**上海应用技术大学**

**课程设计报告**

**课程名称：面向对象程序设计实训**

**2022-2023（1）**

项目名称： 学生成绩管理系统

**学 号 2110450209**

**姓 名 赵诗文**

**专 业、年级 人工智能21级**

**指 导 教 师 王志敏**

**提交日期： 2023年1月6日**

目录

[1 任务书 1](#_Toc30051)

[2 系统简介 2](#_Toc9724)

[3 系统设计说明 3](#_Toc31585)

[3.1 需求分析 3](#_Toc11475)

[3.1.1 开发学生成绩管理系统的背景分析 3](#_Toc3599)

[3.1.2 开发学生成绩管理系统的必要性 3](#_Toc10841)

[3.1.3 开发学生成绩管理系统的可行性 4](#_Toc29550)

[3.1.4 学生成绩管理系统的系统需求 4](#_Toc25733)

[3.1.5 学生管理系统的功能需求分析 4](#_Toc31153)

[3.2 系统功能描述 5](#_Toc22661)

[3.2.1 适用范围 5](#_Toc6636)

[3.2.2 功能介绍 5](#_Toc20797)

[3.3 功能模块划分 5](#_Toc14636)

[3.3.1 学生登录模块 5](#_Toc5743)

[3.3.2 教师登录模块 6](#_Toc17254)

[3.3.3 管理员登录模块 6](#_Toc8818)

[3.4 系统流程分析 6](#_Toc30439)

[3.4.1 系统主流程图分析 6](#_Toc31340)

[3.4.2 系统详细流程图分析 7](#_Toc11299)

[4 系统界面设计说明 10](#_Toc25945)

[4.1 界面列表 10](#_Toc1301)

[4.2 各界面中主要变量、函数 11](#_Toc24692)

[4.2.1 “Person”类 11](#_Toc2194)

[4.2.2 “Student”类 12](#_Toc25527)

[4.2.3 “Teacher”类 13](#_Toc4410)

[4.2.4 “Manager”类 14](#_Toc24462)

[4.3 主要功能的核心源程序 33](#_Toc29932)

[4.3.1 程序预处理 33](#_Toc25844)

[4.3.2 主函数main() 34](#_Toc12522)

[4.4 运行结果 35](#_Toc13193)

[4.4.1 系统初始状态 35](#_Toc2068)

[4.4.2 学生登录 35](#_Toc1392)

[4.4.3 教师登录 37](#_Toc11220)

[4.4.4 管理员登录 41](#_Toc28397)

[4.4.5 系统退出 42](#_Toc3522)

[5 遗留问题 43](#_Toc6170)

[6 设计体会 44](#_Toc9984)

# 任务书

手机屏幕截图

描述已自动生成

# 系统简介

随着网络通信技术的迅速发展和信息技术的广泛应用，计算机的应用已普及到经济和社会生活的各个领域。教育领域也不例外，现今已经出现了各种远程网络在线教育系统、图书馆管理系统、学籍管理系统等。其中学生成绩管理是任一院校都不可缺少的部分，它的内容对于学校的决策者和教学管理者来说都至关重要。学生信息管理系统是一个教育单位不可缺少的部分，它的内容对学校的决策者来说是至关重要的，所以学生成绩管理系统应该能够为用户提供充足的信息和快捷的查询手段。但一直以来人们使用传统人工的方式管理学生成绩，这种管理方式存在着许多的缺点，如:效率低、容易作假，时间一长，将产生大量的文件和数据，这对于查找、更新和维护都带来了不少的困难，为数据的查询、更新和维护代来了极大的不便。所以学生成绩管理信息系统的开发和应用是必要的，它能够为管理人员提供充足的信息和快捷的查询手段，提高学校成绩管理的效率。

此学生成绩管理系统（以下简称“系统”）是基于C++语言，充分利用C++面向对象程序设计抽象、封装、继承和多态性等特点与优势，以达到高效准确管理学生成绩的目的。

# 系统设计说明

## 需求分析

### 开发学生成绩管理系统的背景分析

当今时代是飞速发展的信息时代，在各行各业中离不开信息处理，这正是计算机被广泛应用于信息管理系统的环境。计算机的最大好处在于利用它能够进行信息管理。使用计算机进行信息控制，不仅提高了工作效率，而且大大的提高了其安全性。尤其对于复杂的信息管理，计算机能够充分发挥它的优越性。管理信息系统是进行信息的采集、存储、加工、维护和使用的系统，它是随着管理科学和技术科学的发展而形成的。学生成绩管理系统是一个教育单位不可缺少的部分，它的内容对于学校的决策者和管理者来说都至关重要，所以学生成绩管理系统应该能够为用户提供充足的信息和快捷的查询手段，对学生来说可以轻松的查阅自己在校的成绩以及信息等。但是一直以来人们都是靠传统人工的方式管理学生成绩，这种管理方式存在着许多缺点，如:效率低、保密性差，另外时间一长，将产生大量的文件和数据，这对于查找、更新和维护都带来了不少的困难。随着科学技术的不断提高，计算机科学日渐成熟，其强大的功能已为人们深刻认识，它已进入人类社会的各个领域并发挥着越来越重要的作用。作为计算机应用的一部分使用计算机对学生成绩信息进行管理，有着手工管理所无法比拟的优点，例如：检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高学生成绩管理的效率，也是教育机构的科学化、正规化管理，与世界接轨的重要条件。因此，开发这样的一套学生成绩管理系统成为很有必要的事情。学生成绩管理系统提供了强大的学生成绩管理管理功能，方便系统管理员对学生成绩等信息的查看、添加、修改、删除、查询等操作，同时一样的方面学生对自己各科成绩查询。

### 开发学生成绩管理系统的必要性

在21世纪的科技时代，科学技术突飞猛进，计算机已经不仅是在科技上应用，而且在生活中也是同样得到了广泛在应用。如今，不管是小学、初中、高中，甚至是很多大学生的学生成绩管理基本上是靠人工进行管理，但随着时间的变化，学校规模的扩大有关学生成绩管理工作和所涉及到的数据量越来越大越来越多，大多数学校不得不靠增加人力、物力、财力来进行学生成绩管理。但是人工管理成绩档案具有效率低、查找麻烦、可靠性不高、保密性低等因素。因此开发出一个不仅仅是适用于大中专院校以及其它高校通用的学生成绩管理系统是必要的。

开发一个学生成绩管理系统，采用计算机对学生成绩进行管理，进一步提高了办学效益和现代化水平。为广大教师和学生提高工作效率，实现学生成绩信息管理工作流程的系统化、规范化和自动化。现在我国的大中专院校的学生成绩档案管理水平普遍都不是很高，有的还停留在全用纸介质基础上，这种管理方式已不能适应时代的发展，社会的需求，因为它浪费了大量的人力物力，也存在着许多不足的因素。在今天信息时代这种传统的管理方法必然会被计算机为基础的信息管理系统所代替。一个高效的学生成绩管理系统可以存储历届的学生成绩档案，不需要大量的人力，只需要几名专门录入员即可操作系统，节省大量人力，可以迅速查到所需信息、高效、安全，学生在能方便的查看自己的成绩。

### 开发学生成绩管理系统的可行性

计算机现已经进入千家万户，大部分的学生是可以使用上计算机的，尤其是大学生，几乎是人手一台笔记本电脑，所以系统的可使用范围比较广。

系统基于C++语言开发，灵活和通用性较高，语法相对简单，容易学习和使用。使用面向对象程序设计方法开发系统，易维护、效率高、易扩展。

### 学生成绩管理系统的系统需求

在21世纪的科技时代，科学技术突飞猛进，信息社会的高科技时代，商品经济化的高效益，计算机的应用已普及到经济和社会生活的各个领域。计算机虽然与人类的关系愈来愈密切，还有人由于计算机操作不方便继续用手工劳动。为了适应现代社会人们高度强烈的时间观念，学生成绩管理系统为教学办公带来了极大的方便。该学生成绩管理系统是以C++语言编写。系统管理员、教师、学生只需通过简单的操作，用户都可以了解本系统软件的基本工作原理。用户只需进行输入一些简单数字，即可达到自己想要的目标。

