组成

该程序代码为标准的 QT5 Widgets Application 的工程代码，包含以下几部分，如下表3.4所示：

表 3.4: GUI 程序的代码组成

| 文件名 | 包含后缀 | 实现的功能 |
|---|---|---|
| addstudentwidget | .cpp .h .ui | 增加学生信息界面 |
| adminmenuwidget | .cpp .h .ui | 管理员模式菜单界面 |
| adminwidget | .cpp .h .ui | 管理员模式窗口界面 |
| delwidget | .cpp .h .ui | 删除学生信息界面 |
| findwidget | .cpp .h .ui | 查找学生信息界面 |
| mainwidget | .cpp .h .ui | 登录**主窗口**程序界面 |
| sortwidget | .cpp .h .ui | 排序（输出）学生信息菜单界面 |
| stumenuwidget | .cpp .h .ui | 学生模式菜单界面 |
| stuwidget | .cpp .h .ui | 学生模式窗口界面 |
| node | .cpp .h | 以链表形式存储信息和功能实现函数 |
| stuinfo | .cpp .h | 以类表示学生信息 |
| main | .cpp | 主函数 |
| stu_info | .pro | QT5 工程文件 |

图标文件和 qt5 资源文件 (.qrc) 在此不表。

在 stuinfo 类的 private 中，利用不同的数据类型保存每一项学生信息，以下表3.5为 private 中各个变量的基本信息:

表 3.5: stuinfo 类中 private 的变量

| 变量名称 | 变量类型 | 变量含义 |
|---|---|---|
| stu_number | long long | 学号 |
| stu_name | QString | 姓名 |
| stu_age | QString | 年龄 |
| stu_gender | QString | 性别 |
| stu_bir | int | 生日 |
| stu_address | QString | 地址 |
| stu_tel | long long | 电话 |
| stu_score | double | 成绩 |

3.函数列表

对 *addstudentwidget.cpp*，包含构造和析构函数，以及返回主菜单按钮的槽函数 on_returnButton_clicked、增加学生信息的槽函数 on_addButton_clicked。

对 *adminwidget.cpp*，包含构造和析构函数。

对 *adminmenuwidget.cpp*，包含构造和析构函数，以及各个按钮的槽函数。

对 *delwidget.cpp*，包含构造和析构函数，以及返回主菜单按钮的槽函数 on_returnButton_clicked、删除学生信息的槽函数 on_deletepushButton_clicked。

对 *findwidget.cpp*，包含构造和析构函数，以及返回主菜单按钮的槽函数 on_returnButton_clicked、查找学生信息的槽函数 on_pushButton_clicked。

对 *mainwidget.cpp*，包含构造和析构函数。

对 *menuwidget.cpp*，包含构造和析构函数，以及登录界面各个按钮的槽函数。

对 *modifywidget.cpp*，包含构造和析构函数，以及返回主菜单按钮的槽函数 on_returnButton_clicked、查询修改前学生信息的槽函数 on_searchButton_clicked，修改学生信息的槽函数 on_modifypushButton_clicked。

对 *sortwidget.cpp*，包含构造和析构函数，以及返回主菜单按钮的槽函数 on_returnButton_clicked、获取学生信息的函数 getStuInfo，std::sort 所需的排序函数 cmp_number 和 cmp_score，排序学生信息的槽函数 on_modifypushButton_clicked。

对 *stuwidget.cpp*，包含构造和析构函数。

对 *stumenuwidget.cpp*，包含构造和析构函数，以及各个按钮的槽函数。

对 *stuinfo.cpp*, 包含构造和析构函数，以及传递信息的相关函数，在此不表。

对 *Node.cpp*, 包含构造函数，从文件获取信息函数 InputStudent，删除学生信息函数 DeleteStudent，输出信息到文件函数 OutputStudent, 修改学生信息函数 ChangeStudent，查找学生信息函数 SearchStudent, 相关函数的声明如下。

- void InputStudent();
- void DeleteStudent(long long number);
- void OutputStudent();
- void ChangeStudent(QString name,long long number,QString age,QString gender,long long tel,QString bir,QString address,double score);
- bool SearchStudent(QString &name,long long number,QString &age,QString &gender,long long &tel,QString &bir,QString &address,double &score);

# 四、win32 位界面系统代码设计

## （一）增加学生信息函数

这是一个无参函数，实现新增一位学生信息的功能，新的学生信息节点默认添加到链表尾部。

算法：定义节点指针 p，指向第一个学生信息节点 p=pHead->pNext，通过 while 循环将其指向尾节点，读入学生信息并将其写入新建节点 pt 中，p->pNext 指向新节点 pt，pt->pNext 为 NULL，添加完成。

流程图如下图4.1：

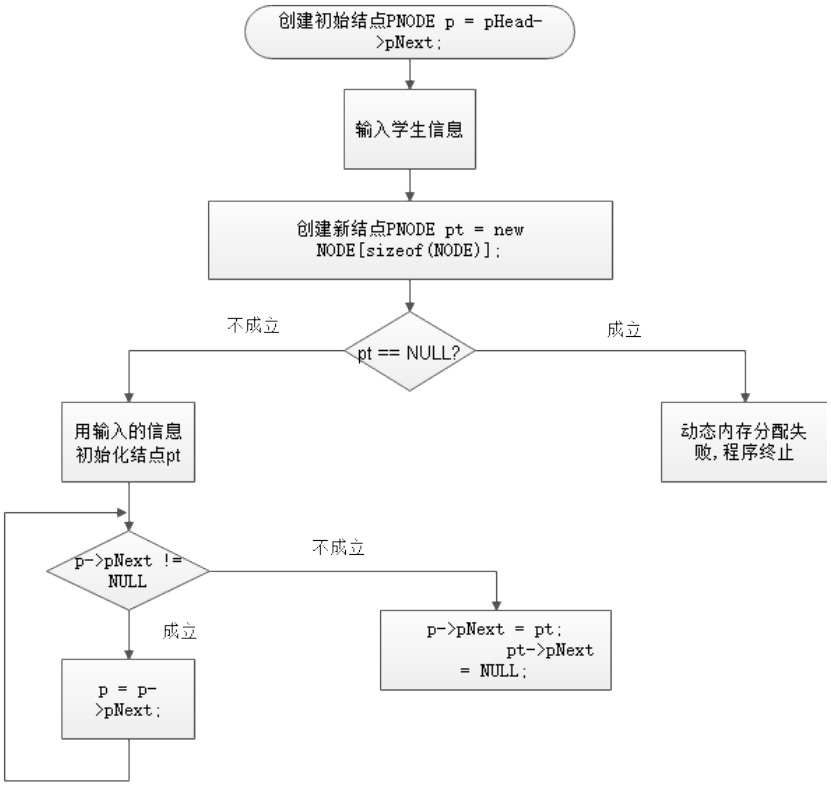


图 4.1: 增加学生信息函数流程图

## （二）输入学生信息函数

Void InputStudent() 这是一个无参函数，实现将文件中存储的学生信息读入到程序中。

算法：先声明一个头节点 pHead，并创建一个指向头节点的指针 pTail，每

输入一个学生的信息就声明一个新节点 pNew 来存储信息，把 pNew 挂到老节点后面，同时移动 pTail 到 pNew 上并使 pTail->pNext = NULL。

流程图如下图4.2：

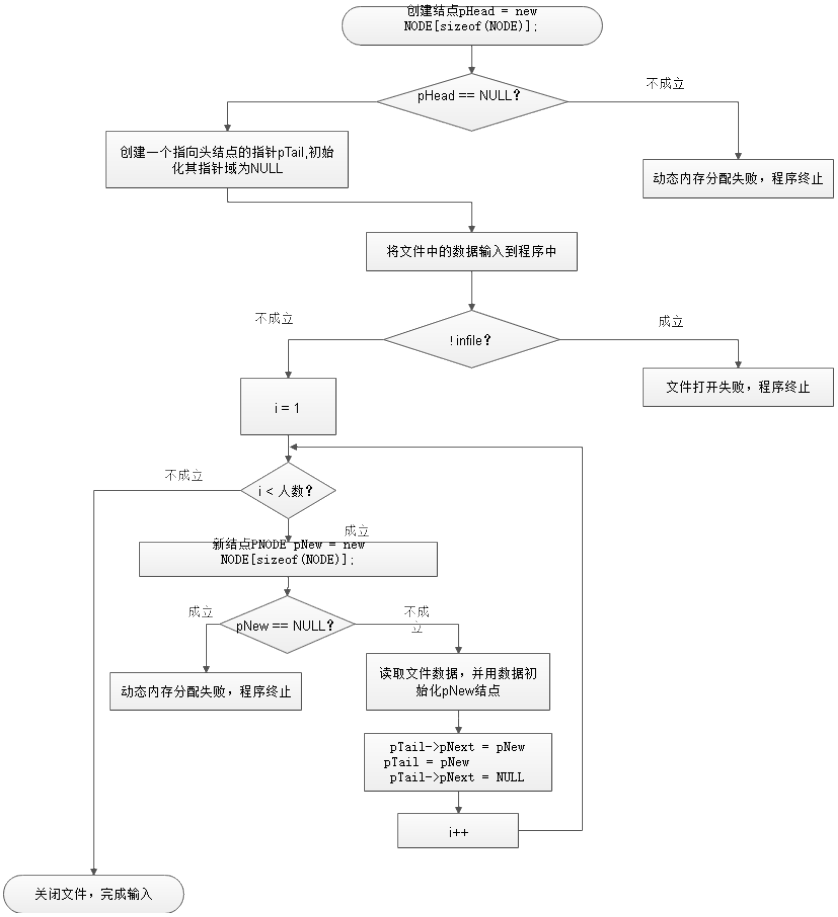


图 4.2: 输入学生信息函数流程图

## （三）输出学生信息函数

这是一个无参函数，负责对链表中全部学生信息的输出。

算法：先将 p 节点的指针指向第一个节点，将 p 节点（即第一个节点）的数据输出。然后再将 p 节点的指针指向下一个节点，即 p = p -> pNext，再次将 p 节点的数据输出。重复执行此步骤直到 p 指针指向 NULL 为止。

流程图如下图4.3：

## （四）删除学生信息函数

这是一个无参函数，可以实现用户输入要删除学生的学号，根据学号删除学生信息。

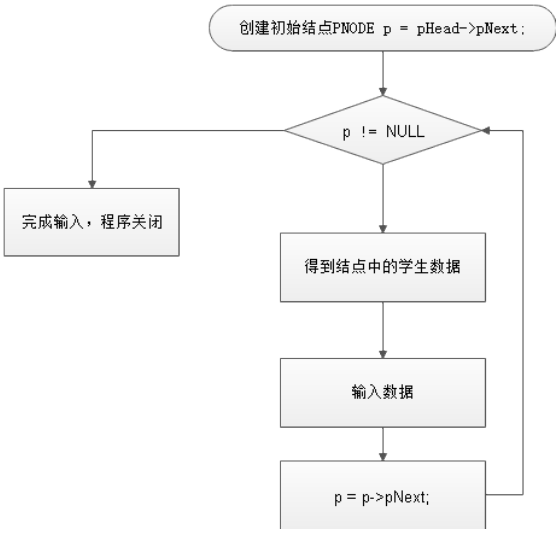


图 4.3: 输出学生信息函数流程图

算法：定义节点指针 be，bp。先将 be，bp 指向头结点，读入要删除学生的学号，通过 while 采用线性查找法往下查找，当找到该学号或 bp->pNext=NULL 时，跳出循环。判断是否找到该学生，若是则删除此节点（be 指向 bp 的上一个节点，通过 be->pNext = bp->pNext 完成删除该节点），输出“已成功删除该学生信息”，否则没有找到该学生，输出“查无此人”。

流程图如下图4.4：

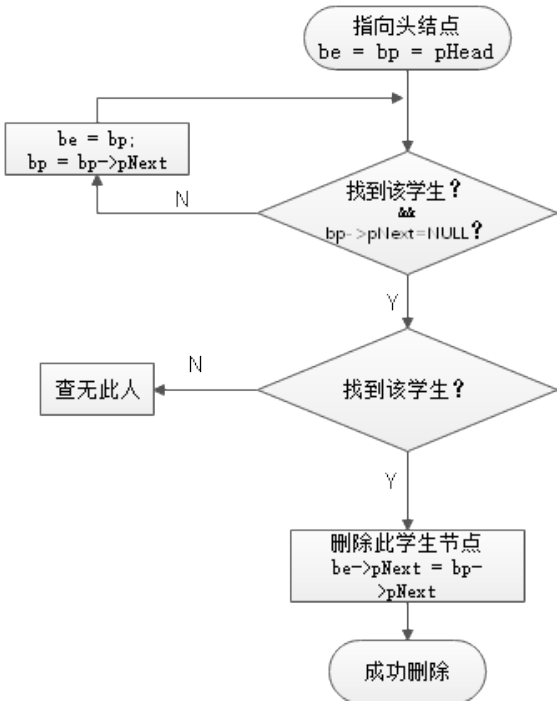


图 4.4: 删除学生信息函数流程图

## （五）修改学生信息函数

这是一个无参函数，可以实现用户通过姓名来查找学生，并根据用户输入的数字选项来确定修改具体的学生信息。

算法：通过 while 循环采用线性查找法往下查找，若找到该学生，输出该学生信息，且使 flag=1，跳出 while 循环。flag 值用于判断是否找到该学生，若找到则根据用户输入的数字通过 swithch() 实现读入新信息和各修改信息函数的调用，完成修改；若没有找到该学生（flag=0），提示用户未找到学生信息。

流程图如下图4.5：

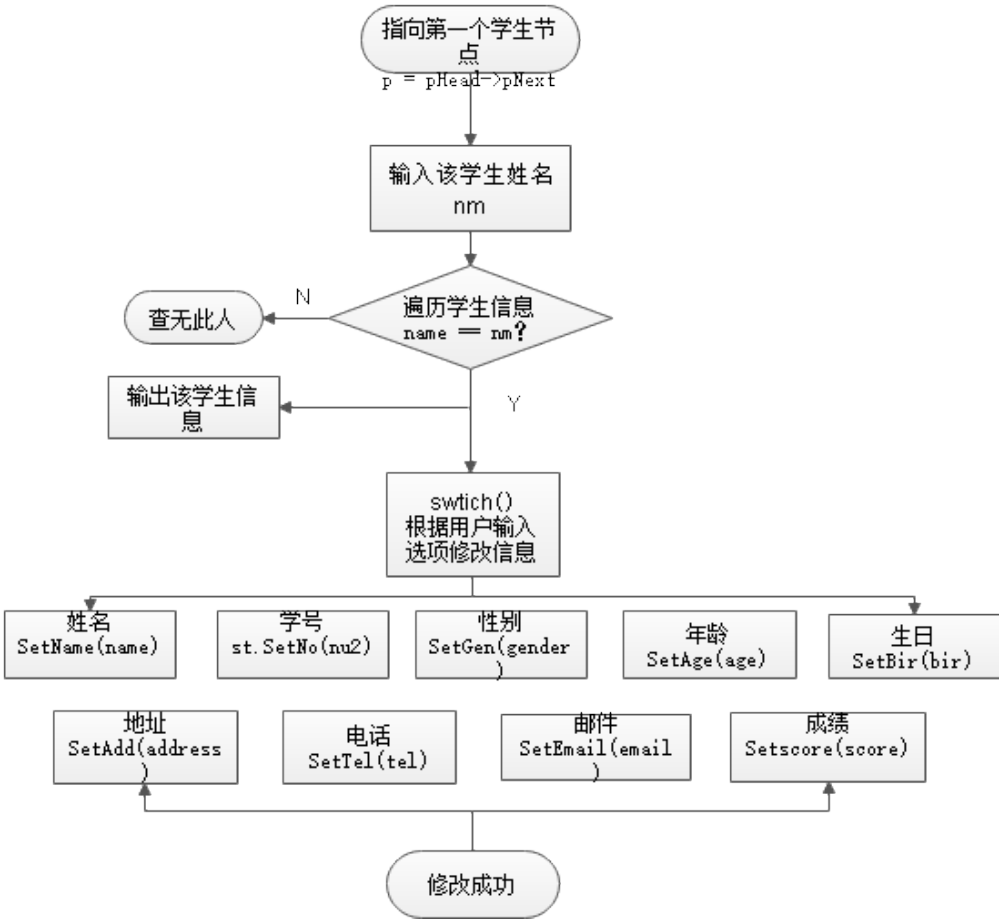


图 4.5: 修改学生信息函数流程图

## （六）查找学生信息函数

这是一个无参函数，可以实现通过姓名来查找学生信息并输出。

算法：先定义节点指针 p 指向第一个学生信息 p = pHead->pNext，通过 while 循环按线性查找法往下查找，找到后输出该学生信息并跳出循环；若 p->pNext=NULL，则说明未找到，则提醒用户未找到该学生信息。

流程图如下图4.6：

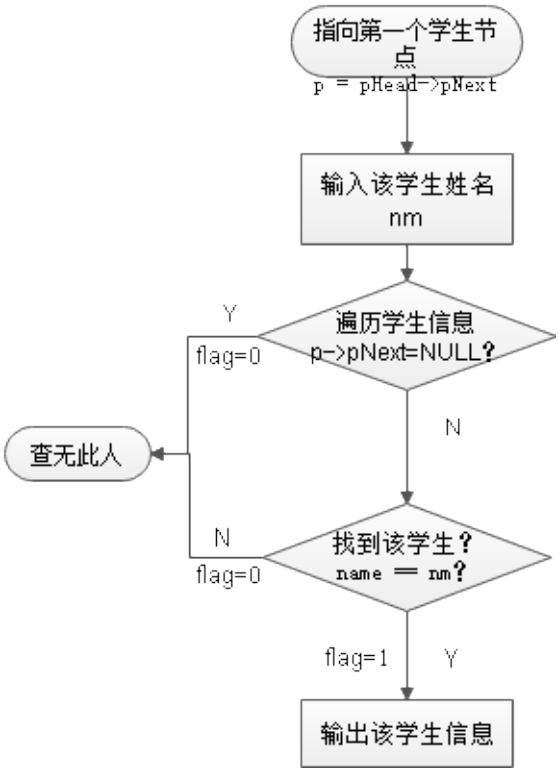


图 4.6: 查找学生信息函数流程图

## （七）将成绩按照从大到小排序函数

这是一个无参函数，实现按学生成绩从大到小进行排序。

算法：采用冒泡排序法，定义一个临时节点 temp 和两个指针 p，q，指向两个相邻学生。外层 for 循环控制排序趟数，内层 for 循环比较相邻两个学生成绩，若前一个学生的成绩小于后一个学生的成绩，则交换学生位置（temp.st = p->st; p->st = q->st;q->st = temp.st;），实现排序。

流程图如下图4.7：

## （八）将学号按照从小到大排序

这是一个无参函数，实现按照学生学号按照从小到大排序。

算法：采用冒泡排序法，定义一个临时节点 temp 和两个指针 p，q，指向两个相邻学生。外层 for 循环控制排序趟数，内层 for 循环控制每趟比较次数，并依次比较相邻两个学生的学号，若前一个学生的学号大于后一个学生的学号，则交换两个学生信息节点的位置（temp.st = p->st; p->st = q->st;q->st = temp.st;），实现排序。

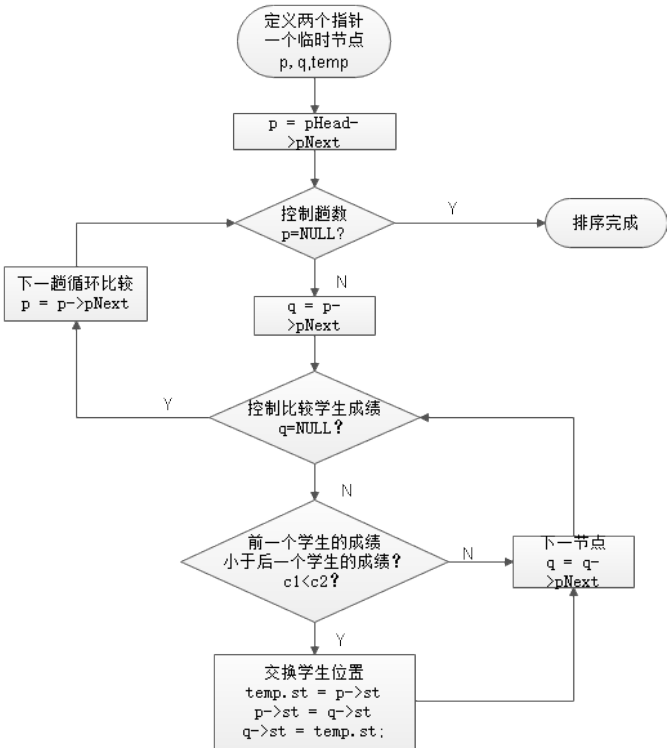


图 4.7: 将成绩按照从小到大排序函数流程图

流程图如下图4.8：

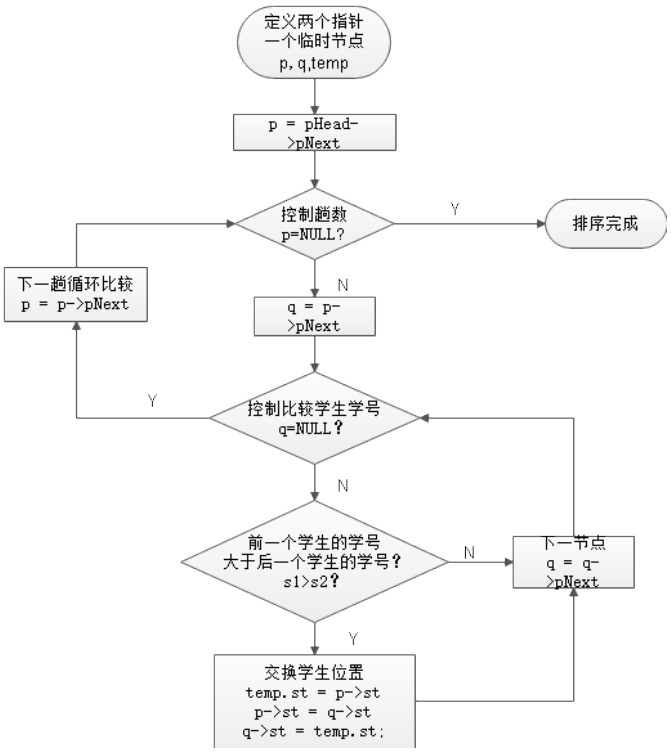


图 4.8: 将学号按照从小到大排序函数流程图

## （九）设置排名函数

这是一个无参函数，实现对学生排名的设置。

算法：定义节点指针 p 和序号 i，p=pHead->pNext，i=0。执行完其他排序函数后，通过 for 循环遍历每个节点（p=p->pNext，i++），为每个节点的学生排名赋值。