### 学生管理系统的功能需求分析

作为一个学生成绩管理系统，该程序的使用场景在学校，使用人群为学校的教师和学生。所以该程序在充分考虑教师和学生的使用场景后，提供功能需求分析。

对于学生，每位学生拥有唯一的账号并搭配密码，使用账号和密码登陆到系统后，可以查询自己的分数等信息。

对于教师，也拥有唯一的账号并搭配密码，使用账号和密码登录到系统后，可以录入学生的成绩、查看学生的成绩、修改学生的成绩、删除学生的成绩，也可以对成绩进行基本的数据分析操作：排序、分类、搜索、筛选。比如，能够按照学号或姓名及成绩等进行排序和筛选，够按照班级或专业进行分类的管理。在录入成绩和保存成绩时，为了方便数据的存储和维护，系统可提供对成绩文件的读取和写入功能。

为了方便对学生和教师的账号以及系统的管理，开辟管理员登录选项，其作用为为忘记账号密码的学生和教师重置密码和维护系统。

## 系统功能描述

### 适用范围

本系统广泛适用于各大、中、小学校等教育机构使用，促进成绩管理信息化建设。

### 功能介绍

学生登录该系统后，可以查看自己的成绩和修改自己的密码。教师登录该系统后，可以查看所有学生的成绩，并且可以对成绩进行排序查看。教师还可以添加学生信息、删除学生信息、修改学生信息、查看学生信息和清空学生信息操作。管理员登录该系统后，可以重置学生和教师的密码，以方便学生和教师登录。

## 功能模块划分

如图3.3-1所示，本系统主要分四大模块。

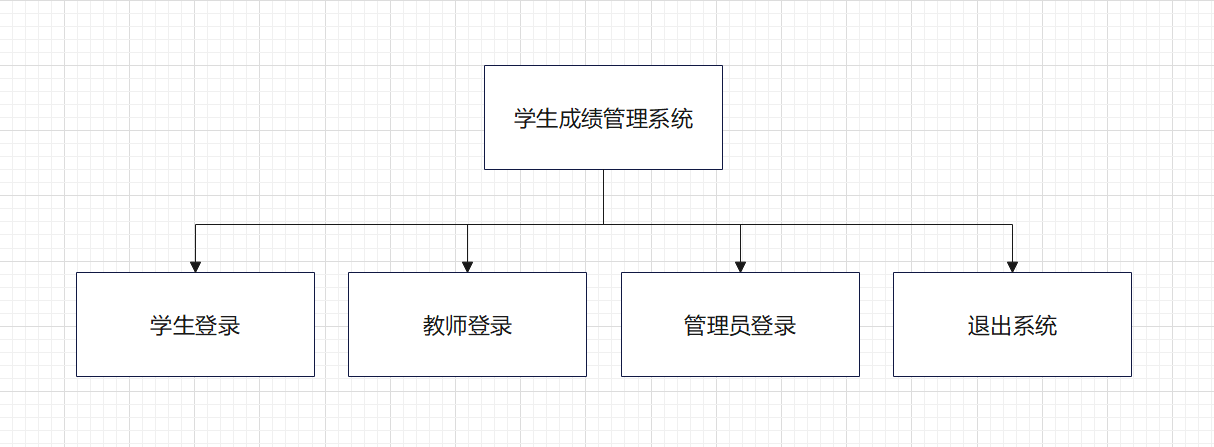


图 3.3-1 系统主要四大模块

### 学生登录模块

进入该模块后，学生使用账号和密码登录。账号和密码匹配正确登录后，系统又分为3个模块，分别是查看成绩、修改密码和返回上一级。如图3.3-2所示。

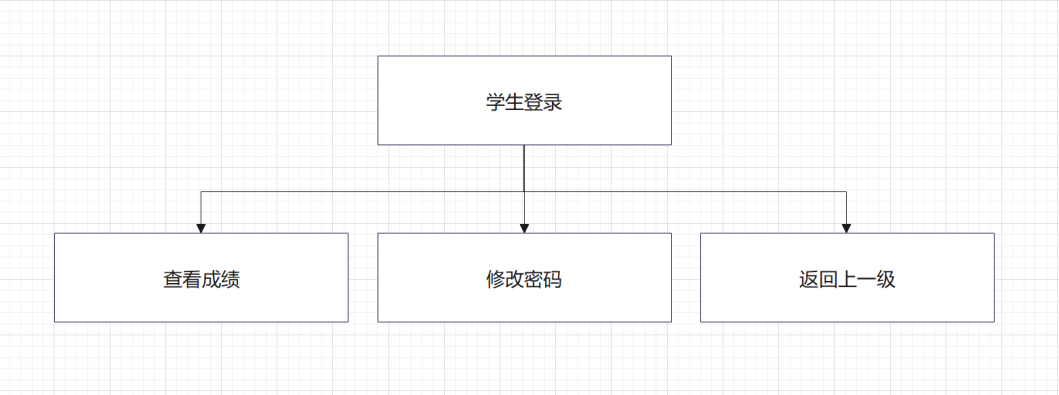


图 3.3-2 学生登录模块

### 教师登录模块

进入该模块后，教师使用账号和密码登录。账号和密码匹配正确登录后，系统又分为7个模块，分别是查看学生信息、添加学生信息、删除学生信息、修改学生信息、查找学生信息、清空学生信息和返回上一级。如图3.3-3所示。