流程图如下图4.9：

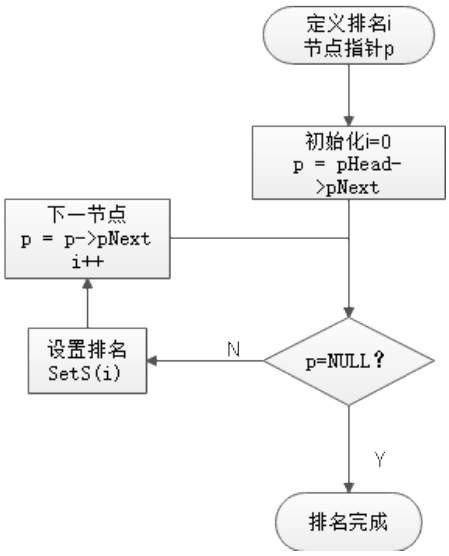


图 4.9: 设置排名函数流程图

## （十）登录菜单设计

这是一个无参函数，实现判断管理员和学生用户功能。

算法：先判断是否存在有管理员信息的文件，没有的话创建管理员信息，并创建管理员信息文件。若有管理员信息文件，我们选择登录时若选择管理员界面，需要验证管理员信息，不通过则返回。若选择学生界面则进入学生界面。输入其他信息提示输入不合法。

流程图如下图4.10：

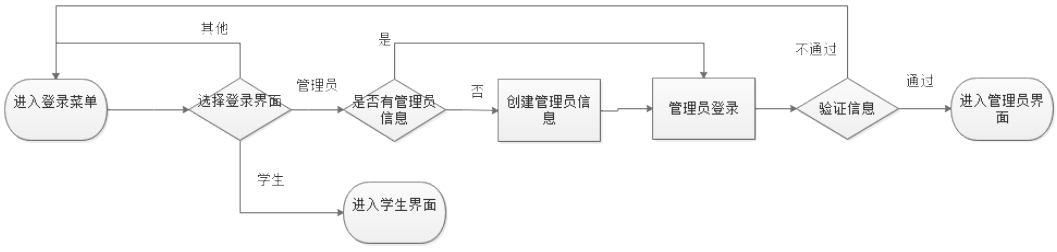


图 4.10: 登录菜单设计流程图

# 五、图形化界面系统代码设计

对于我们的图形化界面系统代码设计而言，除了函数的功能实现之外，一个重要的功能是实现窗口界面和进行用户交互，我们从以下几个方面对我们的图形化界面的系统代码设计作简要的介绍

## （一）登录界面设置

我们在创建 QT5 Widgets Application 工程时，默认 mainwidget 为我们的主窗口程序界面，是我们整个程序进入后第一个打开的窗口，于是我们在 QT Designer 中进行布局，通过 QLabel 来设置提示信息文字，用 QPush Button 来设置按钮，用 QLine Edit 来设置文本框，并通过水平和垂直布局来排版界面使之工整。

创建相关信息后，我们进行登录，学生界面，退出三个不同按钮的槽函数的设置

对于退出按钮，我们直接在 QT Designer 中信号与槽部分直接添加信息，以下表5.1为添加信号与槽的联系

表 5.1: 退出按钮信号与槽设置

| 发送者 | 信号 | 接收者 | 槽 |
|---|---|---|---|
| exitpushButton | clicked() | MainWidget | close() |

对于学生界面按钮，我们在 QT Designer 中选择该按钮后右键点击转到槽，然后在槽函数中新建一个学生模式界面并设计窗口名称和图标后展现即可。

对于登录按钮，我们同上可以设计槽函数，不过在进入管理员模式界面前需要验证用户信息，通过判断条件验证通过后进入管理员模式界面，否则清空输入数据。

## （二）菜单界面设计

由于学生模式菜单界面和管理员模式菜单界面设计方法基本一致，以下已管理员模式菜单界面为例说明菜单界面设计，学生模式菜单界面不表。

我们在设定进入管理员模式界面后，点击按钮后在同一界面进行相关操作。对此，我们可以采用 QStackedLayout，QStackedLayout 继承自 QLayout，提

供了多页面切换的布局，一次只能看到一个界面，而其大部分控件都继承自 QFrame，而 QFrame 又继承自 QWidget，所以 QStackedLayout 完全可以实现窗口之间的切换。

我们在每一个界面类中实现一个信号函数，在按钮的槽函数中发送该信号，使用 QStackedLayout 存放所有的界面，在管理员窗口使用 connect 将信号和 QStackedLayout::setCurrentIndex 关联起来。

我们在 adminwidget.h 中修改相关定义，添加各个操作界面的成员，并在 adminwidget.cpp 中将 adminwidget 设为 stackLayout 布局，声明我们需要的窗口，并将其加入 stackLayout 布局中去。

之后在 adminmenuwidget.ui 用 QT Designer 设计菜单的相关按钮，为这些槽设计槽函数。

同时还要在 adminmenuwidget.h 中的 signals 里设计函数 display 作为 stackLayout 切换页面的信号。并且槽函数中用 emit 把 connect 把信号和槽进行关联，在这里 emit 某个信号，就相当于调用这个信号 connect 时所关联的槽函数。对于退出按钮直接使用 QApplication::exit() 即可。

之后我们回到 adminwidget.cpp 中，对于按钮的跳转我们还需要在 adminwidget.cpp 中用 connect 函数连接。在连接之后便实现了菜单的页面切换功能。

## （三）各功能界面设计

在实现菜单的页面切换功能之后，我们要对各个功能的界面进行设计，这也是我们开发图形化界面系统最为重要的部分，因为程序主要是用来实现功能的。以下是相关功能界面设计的相关内容。

### 1.添加学生信息

我们需要提示标签来指明输入的数据，也需要输入框来接收用户的输入，QLabel 与 QLineEdit 可以满足我们的要求。

QLabel，QLineEdit 有方法 setText(“内容”) 来改变其显示的文字，有方法 setGeometry(起始横坐标，起始纵坐标，宽，高) 来设置左上角起始位置与大小。不过这些当然这些都可以在 Qt Designer 中直接设置。打开 ui 文件，直接调整即可。

根据之前的 stuinfo 里面的变量信息，我们需要用 QLineEdit 来接受输入学生的学号、姓名、年龄、性别（采用 QcomboBox 菜单栏）、生日、地址、电话、成绩等相关信息，同时在文本框之前需要用 QLabel 来提醒输入学生信息数据，同时也需要一个增加按钮来实现将输入的信息写入文件中。

由于我们设计用文件对学生信息进行储存，所以需要引用 QFile，这里我使用对话框来提示用户操作中的各种问题，所以需要引入 QMessageBox。

我们在 lineEdit 中取出用户输入的数据，lineEdit 有 text() 函数，会返回一个 QString，而 QString 中又拥有入 toInt() toDouble() 等各种类型转换函数，可以适应我们的大多数要求。对于 comboBox, 我们可以根据 currentIndex 来获取菜单栏选择情况。

在取出数据的时候，除了处理存在空项和无法打开文件的情况下，我们还需要处理分数是否为 0——100，如果不在的话就会提示分数不符合区间从而不能写入文件中。

最后我们再打开数据文件将我们刚刚获取的数据追加在文件结尾，即可实现添加学生信息。

2.查询学生信息

同样我们可以用 QLabel 提醒用户输入数据，需要一个 QLineEdit 接收学生的学号，之后需要七个 QLabel 提醒用户显示的数据，同时另外需要七个 QLabel 来接收查询到的数据，最后需要一个 QPushButton 来提交查询。

在执行查询操作时，我们先从 QLineEdit 中获取到学号信息，之后再新建一个链表，从文件读入学生信息读入，采用 InputStudent 函数，之后再执行 SearchStudent 函数，如果查到的话返回 true，并且将获取的信息写入用来接入查询数据的 QLabel 中，否则返回 false，提示未找到学生信息。

3.删除学生信息

我们需要六个 QLineEdit 接收学生的姓名、年龄、生日、地址、电话、成绩, 性别采用 QcomboBox，需要七个 QLabel 提醒用户输入数据，需要一个 QPushButton 来根据学号删除学生，和根据学号修改学生信息。

在执行查询操作时，我们先从 QLineEdit 中获取到学号信息，之后再新建一个链表，从文件读入学生信息读入，采用 InputStudent 函数，之后再执行 DeleteStudent 函数，如果找到信息并且能够删除的话返回删除成功，并且将链表的更新的数据重新写入文件中，采用 OutputStudent 函数。否则提示未找到学生信息。

4.修改学生信息

同删除，我们需要七个 QLineEdit 接收学生的姓名、年龄、性别、生日、地址、电话、成绩，需要七个 QLabel 提醒用户输入数据，需要一个 QPushButton 来根据学号修改学生信息。

在执行获取修改前学生信息操作时，同查找功能类似，我们先从 QLineEdit 中获取到学号信息，之后再新建一个链表，从文件读入学生信息读入，采用 InputStudent 函数，之后再执行 SearchStudent 函数，如果查到的话返回 true，并且将获取的信息写入对应的 QLineEdit 和 QcomboBox 中，否则返回 false，提示未找到学生信息。

在获取修改前学生信息之后，我们可以进行相关信息的修改，完成之后我们可以进行修改操作，同样新建一个链表，从文件读入学生信息读入，采用 InputStudent 函数，之后再执行 ChangeStudent 函数，修改后将修改后的链表信息写入文件，采用 OutputStudent 函数, 若不能正常修改提示修改失败。

5.学生信息排序

对于学生信息排序，我们有按照学号进行排序和按成绩排序两种方法，对于选择所需的排序方法，QRadioButton 单选框可以解决我们的排序根据选择问题。对于排序结果的显示，我们可以采用 QTableWidget 表格可以用来显示排序后的信息。同时我们同样也需要一个 QPushButton 来提交排序操作。

对于排序，由于不需要用链表实现，我们采用 <algorithm> 中的 sort 函数，将数据从文件中读入类型为 stuinfo 的 QVector 容器后进行排序。

我们需要排序时只能根据学号和成绩中的一种进行排序，所以我们要实现代表这两种种排序方式的 QRedioButton 有且只有一个能被选中，我们可以将其加入一个 QButtonGroup，这样便可以实现这个需求。对应至于显示排序后的数据可以显示在表格控件 QTableWidget 中。

对应 QButtonGroup，它是直接继承自 QObject，它有一个属性 exclusive 当这个属性为真时，所有在 QButtonGroup 内的按钮同一时间只有一个可以被选择。在 QButtonGroup 中，我们采用了 addButton 添加按钮与 checkedId 获取选中按钮的 id。

QTableWidget 继承自 QTableView, 但 QTableWidget 有固定的数据模型，在像其中填入数据时只需要一个单元格一个单元格的填好就可以了 (setItem)

QTableWidget 我们用到了 setHorizontalHeaderLabels 设置水平表头标签 horizontalHeader()->setSectionResizeMode(QHeaderView::Stretch); 设置根据总宽度调整列宽；setRowCount 设置行数；setItem 设置单元格。我们在排序之后设置各单元格的数据，对于非 QString 类型的数据需要转换为 QString。

6.node 类中的函数实现

对于 node 类中的函数实现，基本代码均来自于 win32 位系统设计的代码, 具体请参考第四部分，不过对于 QT5 开发，我们在文件的读写和部分数据类型上进行了相关的修改使其适合于 QT5 框架，在此不在赘述。

# 六、系统实现情况

本章主要从两个不同的程序进行程序实现情况的展示。

## （一）win32 位程序

首先进入程序，在没有管理员信息时, 系统会提示没有管理员相关信息，需要设置管理员账号密码等相关信息，如图6.1：

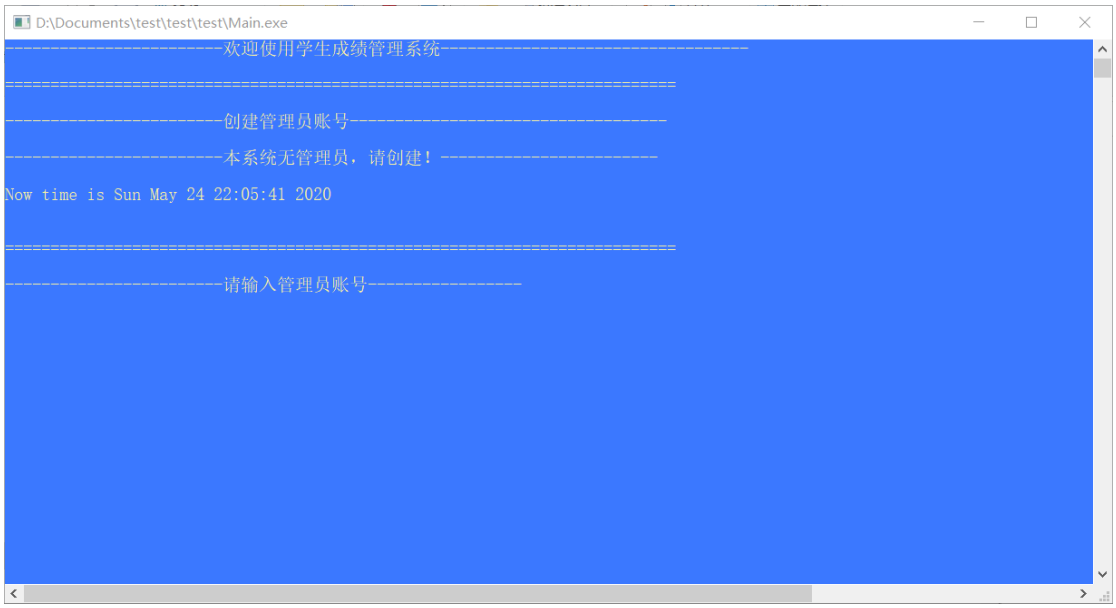


图 6.1: 没有管理信息时的界面

在图6.5进入程序后，必须先执行操作一，从文件输入相关信息建立链表。在图6.6之后可以进行相关操作。

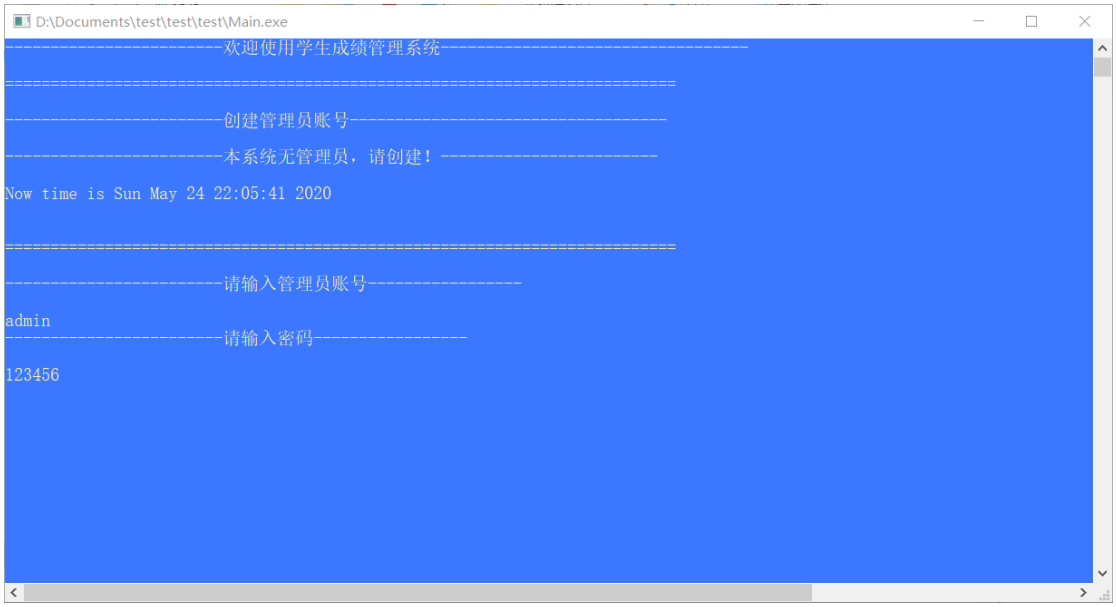


图 6.2: 输入待设置的账号密码信息

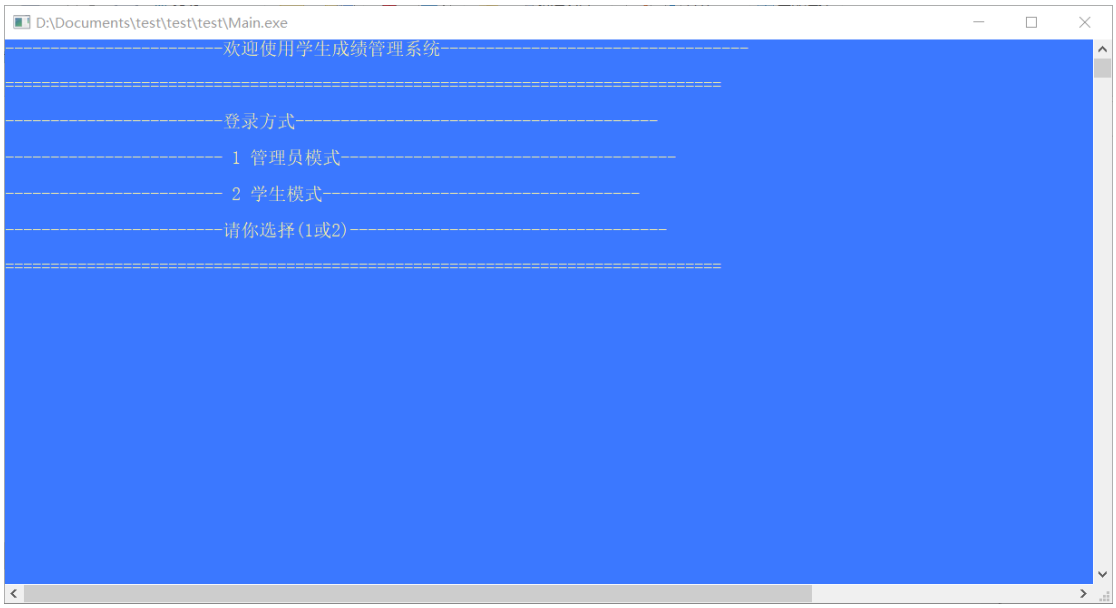


图 6.3: 设置完毕后重新进入登录界面

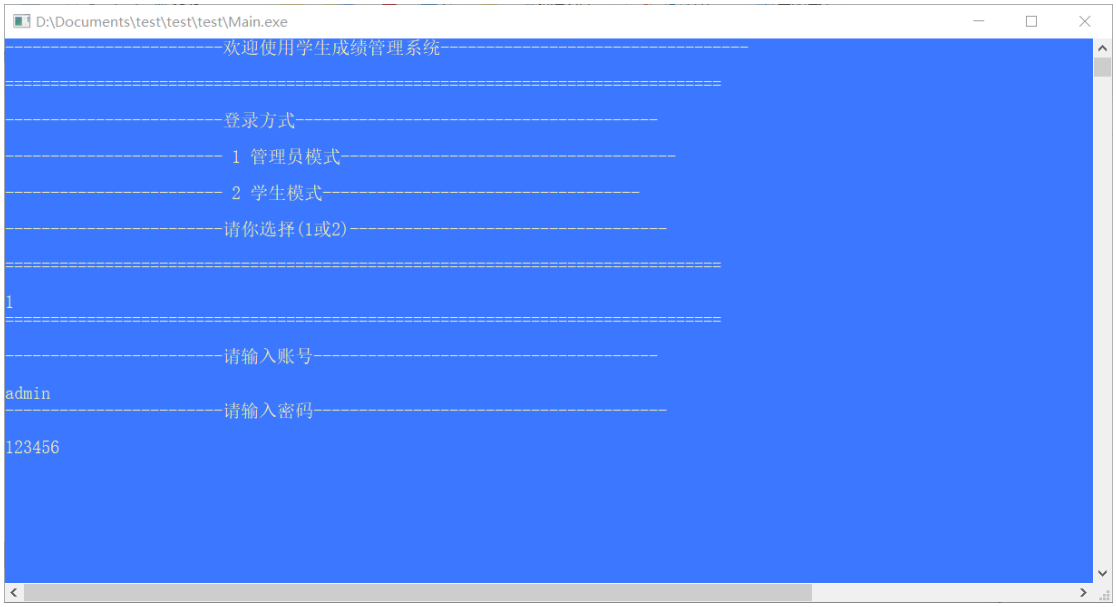