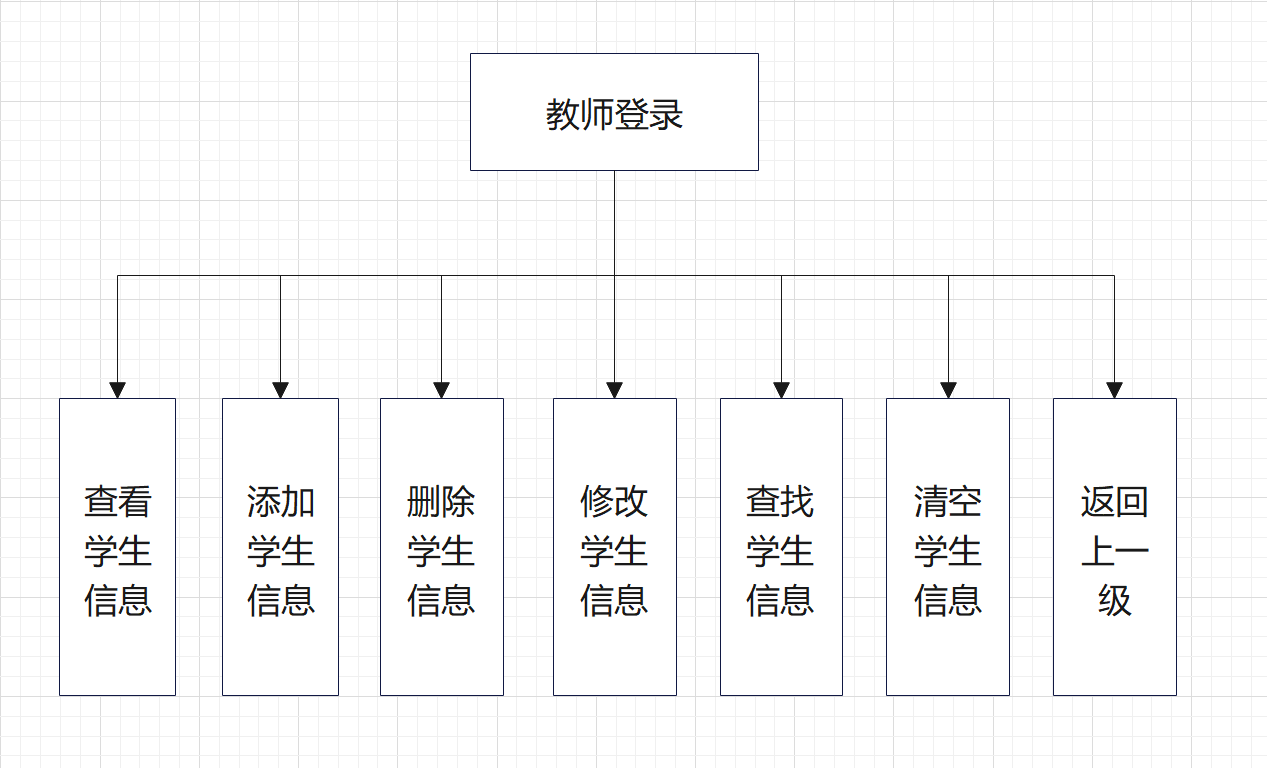


图 3.3-3 教师登录模块

### 管理员登录模块

进入该模块后，输入管理员操作密码，密码输入正确后，系统又分为3个模块，分别是重置学生密码、重置教师密码和返回上一级。如图3.3-4所示。

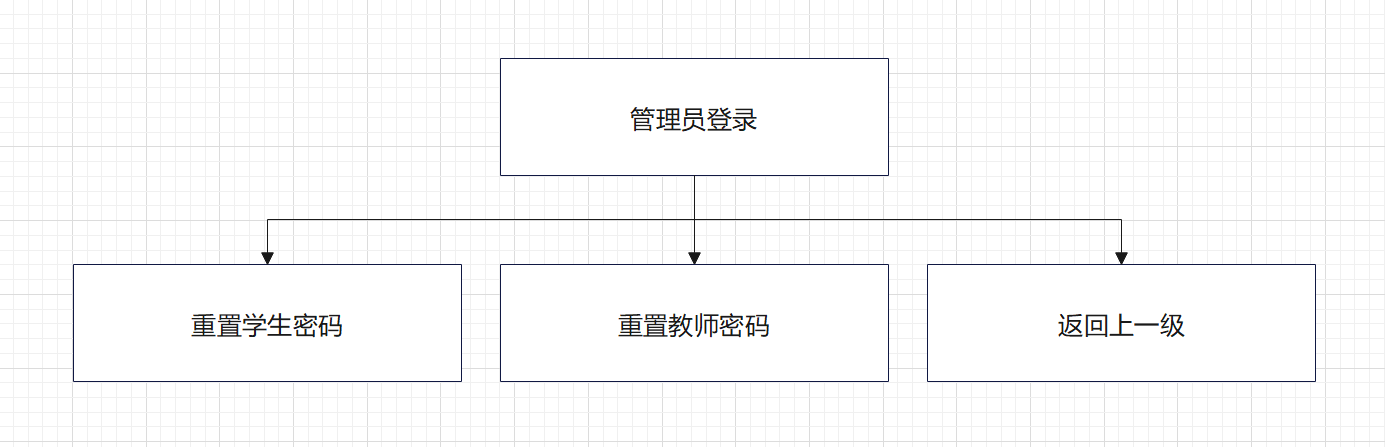


图 3.3-4 管理员登录模块

## 系统流程分析

### 系统主流程图分析

本系统主流程图如图3.4-1所示。系统运行后，显示一级菜单，用户根据不同身份选择不同登录方式。学生登陆后，显示二级学生登录菜单，学生可继续选择查看成绩、修改密码和返回上一级。教师登录后，显示二级教师登录菜单，教师可继续选择查看学生成绩、添加学生成绩、删除学生成绩、修改学生成绩、查找学生成绩、清空学生成绩和返回上一级。管理员登录后，显示二级管理员登录菜单，管理员可继续选择重置学生密码、重置教师密码和返回上一级。

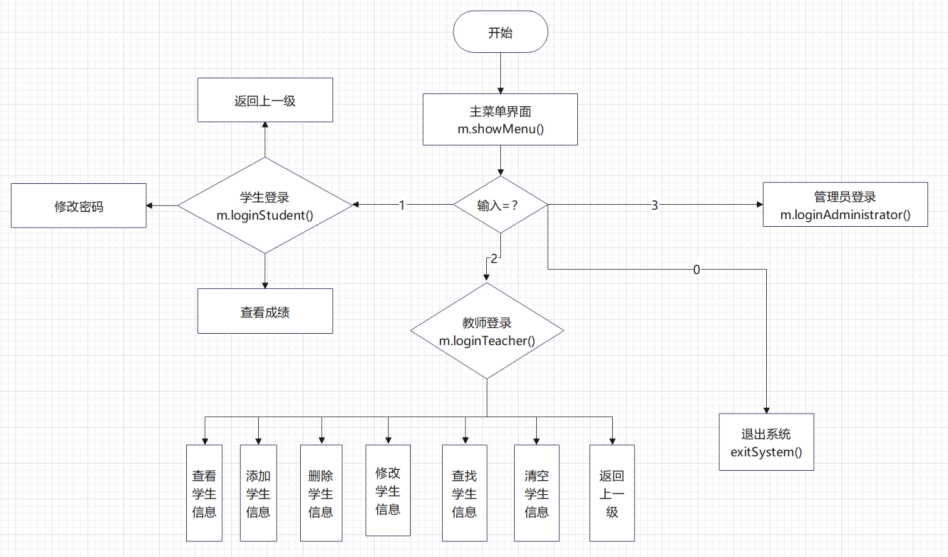


图 3.4-1 系统主流程图

### 系统详细流程图分析

本系统主流程图如图3.4-2所示。

首先，系统创建Manager类对象：m，用来控制整个程序。在创建m对象时，m对象的构造函数会创建一个默认Teacher类对象：teacher1；设置默认管理员密码；读studentFile.txt文件，若文件不存在则将提示用户；调用initStudent()函数初始化学生对象。然后系统进入while(true)循环，等待用户输入。若用户输入“1”，m调用loginStudent()函数，进行学生登录；若用户输入“2”，m调用loginTeacher()函数，进行教师登录；若用户输入“3”，m调用loginAdministrator()函数，进行管理员登录；若用户输入“0”，m调用exitSystem()函数，退出系统；若用户输入无效字符，系统将重新进入while(true)循环。

m调用loginStudent()函数，学生使用账号和密码登录。若账号存在且密码正确后，正式进入学生端，此时会调用showMenuStudent()函数，显示二级学生登录菜单，并等待用户输入。若用户输入“1”，系统将显示学生成绩等信息。若用户输入“2”，系统将提示用户完成修改密码操作。在修改密码时，提示用户输入两次新密码，如果输入的两次的新密码不同，系统将提示修改密码失败。如果新密码与旧密码相同也无法完成修改密码操作。以上两项操作在结束后，系统都将回到学生端的界面。若用户输入“0”，系统将返回上一级菜单，即回到主界面while(true)循环。若用户在登录时，账号不存在或密码错误，系统也将回到主界面。