图 6.4: 输入账号信息

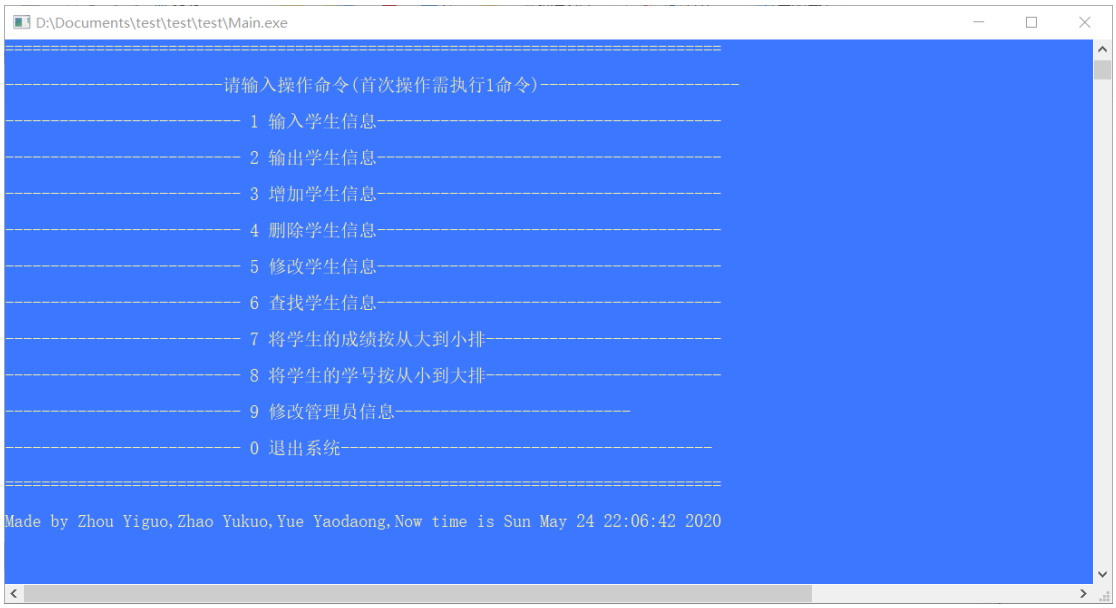


图 6.5: 进入管理员界面


图 6.6: 执行操作 1，输入学生信息

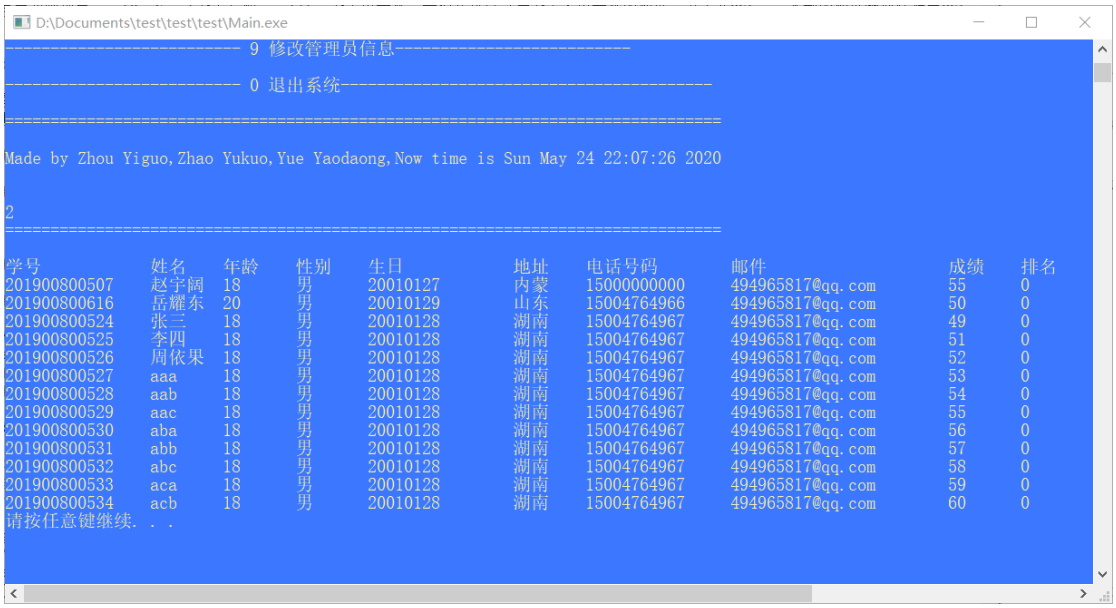


图 6.7: 执行操作 2，输出学生信息

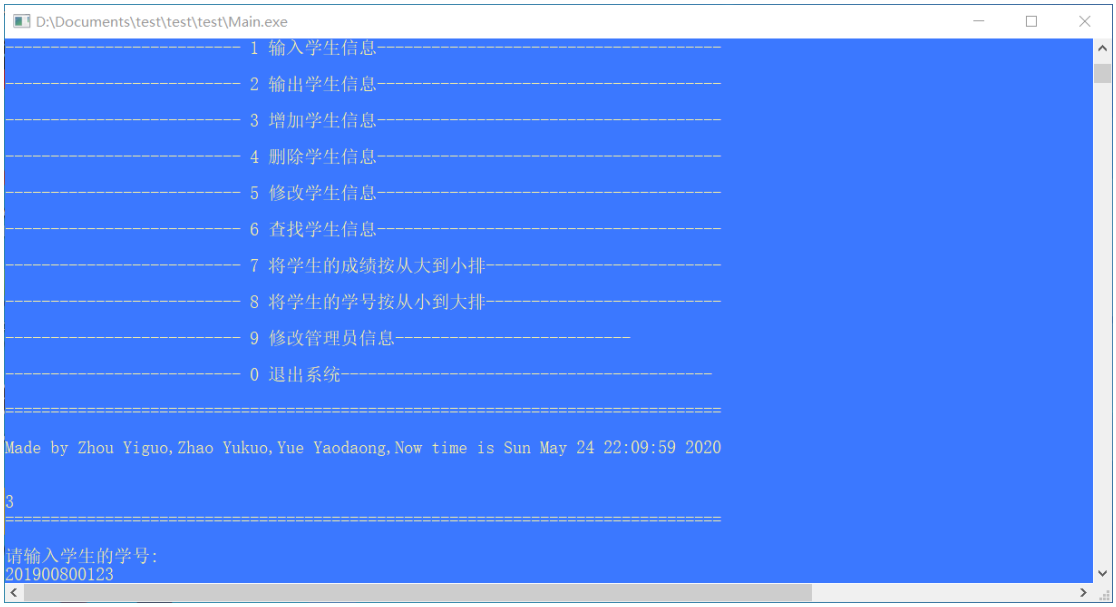


图 6.8: 执行操作 3，增加学生信息

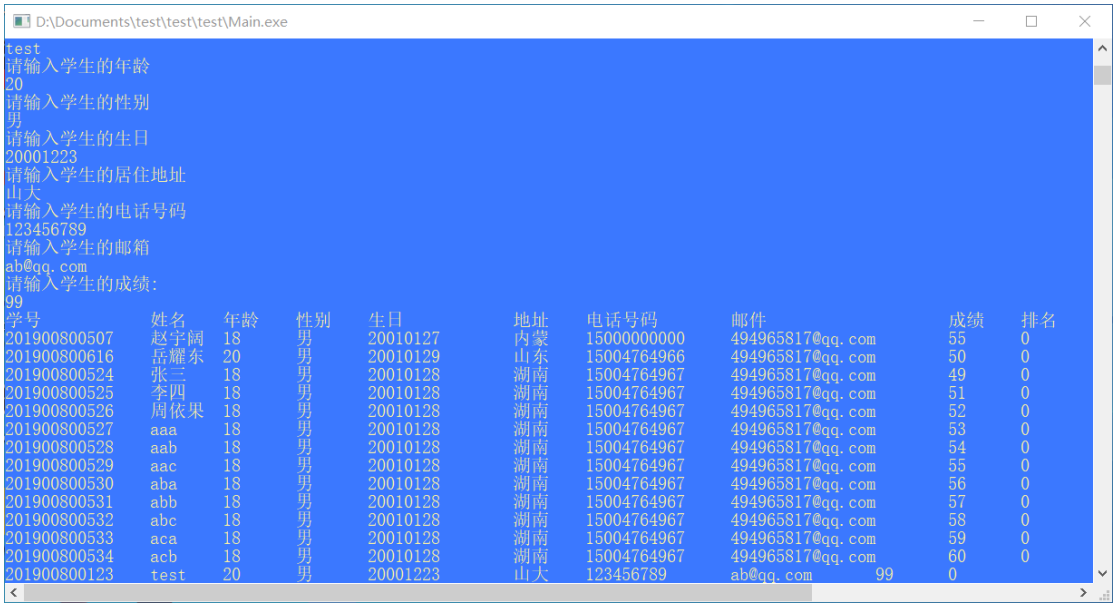


图 6.9: 执行操作 3，增加学生信息结果

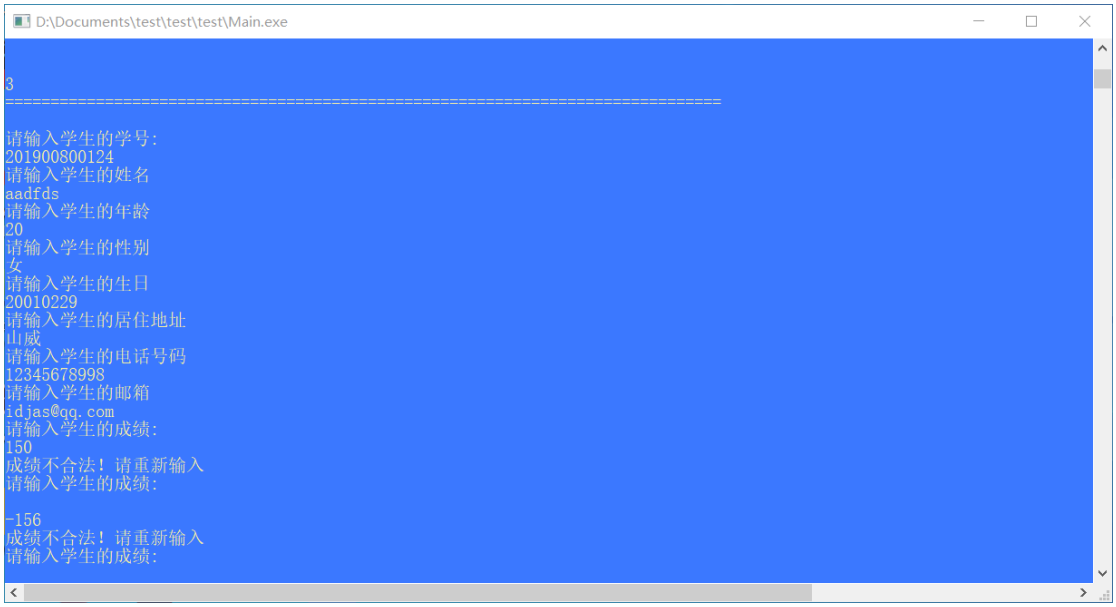


图 6.10: 执行操作 3，成绩不在范围内时