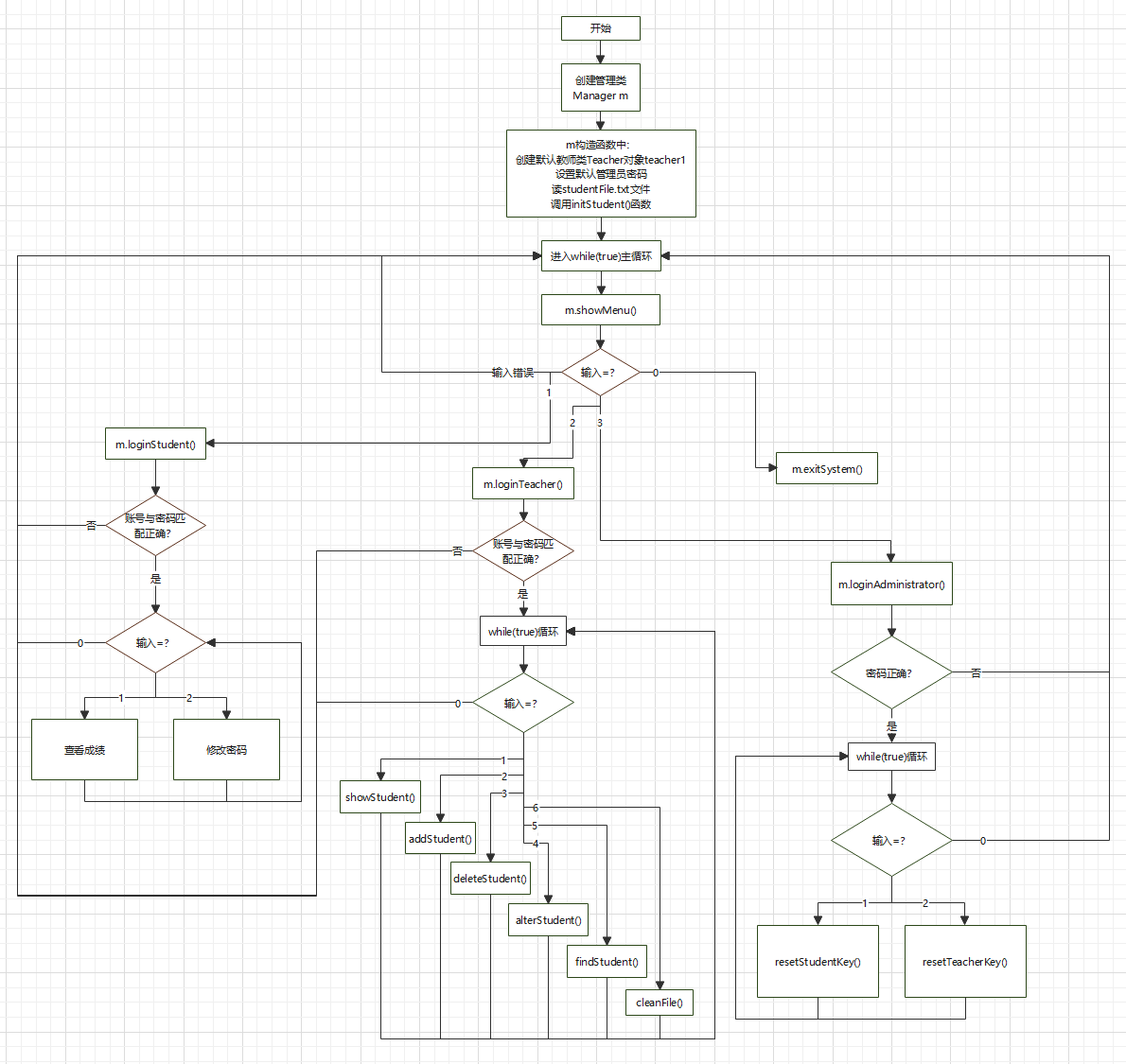


图 3.4-2 系统详细流程图

m调用loginTeacher()函数，教师使用账号和密码登录。若账号存在且密码正确后，正式进入教师端，此时进入一个新的while(true)循环，并等待用户输入。若用户输入“1”，系统调用showStudent()函数，系统可显示学生信息供教师查看；若用户输入“2”，系统调用addStudent()函数，教师可按系统提示完成添加学生信息操作；若用户输入“3”，系统调用deleteStudent()函数，教师可按系统提示完成删除学生信息操作；若用户输入“4”，系统调用alterStudent()函数，教师可按系统提示完成修改学生信息操作；若用户输入“5”，系统调用findStudent()函数，教师可按系统提示完成查找学生信息操作；若用户输入“6”，系统调用cleanFile()函数，教师可按系统提示完成清空学生信息操作。以上六项操作结束后，系统都将回到教师端的界面。若用户输入“0”，系统将返回上一级菜单，即回到主界面while(true)循环。若用户在登录时，账号不存在或密码错误，系统也将回到主界面。

m调用loginAdministrator()函数，管理员输入管理员操作密码登录。若密码输入正确后，正式进入管理员操作端，此时进入一个新的while(true)循环，并等待用户输入。若用户输入“1”，系统调用resetStudentKey()函数，管理员可按系统提示完成重置学生密码操作；若用户输入“2”，系统调用resetTeacherKey()函数，管理员可按系统提示完成重置教师密码操作。以上两项操作结束后，系统都将回到管理员端的界面。若用户输入“0”，系统将返回上一级菜单，即回到主界面while(true)循环。若用户在登录时，输入密码错误，系统也将回到主界面。

# 系统界面设计说明

## 界面列表

程序主界面，如图4.1-1所示。

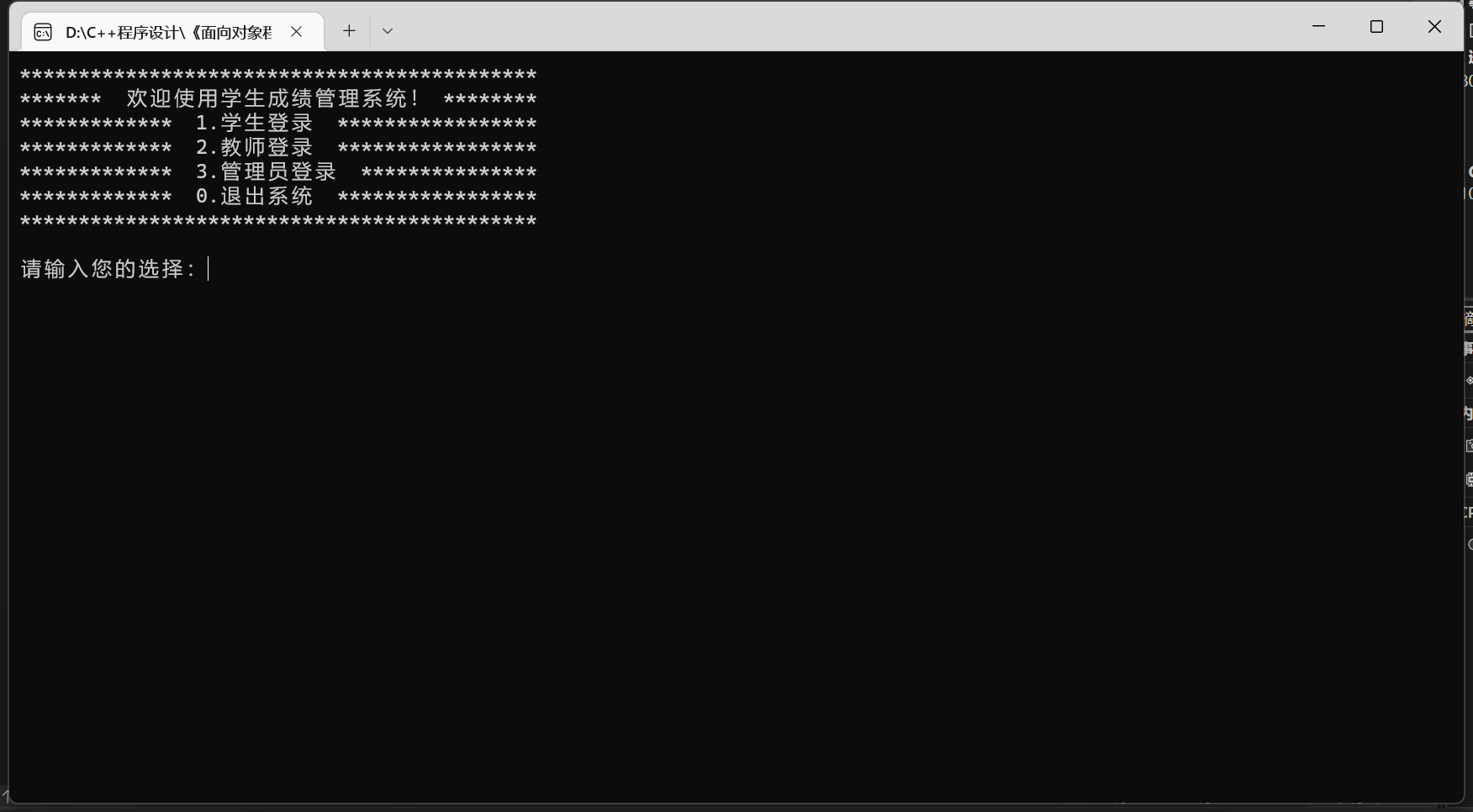


图 4.1-1 程序主界面

二级学生登录菜单界面，如图4.1-2所示。

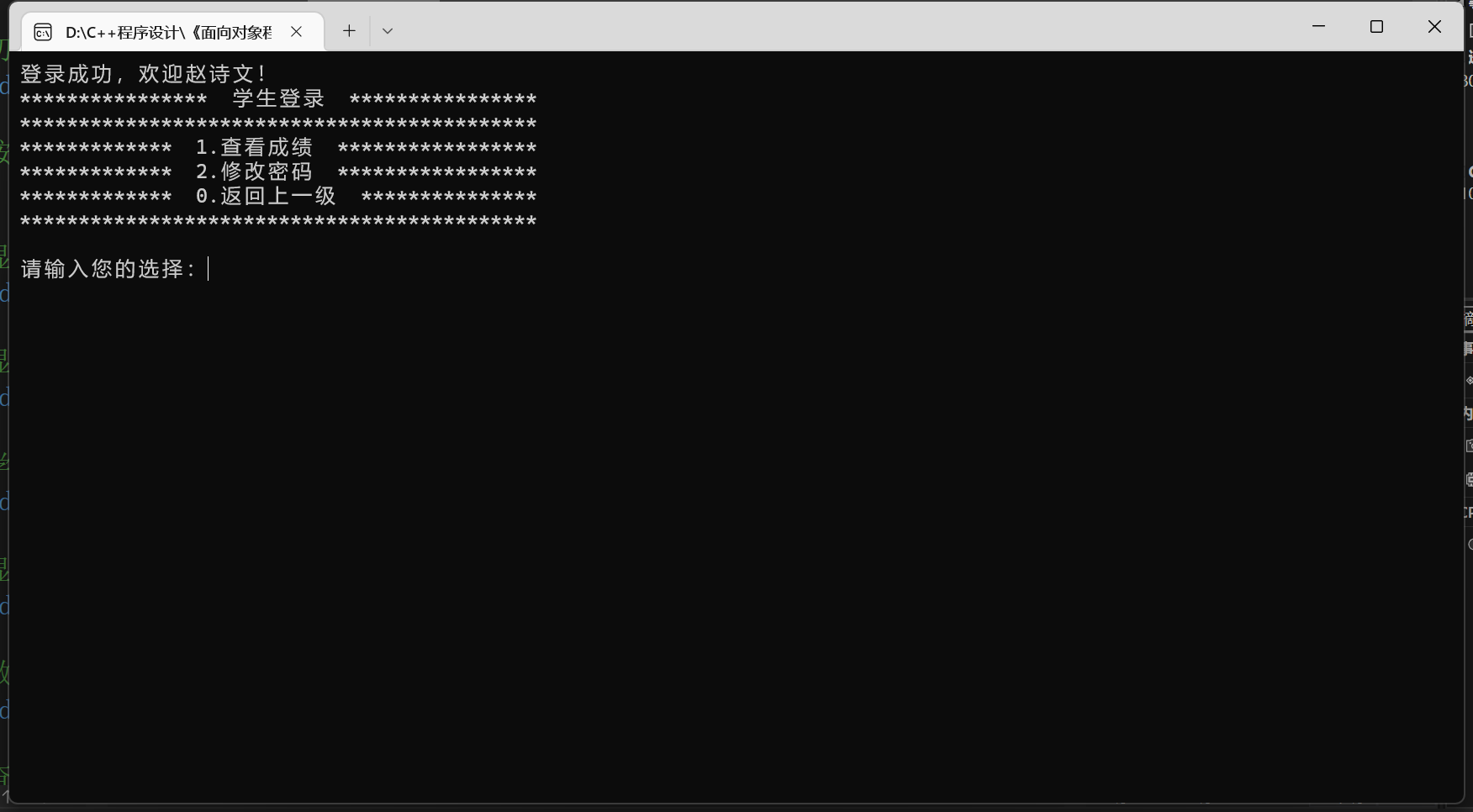


图 4.1-2 二级学生登录菜单界面

二级教师登录菜单界面，如图4.1-3所示。

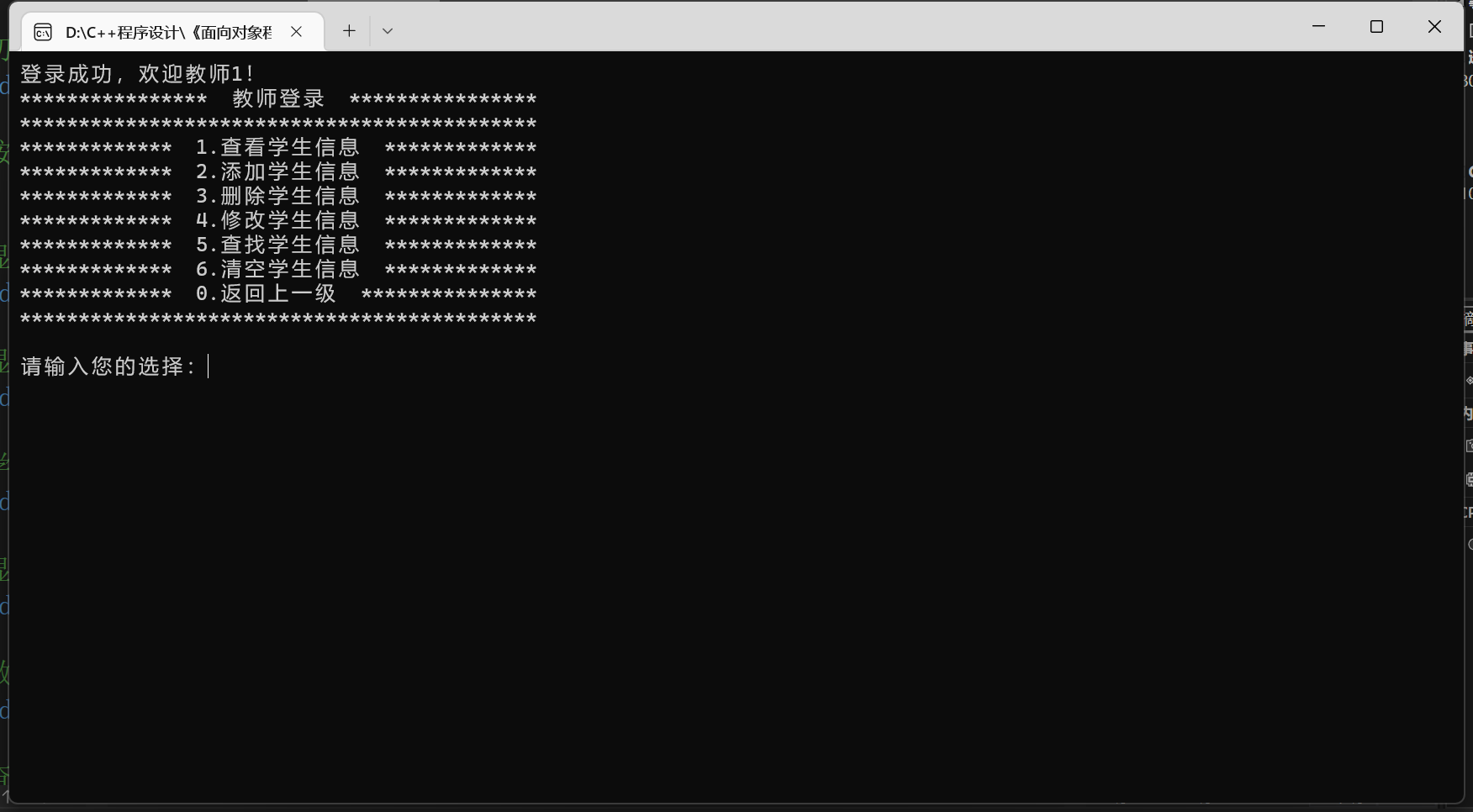


图 4.1-3 二级教师登录菜单界面

二级管理员登录菜单界面，如图4.1-4所示。

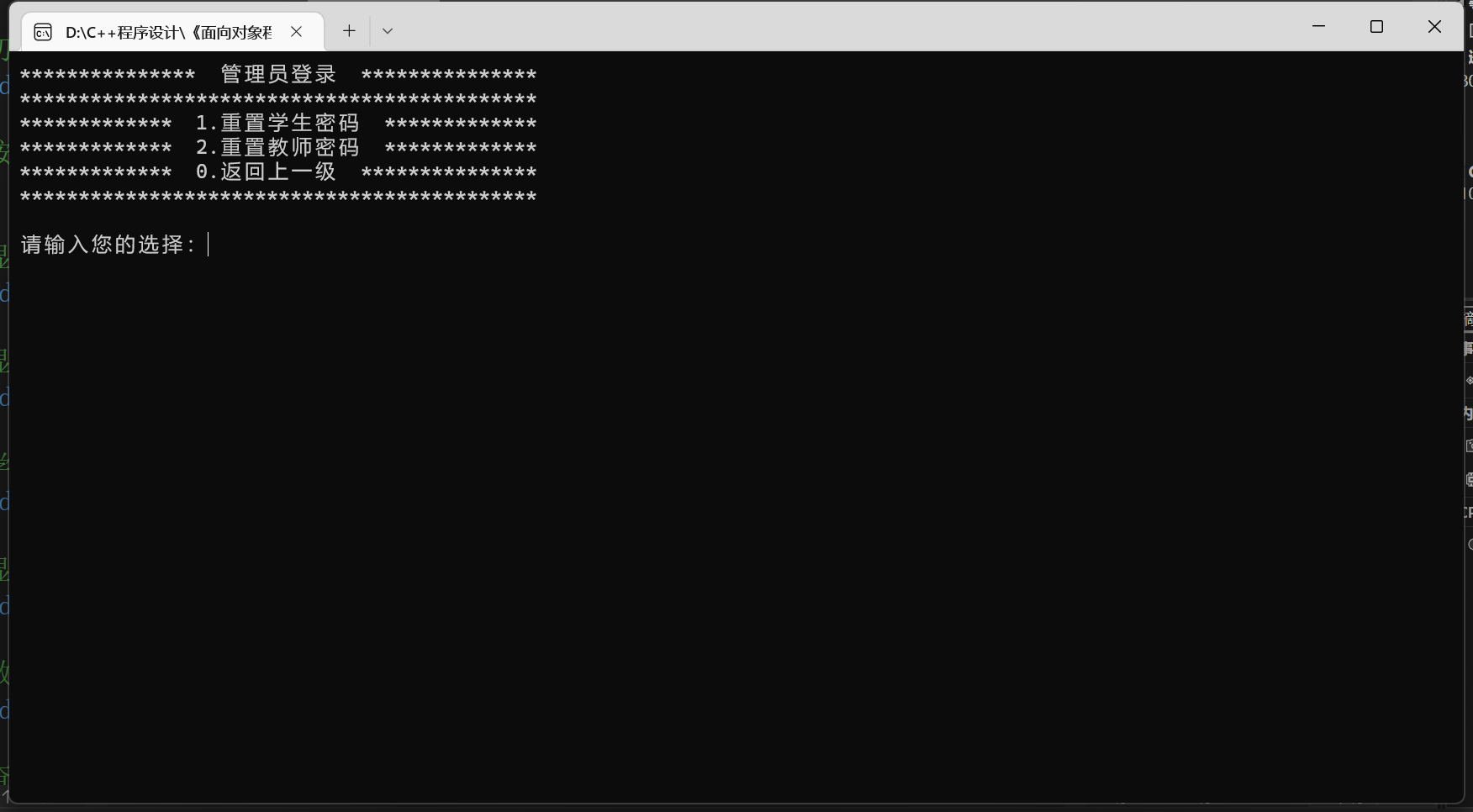


图 4.1-3 二级管理员登录菜单界面

## 各界面中主要变量、函数

本系统基于C++语言，使用面向对象程序设计方法设计。整个程序的主体类设计如图4.2-1所示。其中“Person”类（见表4.2-1）为一个抽象类，将学生“Student”类（见表4.2-2）和教师“Teacher”（见表4.2-3）类抽象到一个类内，利用多态管理。同时，创建“Manager”类，以管理程序运行。

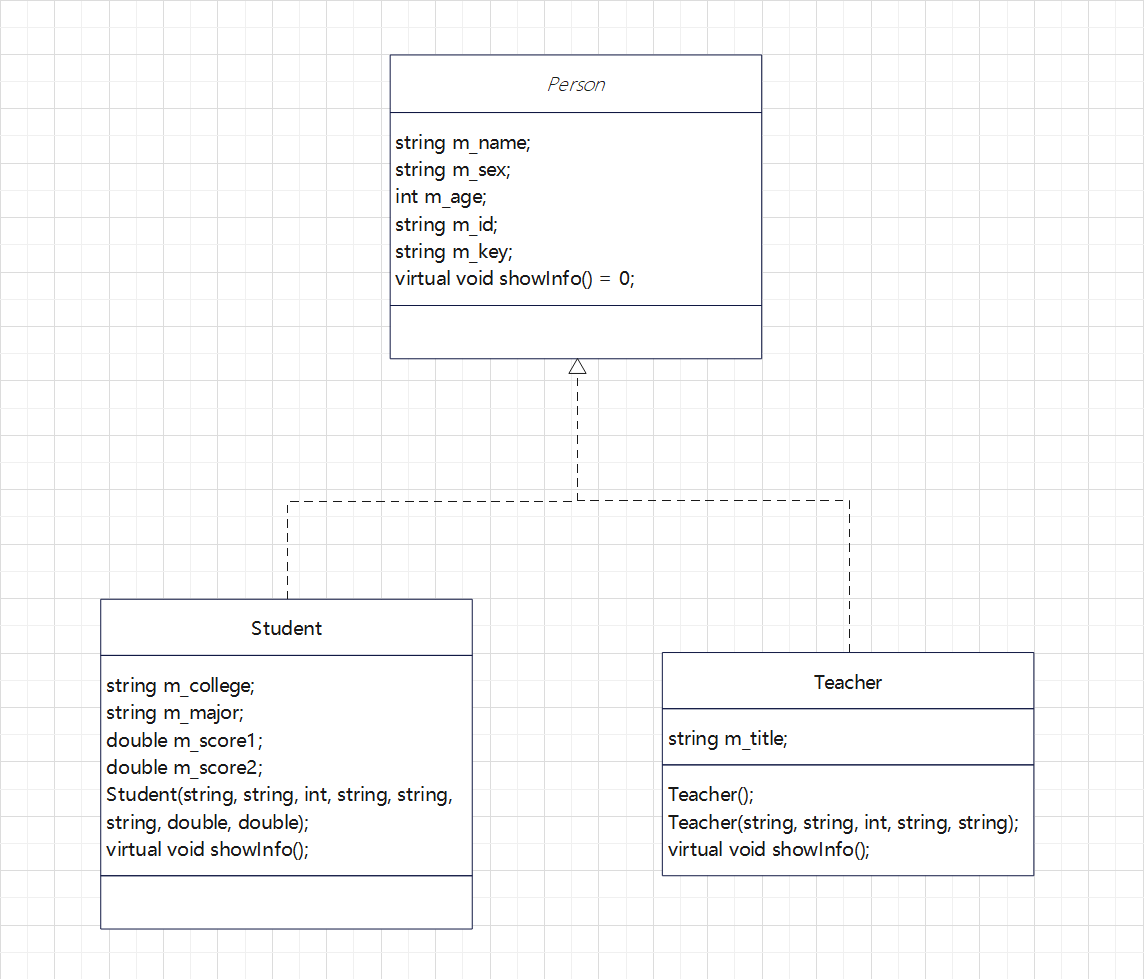


图 4.2-1 主体类设计

### “Person”类

|  |  |  |
| --- | --- | --- |
| **关键字** | **描述** | **数据类型** |
| m\_name | 姓名 | string |
| m\_sex | 性别 | string |
| m\_age | 年龄 | int |
| m\_id | 账号 | string |
| m\_key | 密码 | string |

表 4.2-1 Person类属性表

“Person”类中有一个void showInfo()函数，为纯虚函数。

### “Student”类

|  |  |  |
| --- | --- | --- |
| **关键字** | **描述** | **数据类型** |
| m\_name | 姓名 | string |
| m\_sex | 性别 | string |
| m\_age | 年龄 | int |
| m\_id | 账号 | string |
| m\_key | 密码 | string |
| m\_college | 学院 | string |
| m\_major | 专业 | string |
| m\_score1 | 分数1 | double |
| m\_score2 | 分数2 | double |

表 4.2-2 Student类属性表

“Student”类中有构造函数，并将“Person”类中的showInfo()函数重写。

Student::Student(string name, string sex, int age, string id, string college, string major, double score1, double score2) {

m\_name = name;

m\_sex = sex;

m\_age = age;

m\_id = id;

m\_key = "123456";

m\_college = college;

m\_major = major;

m\_score1 = score1;

m\_score2 = score2;

}

void Student::showInfo() {

cout << "姓名: " << m\_name << "\t";

cout << "性别: " << m\_sex << "\t";

cout << "年龄: " << m\_age << "\t";

cout << "学号: " << m\_id << "\t";

cout << "学院: " << m\_college << "\t";

cout << "专业: " << m\_major << "\t";

cout << "分数1: " << m\_score1 << "\t";

cout << "分数2: " << m\_score2 << endl;