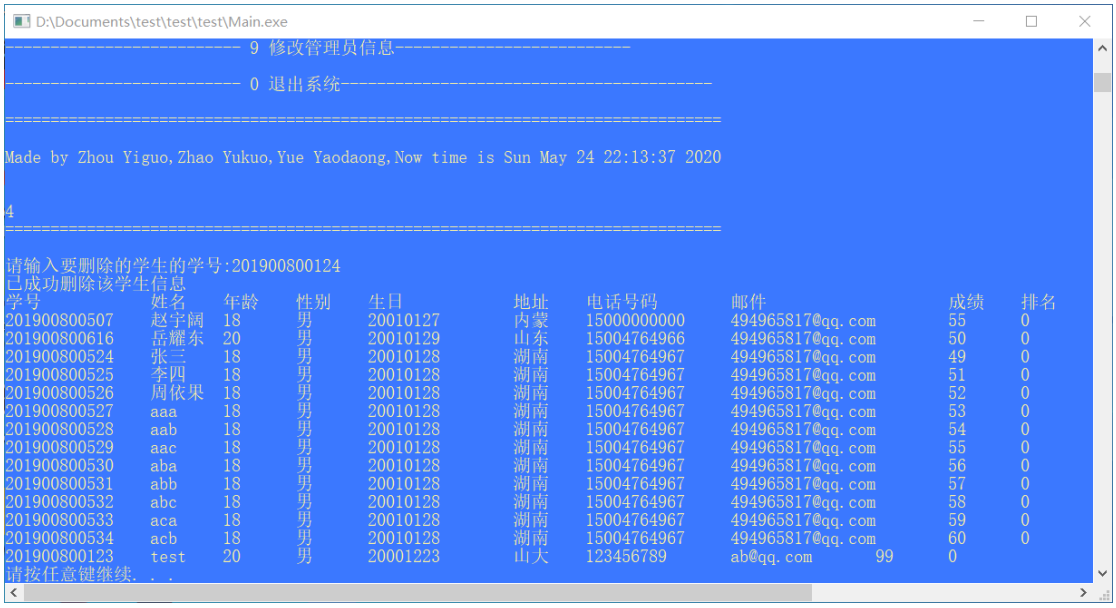


图 6.11: 执行操作 4，删除学生信息

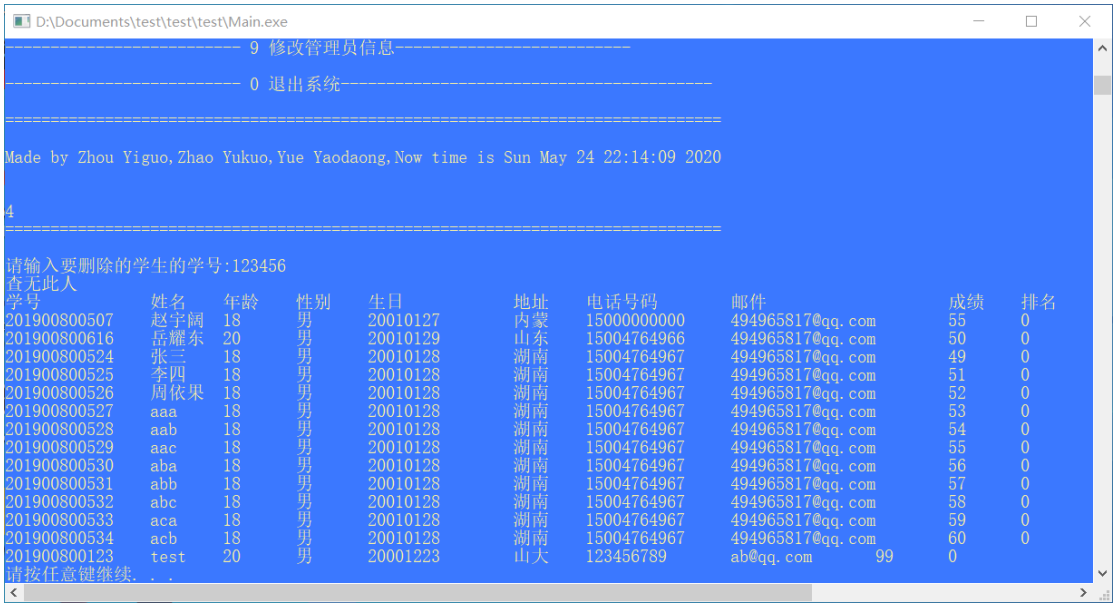


图 6.12: 执行操作 4，不存在已知学号时


图 6.13: 执行操作 5，修改学生信息菜单

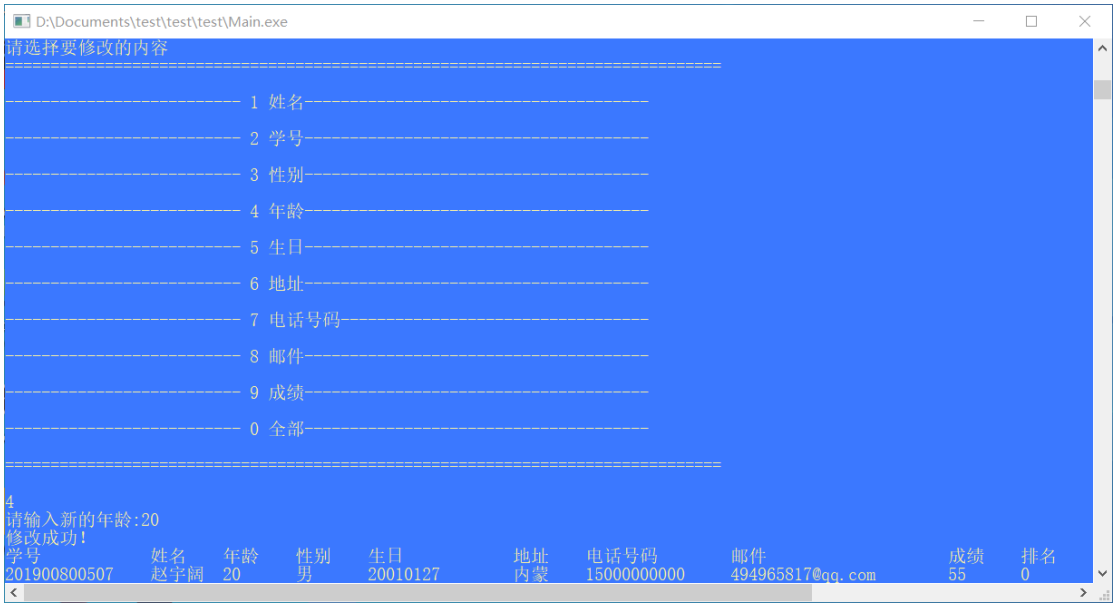


图 6.14: 执行操作 5，仅修改单个信息

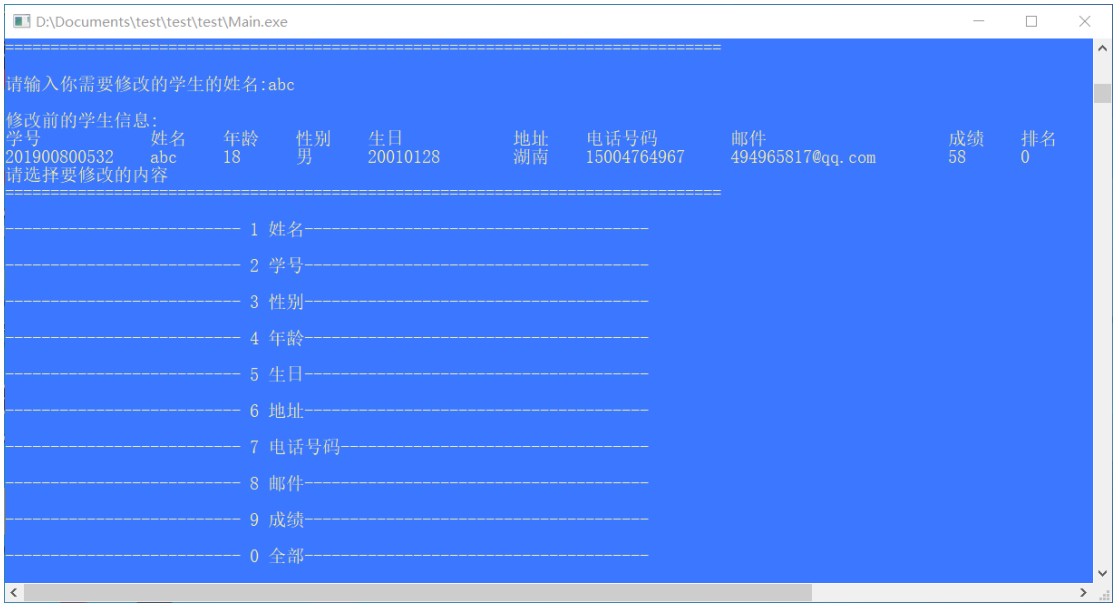


图 6.15: 执行操作 5，修改全部信息

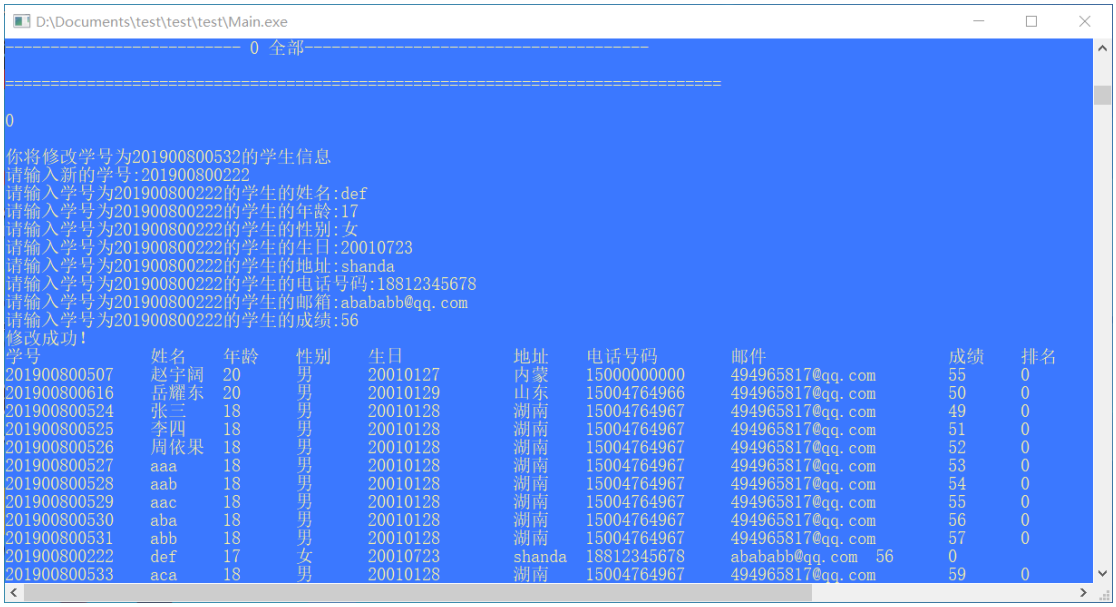


图 6.16: 执行操作 5，修改全部信息

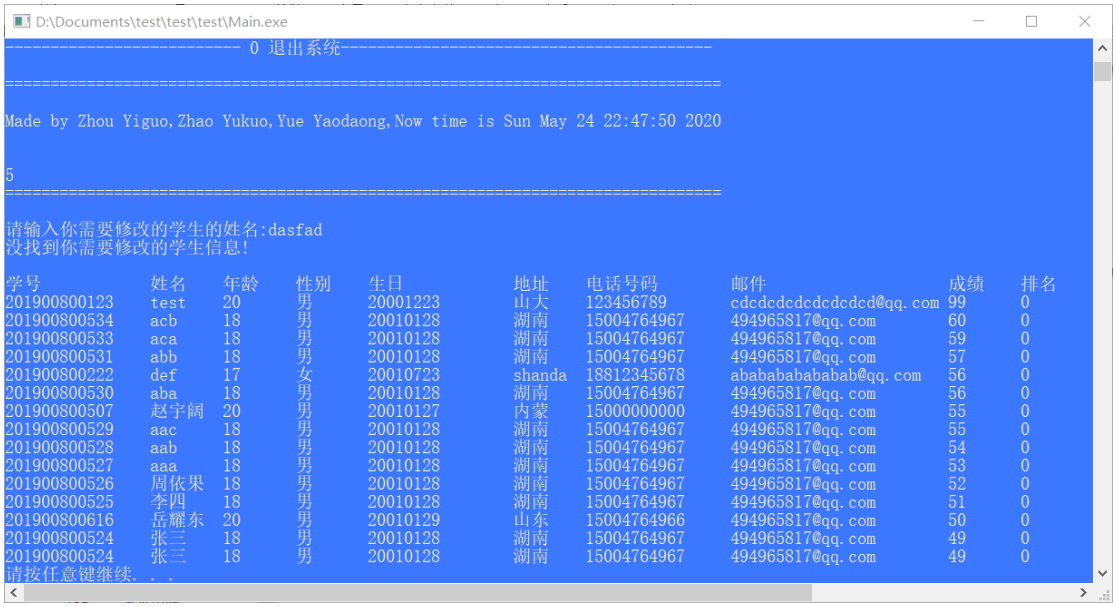


图 6.17: 执行操作 5，输入不存在的名字时


图 6.18: 执行操作 6，查找学生信息


图 6.19: 执行操作 6，查找时输入不存在的名字

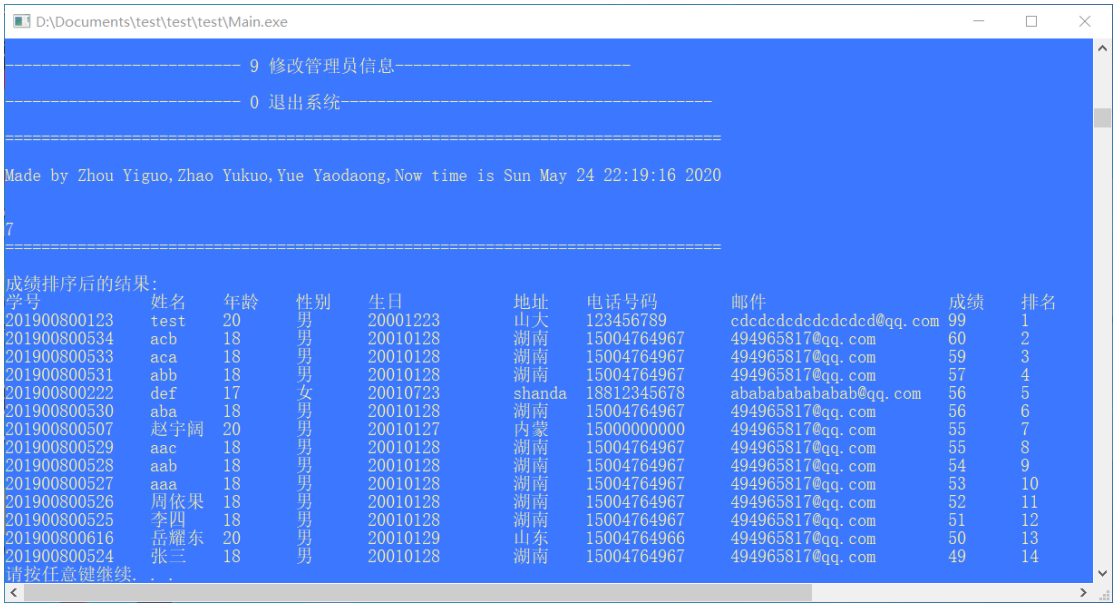


图 6.20: 执行操作 7，按成绩排序

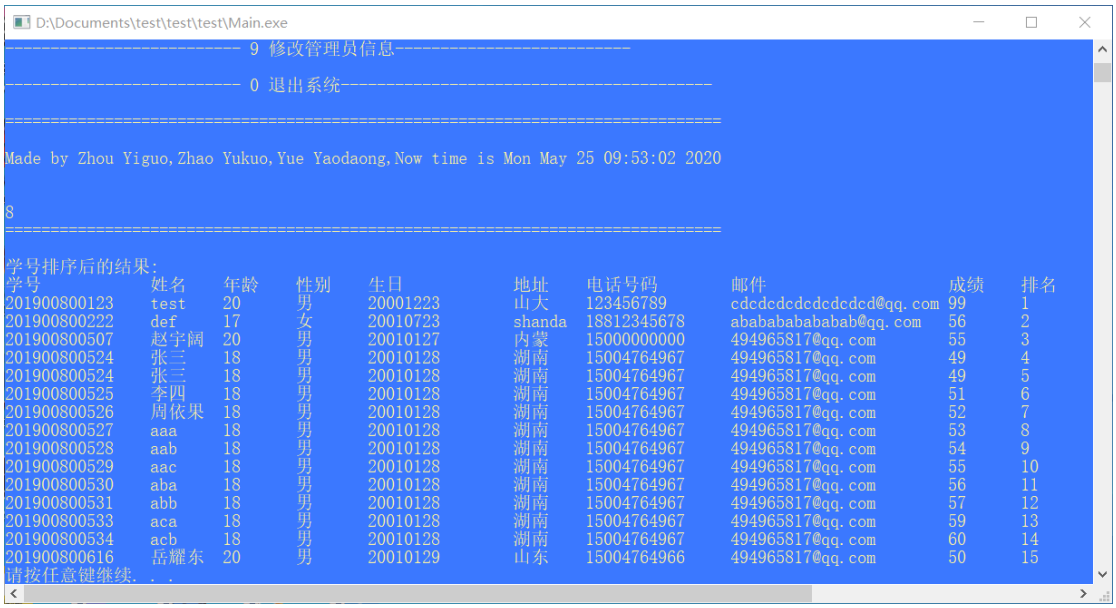


图 6.21: 执行操作 8，按学号排序


图 6.22: 执行操作 9，修改管理员信息

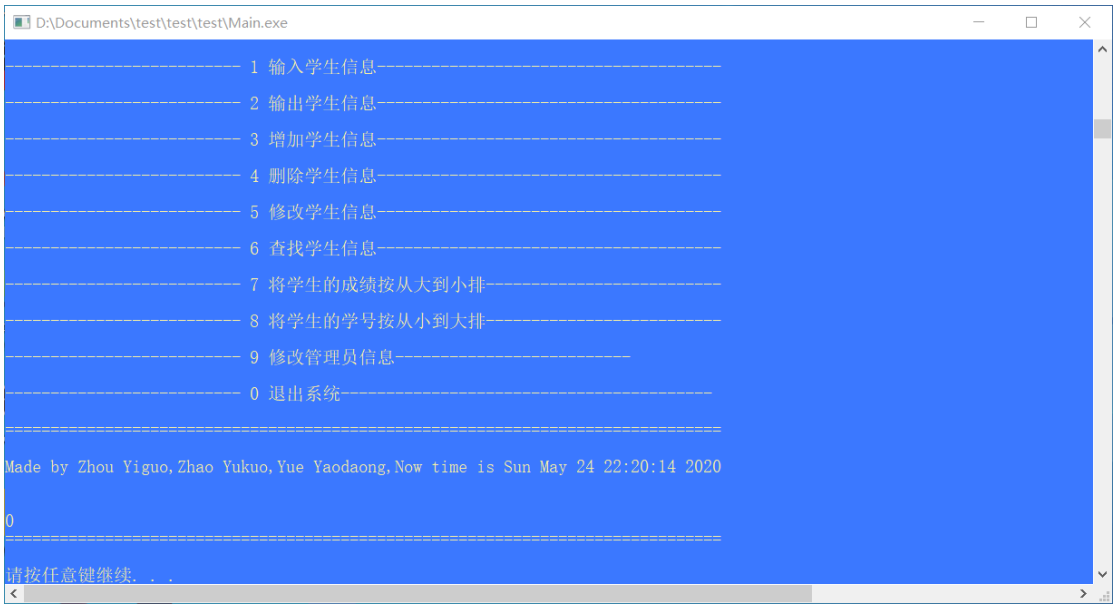


图 6.23: 执行操作 0，退出

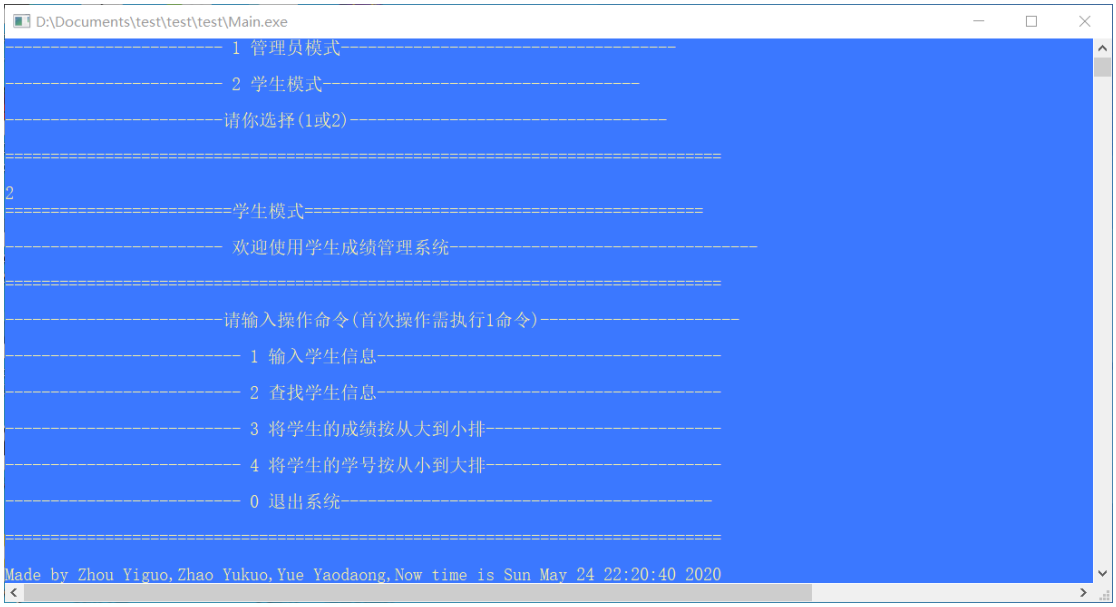


图 6.24: 学生模式界面 (学生模式下的操作不再演示)