}

### “Teacher”类

|  |  |  |
| --- | --- | --- |
| **关键字** | **描述** | **数据类型** |
| m\_name | 姓名 | string |
| m\_sex | 性别 | string |
| m\_age | 年龄 | int |
| m\_id | 账号 | string |
| m\_key | 密码 | string |
| m\_title | 职位 | string |

表 4.2-3 Teacher类属性表

“Teacher”类中有构造函数，还有一个含有默认值的构造函数，并将“Person”类中的showInfo()函数重写。

Teacher::Teacher() {

m\_name = "教师";

m\_sex = "未知";

m\_age = 0;

m\_id = "123456";

m\_key = "123456";

m\_title = "未知";

}

Teacher::Teacher(string name, string sex, int age, string id, string title) {

m\_name = name;

m\_sex = sex;

m\_age = age;

m\_id = id;

m\_key = "123456";

m\_title = title;

}

void Teacher::showInfo() {

cout << "姓名: " << m\_name << "\t";

cout << "性别: " << m\_sex << "\t";

cout << "年龄: " << m\_age << "\t";

cout << "工号: " << m\_id << "\t";

cout << "职位: " << m\_title << endl;

}

### “Manager”类

#### 数据成员

“Manager”类包含5个数据成员，详细见表4.2.4-1。

|  |  |  |
| --- | --- | --- |
| **关键字** | **描述** | **数据类型** |
| teacher1 | 默认教师 | Teacher |
| keyAdministrator | 默认管理员操作密码 | string |
| m\_stuNum | 记录文件中的人数个数 | int |
| m\_student | 学生数组的指针 | Student\*\* |
| m\_FileIsEmpty | 标志文件是否为空 | bool |

表 4.2.4-1 “Manager”类属性表

#### 成员函数

“Manager”类包含23个成员函数，详细见表4.2.4-2。

|  |  |  |
| --- | --- | --- |
| **函数名** | **描述** | **函数类型** |
| Manager() | “Manager”类构造函数 | / |
| ~Manager() | “Manager”类析构函数 | / |
| get\_stuNum() | 统计文件内记录学生人数函数 | int |
| initStudent() | 初始化学生信息函数 | void |
| isExist(string id) | 按照学号判断学生是否存在函数 | int |
| showMenu() | 显示一级菜单函数 | void |
| showMenuStudent() | 显示二级学生登录菜单函数 | void |
| loginStudent() | 学生登录函数 | void |
| showMenuTeacher() | 显示二级教师登录菜单函数 | void |
| loginTeacher() | 教师登录函数 | void |
| showStudent() | 查看学生信息函数 | void |
| addStudent() | 添加学生信息函数 | void |
| deleteStudent() | 删除学生信息函数 | void |
| alterStudent() | 修改学生信息函数 | void |
| findStudent() | 查找学生信息函数 | void |
| sortStudent() | 排序学生信息函数 | void |
| saveFile() | 保存文件函数 | void |
| cleanFile() | 保存文件函数 | void |
| showMenuAdministrator() | 显示二级管理员登录菜单函数 | void |
| oginAdministrator() | 管理员登录函数 | void |
| resetStudentKey() | 管理员重置学生密码函数 | void |
| resetTeacherKey() | 管理员重置学生密码函数 | void |
| exitSystem() | 退出系统函数 | void |

表 4.2.4-2 “Manager”类成员函数表

##### Manager()构造函数

在Manager()构造函数中，系统将创建一个默认Teacher类对象：teacher1；设置默认管理员密码；读studentFile.txt文件，若文件不存在则将提示用户；调用initStudent()函数初始化学生对象。

Manager::Manager() {

teacher1 = { "教师1", "男", 29, "1234567", "教授" };

keyAdministrator = "123456";

ifstream ifs;

ifs.open(FILENAME, ios::in);

//文件不存在情况

if (!ifs.is\_open()){

cout << "文件不存在！" << endl; //测试输出

m\_stuNum = 0; //初始化人数

m\_FileIsEmpty = true; //初始化文件为空标志

m\_student = NULL; //初始化数组

ifs.close(); //关闭文件

return;

}

//文件存在，但没有记录

char ch;

ifs >> ch;

if (ifs.eof()){

cout << "文件为空!" << endl;

m\_stuNum = 0;

m\_FileIsEmpty = true;

m\_student = NULL;

ifs.close();

return;

}

int num = get\_stuNum();

m\_stuNum = num; //更新成员属性

m\_student = new Student \* [m\_stuNum];//根据学生数创建数组

initStudent();//初始化学生

}

##### ~Manager()析构函数

在析构函数中，系统将释放堆区数据。

Manager::~Manager() {

if (m\_student != NULL) {

delete[] this->m\_student;

}

}

##### get\_stuNum()函数

在get\_stuNum()函数中，系统将读studentFile.txt文件，根据文件内容计算文件内包含的学生个数，以便创建相对于大小的空间存放数据。

int Manager::get\_stuNum(){

ifstream ifs;

ifs.open(FILENAME, ios::in);

string name;

string sex;

int age;

string id;

string college;

string major;

double score1;

double score2;

int num = 0;

while (ifs >> name && ifs >> sex && ifs >> age && ifs >> id && ifs >> college && ifs >> major && ifs >> score1 && ifs >> score2){

num++;//记录人数

}

ifs.close();

return num;

}

##### initStudent()函数

读取文件studentFile.txt，将文件上内容存放到数组中。

void Manager::initStudent(){

ifstream ifs;

ifs.open(FILENAME, ios::in);

string name;

string sex;

int age;

string id;

string college;

string major;

double score1;

double score2;

int index = 0;

while (ifs >> name && ifs >> sex && ifs >> age && ifs >> id && ifs >> college && ifs >> major && ifs >> score1 && ifs >> score2){

Student\* student = NULL;

student = new Student(name, sex, age, id, college, major, score1, score2);//创建不同对象

m\_student[index] = student;//存放在数组中

index++;

}

}

##### isExist(string id)函数

此函数目的是按照学号判断学生是否存在函数，是判断学生是否存在的模块。

int Manager::isExist(string id) {

int index = -1;

for (int i = 0; i < m\_stuNum; i++) {

if (m\_student[i]->m\_id == id) {

index = i;

break;

}

}

return index;

}

##### showMenu()函数

显示一级菜单函数。

void Manager::showMenu() {

system("cls");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\* 欢迎使用学生成绩管理系统！ \*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 1.学生登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 2.教师登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 3.管理员登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 0.退出系统 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

}

##### showMenuStudent()函数

显示二级学生登录菜单函数。

void Manager::showMenuStudent() {

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 学生登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 1.查看成绩 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 2.修改密码 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 0.返回上一级 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

}

##### loginStudent()函数

学生登录函数。

void Manager::loginStudent() {

system("cls");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 学生登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

string id;

cout << "请输入学号：";

cin >> id;

bool flag = false; //查找到的标志

for (int i = 0; i < m\_stuNum; i++) {

if (m\_student[i]->m\_id == id) {

flag = true;

string key;

cout << "请输入密码：";

cin >> key;

if (m\_student[i]->m\_key == key) {

system("cls");

int select;

while (true) {

system("cls");

cout << "登录成功，欢迎" << m\_student[i]->m\_name << "！" << endl;

showMenuStudent();

cout << "请输入您的选择：";

cin >> select;

if (select == 1) {

m\_student[i]->showInfo();

system("pause");

system("cls");

}

if (select == 2) {

while (true) {

string newKey1;

cout << "请输入新密码：";

cin >> newKey1;

if (newKey1 == m\_student[i]->m\_key) {

cout << "新密码与旧密码相同，修改密码失败！" << endl;

system("pause");

system("cls");

break;

}

string newKey2;

cout << "请再次确认密码：";

cin >> newKey2;

if (newKey1 == newKey2) {

if (newKey1 != m\_student[i]->m\_key) {

m\_student[i]->m\_key = newKey1;

cout << "修改密码成功！" << endl;

system("pause");

system("cls");

break;

}

}

else {

cout << "两次密码输入不同，修改密码失败！" << endl;

system("pause");

system("cls");

break;