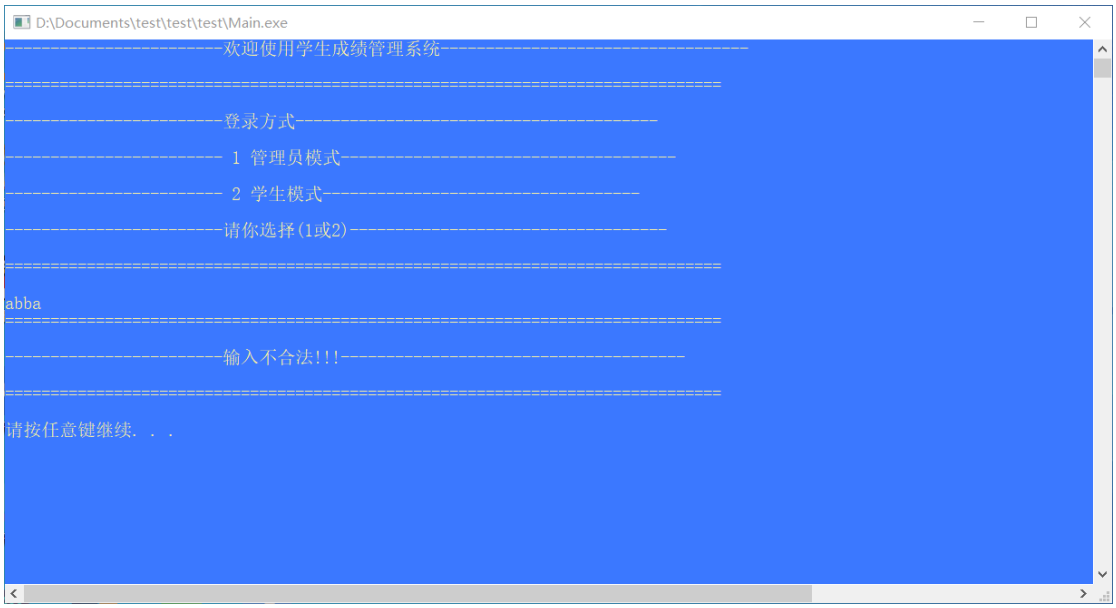


图 6.25: 选择非规定内容时结果

## （二）图形化界面程序

对于我们的图形化界面程序，首先进入登录界面。如下图


图 6.26: 登录界面


图 6.27: 输入账号密码信息

进入管理员界面，开始操作


图 6.28: 管理员模式菜单界面


图 6.29: 添加学生界面


图 6.30: 正常输入，成功保存

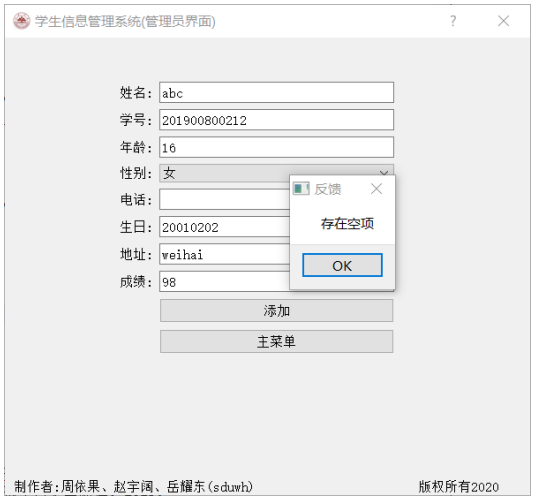


图 6.31: 存在空项

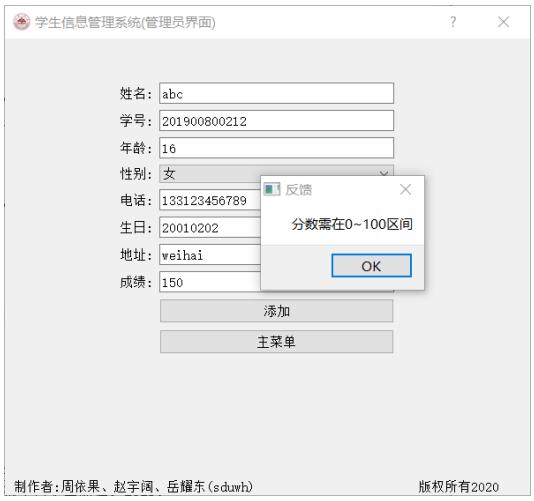


图 6.32: 成绩不在输入的区间范围内


图 6.33: 查找学生信息界面

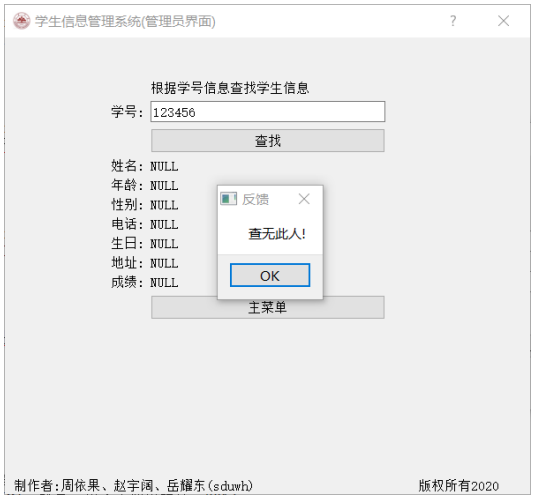


图 6.34: 没有查到相关学生信息


图 6.35: 查找到相关学生信息


图 6.36: 删除学生信息界面


图 6.37: 没有找到待删除的学生信息

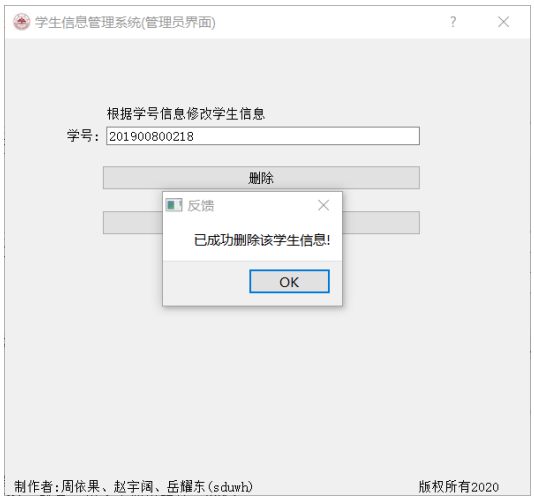


图 6.38: 成功找到删除的学生信息


图 6.39: 修改学生信息界面


图 6.40: 获取修改前信息时查无此人


图 6.41: 获取修改前信息

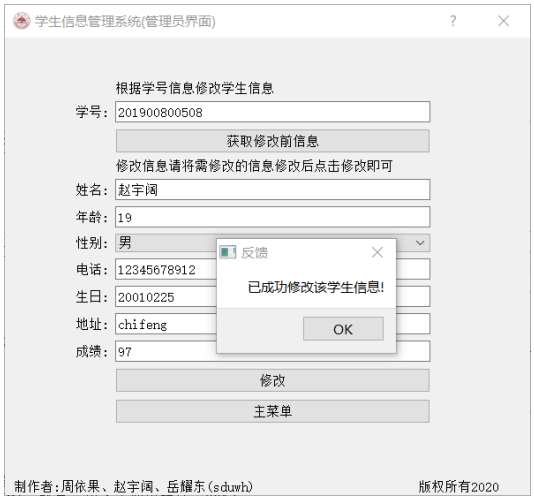


图 6.42: 成功修改学生信息


图 6.43: 按学号排序学生信息


图 6.44: 按成绩排序学生信息


图 6.45: 学生模式界面

# 七、系统程序调试

以下是在设计系统程序时出现的问题。

## （一）win32 位系统设计

### 1.SetScore() 函数

在执行 SetScore() 函数后学生排名未显示

首先检查的此函数是否能成功运行，因其非常简单只有一个 for 循环，检查后认为不是此函数出的问题，此后又检查与之相关的成员函数 SetS 与 GetS，也未发现问题，最后检查输出，发现未成功调用 GetS 函数，原来是 st.GetS，改成 p->st.GetS 后问题解决。

### 2.DeleteStudent() 函数

执行删除学生信息函数 DeleteStudent() 后下一个学生学号异常（bug1）

首先检查了删除节点后连接上下两个节点的方法，但能输出学生信息，说明成功连接了节点，然后考虑释放应删除的节点空间是否出了问题，发现代码是先连接，再删除节点，这样导致了应删除的节点地址丢失，且删除了其他错误的节点。改正后的方法是先用一个临时节点指针指向应删除节点，再连接上下两个节点，最后释放应删除节点的空间。

执行删除学生信息函数 DeleteStudent() 后其他学生学号异常（bug2）

多次测试后发现短学号在执行函数后正常显示，而过长的学号会自动发生变化，显示异常，所以认为是学号数据类型 long long 的问题，查找资料后得知长学号并未超出 long long 的范围，于是检查函数发现在读入要删除的学生学号时数据类型为 int，长学号会超过 int 范围，修改为 long long 后问题解决。

### 3.ChangeStudent() 函数

执行修改学生信息函数 ChangeStudent() 后该学生学号异常

检查函数后发现 case 语句中只是将要修改的学生信息读入，并未执行相应的设置函数，而是在 switch() 外将所有设置函数都执行，这样必然会导致数据出现错误，修改为每条 case 语句中读入要修改的信息，且执行相应的设置函数后再跳出，问题解决。

4.InputStudent() 函数

1. 出现多次段错误。

原因：链表使用不熟，多次指到了不应该指向的地方

解决方案：及时将指针的 pNext 更新

2. 在其他函数中学号显示错误

原因：学号超出 int 范围

解决方案：将学号改为 long long，其他函数中也进行改动

3. 在动态链表中未能成功使用 fwrite,fread 等函数，未能很好的实现功能

解决方案：使用 infile，ios::in 等文件操作流代替

5.OutputStudent() 函数

1. 输出成绩时，无论文件中的成绩为何值，输出的成绩均为零。

原因：Student 中函数使用错误

解决方案：将 p->st.GetScore() 改为 score

2. 出现段错误

原因：链表使用不熟，在一开始 p = pHead->pNext 的情况下，遍历时使用 (p->pNext != NULL)

解决方案：将判断条件中的 p->pNext 改为 p 即可

6.AddStudent() 函数

1. 未考虑成绩不在 0 100 之间的情况

在输入成绩时，增加了对成绩范围的判断

2. 出现段错误

原因：1. 遍历链表时，判断条件为 p != NULL2. 最后忘记 pt->pNext = NULL

解决方案：1. 改为 p -> pNext != NULL2. 加上 pt -> pNext = NULL

3. 在增加学生信息的时候，直接输出是正确的，但是增加信息后文件并没有改变。即增加操作没有对文件产生影响。

解决方案：新添加了一个 FileOutput 函数，使得在输出信息的同时，能够将信息重新写入文件中。

7.menu_login() 函数

在输入命令时，若输入字母程序会运行时错误

原因：命令用的 int 存储，读入时当然会出错。

解决方案：将存储命令的变量用 string 存储，这样就不会报错了。

## （二）图形化程序设计

1.addstudentwidget 类

问题：增加学生信息后，查看学生信息出现显示异常，添加内容出现移位

原因：增加学生信息时，将信息写入文件时按照的信息顺序有误。

解决方案：重新修改写入信息的顺序即可

2.mainwidget 类

问题：选择学生界面时，程序无响应。

原因：在点击学生界面的槽函数中设置窗口标题时误输入成管理员界面的 ui

解决方案：修改成学生界面的 ui 即可。

3.modifywidget 类

优化：可以提前获取待修改的学号信息，以便重复输入无需修改的信息。

解决方案：添加一个 QPushButton 来获取学生信息，并将获得的信息加载到文本框和菜单选择栏中，采用查找类中的相关函数，稍作修改即可。

4.node 类

问题：在查找学生信息时，信息没有更新。

原因：函数调用时采用的形参，没有进行传值调用。

解决方案：在需要修改的变量加入 & 来进行调用。

5.登录界面

问题：在之前使用的登录界面不能够直接打开学生界面，会造成程序运行错误。

原因：由于之前设计时将管理员菜单界面设为了主窗口界面，导致不能在主窗口界面之前新生成窗口界面。

解决方案：将登录界面设为主窗口界面，管理员菜单界面和学生菜单界面设为平级关系，分别设置菜单和窗口界面，这样就可以解决该问题。具体设计一系列的代码修改在此不表。

# 八、总结与不足

本设计使用 C/C++ 程序设计语言，采用链表实现实现增删改查功能，同时程序将学生信息均采用文件储存，也设计了具有分角色管理使用功能。较好的实现了大作业的相关要求。

不过对于 win32 位系统设计而言，不足之处在于需要先进行初始化相关操作，且在输出信息时界面可能会出现错位的情况。

对于 GUI 系统设计而言，界面的美观度有待提高，管理员信息不能直接修改，可能需要进行更多知识的学习来实现相关操作。

从第十一周开始，直到现在，我们三位同学团结合作，结合自己所学和临时从网上或论文中学习相关知识，在短短几周的时间内占用许多原本的闲暇时间完成了程序设计，锻炼了我们的代码能力，同时巩固了链表和面向对象等本学期所新学习的内容和操作。

相信这次的课程设计为我们之后继续从事程序设计工作奠定了坚实的基础，打下了良好的开端。

由于这是第一次进行系统设计和实现，撰写的论文难免会有不足，希望老师和同学们能够指出不足之处或者提出修改的建议，之后可以不断完善和提高。

# 参考文献

[1] 钟玲玲, 刘冬雪, 黄小平, 吴密. 基于 C 语言的学生信息管理系统设计与实现 [J]. 河南科技学院学报 (自然科学版),2019,47(04):62-67+78.

[2] 隋郁. 高校学生信息管理的系统设计与实现 [J]. 教育发展研究,2017(S1):13-15.

[3] 刘家熙. 基于 VC 的学生教务信息管理系统设计 [J]. 电脑知识与技术,2017,13(01):61-62.

# 谢　　辞

学无止境，气有浩然。

感谢老师、感谢同学、感谢自己