}

}

}

if (select == 0) {

break;

}

}

}

else {

cout << "密码错误！" << endl;

system("pause");

}

}

}

if (flag == false) {

cout << "账号不存在或账号输入有误！" << endl;

system("pause");

}

}

##### showMenuTeacher()函数

显示二级教师登录菜单函数。

void Manager::showMenuTeacher() {

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 教师登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 1.查看学生信息 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 2.添加学生信息 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 3.删除学生信息 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 4.修改学生信息 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 5.查找学生信息 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 6.清空学生信息 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 0.返回上一级 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

}

##### loginTeacher()函数

教师登录函数。

void Manager::loginTeacher() {

system("cls");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 教师登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

string id;

cout << "请输入工号：";

cin >> id;

bool flag = false; //查找到的标志

if (teacher1.m\_id == id) {

flag = true;

string key;

cout << "请输入密码：";

cin >> key;

if (teacher1.m\_key == key) {

system("cls");

int choice = 0;

while (true) {

system("cls");

cout << "登录成功，欢迎" << teacher1.m\_name << "！" << endl;

showMenuTeacher();

cout << "请输入您的选择：";

cin >> choice;

//返回上一级

if (choice == 0)

break;

switch (choice) {

//查看学生信息

case 1:

showStudent();

break;

//添加学生信息

case 2:

addStudent();

break;

//删除学生信息

case 3:

deleteStudent();

break;

//修改学生信息

case 4:

alterStudent();

break;

//查找学生信息

case 5:

findStudent();

break;

//清空学生信息

case 6:

cleanFile();

break;

default:

system("cls");

break;

}

}

}

else {

cout << "密码错误！" << endl;

system("pause");

}

}

if (flag == false) {

cout << "账号不存在或账号输入有误！" << endl;

system("pause");

}

}

##### showStudent()函数

查看学生信息函数。

void Manager::showStudent(){

if (m\_FileIsEmpty){

cout << "文件不存在或记录为空！" << endl;

}

else{

for (int i = 0; i < m\_stuNum; i++){

m\_student[i]->showInfo();//利用多态调用接口

}

cout << endl;

cout << "是否选择排序查看（是--按1/否--按任意键）：";

int choice = 0;

cin >> choice;

if (choice == 1) {

sortStudent();

}

}

}

##### addStudent()函数

添加学生信息函数。

void Manager::addStudent() {

cout << "请输入增加学生数量：" ;

int addNum = 0;

cin >> addNum;

if (addNum > 0) {

int newSize = m\_stuNum + addNum;//计算新空间大小

Student\*\* newSpace = new Student\*;//开辟新空间

//将原空间下内容存放到新空间下

if (m\_student != NULL) {

for (int i = 0; i < m\_stuNum; i++) {

newSpace[i] = m\_student[i];

}

}

//输入新数据

for (int i = 0; i < addNum; i++) {

string name;

string sex;

int age;

string id;

string college;

string major;

double score1;

double score2;

cout << "请输入第" << i + 1 << "个学生的姓名：";

cin >> name;

cout << "请输入第" << i + 1 << "个学生的性别：";

cin >> sex;

cout << "请输入第" << i + 1 << "个学生的年龄：";

cin >> age;

cout << "请输入第 " << i + 1 << " 个学生的学号：";

cin >> id;

cout << "请输入第 " << i + 1 << " 个学生所属学院：";

cin >> college;

cout << "请输入第 " << i + 1 << " 个学生所在专业：";

cin >> major;

cout << "请输入第 " << i + 1 << " 个学生分数1的分数：";

cin >> score1;

cout << "请输入第 " << i + 1 << " 个学生分数2的分数：";

cin >> score2;

Student\* student = NULL;

student = new Student(name, sex, age, id, college, major, score1, score2);

newSpace[m\_stuNum + i] = student;

}

delete[] m\_student;//释放原有空间

m\_student = newSpace;//更改新空间的指向

m\_stuNum = newSize;//更新新的个数

saveFile();//保存数据

m\_FileIsEmpty = false;//更新职工不为空标志

cout << "成功添加" << addNum << "名同学！" << endl;//提示信息

}

else {

cout << "输入有误" << endl;

}

system("pause");

system("cls");

}

##### deleteStudent()函数

删除学生信息函数。

void Manager::deleteStudent() {

if (m\_FileIsEmpty){

cout << "文件不存在或记录为空！" << endl;

}

else {

//按学号删除

cout << "请输入想要删除的学生的学号：";

string id;

cin >> id;

int index = isExist(id);

//说明index上位置数据需要删除

if (index != -1) {

for (int i = index; i < m\_stuNum - 1; i++)

{

m\_student[i] = m\_student[i + 1];

}

m\_stuNum--;

saveFile(); //删除后数据同步到文件中

cout << "删除成功！" << endl;

}

else

cout << "删除失败，未找到该学生" << endl;

}

system("pause");

system("cls");

}

##### alterStudent()函数

修改学生信息函数。

void Manager::alterStudent() {

if (m\_FileIsEmpty){

cout << "文件不存在或记录为空！" << endl;

}

else{

cout << "请输入修改学生的学号：";

string id;

cin >> id;

int ret = isExist(id);

if (ret != -1){

//查找到编号的学生

cout << "查到" << id << "，学生：" << m\_student[ret]->m\_name << endl;

delete m\_student[ret];

string newName = "";

string newSex = "";

int newAge = 0;

string newId = "";

string newCollege = "";

string newMajor = "";

double newScore1 = 0;

double newScore2 = 0;

cout << "请输入修改后的姓名：";

cin >> newName;

cout << "请输入修改后的性别：";

cin >> newSex;

cout << "请输入修改后的年龄：";

cin >> newAge;

cout << "请输入修改后的学号：";

cin >> newId;

cout << "请输入修改后的学院：";

cin >> newCollege;

cout << "请输入修改后的专业：";

cin >> newMajor;

cout << "请输入修改后的分数1的分数：";

cin >> newScore1;

cout << "请输入修改后的分数2的分数：";

cin >> newScore2;

Student\* student = NULL;

student = new Student(newName, newSex, newAge, newId, newCollege, newMajor, newScore1, newScore2);

m\_student[ret] = student;

cout << "修改成功！" << endl;

saveFile();//保存到文件中

}

else{

cout << "修改失败，查无此人!" << endl;

}

}

system("pause");

system("cls");

}

##### findStudent()函数

查找学生信息函数。

void Manager::findStudent() {

if (m\_FileIsEmpty){

cout << "文件不存在或记录为空！" << endl;

}

else{

cout << "查找的方式：" << endl;

cout << "1、按学号查找" << endl;

cout << "2、按姓名查找" << endl;

int select = 0;

cout << "请选择：";

cin >> select;

//按学号查找

if (select == 1){

string id;

cout << "请输入查找的学生的学号：";

cin >> id;

int ret = isExist(id);

if (ret != -1){

cout << "查找成功！该学生信息如下：" << endl;

m\_student[ret]->showInfo();

}

else{

cout << "查找失败，查无此人" << endl;

}

}

//按姓名查找

else if (select == 2) {

string name;

cout << "请输入查找的姓名：";

cin >> name;

bool flag = false; //查找到的标志

for (int i = 0; i < m\_stuNum; i++){

if (m\_student[i]->m\_name == name){

cout << "查找成功!" << endl;

flag = true;

m\_student[i]->showInfo();

}

}

if (flag == false){

cout << "查找失败，查无此人" << endl;

}

}

else{

cout << "输入选项有误" << endl;

}

}

system("pause");

system("cls");

}

##### sortStudent()函数

排序学生信息函数。

void Manager::sortStudent() {

if (m\_FileIsEmpty){

cout << "文件不存在或记录为空！" << endl;

system("pause");

system("cls");

}

else{

cout << "请选择排序方式： " << endl;

cout << "1、按学号升序排序" << endl;

cout << "2、按学号降序排序" << endl;

cout << "3、按分数1升序排序" << endl;

cout << "4、按分数1降序排序" << endl;

cout << "5、按分数2升序排序" << endl;

cout << "6、按分数2降序排序" << endl;

cout << "请选择：";

int select = 0;

cin >> select;

for (int i = 0; i < m\_stuNum; i++){

int MinOrMax = i;

for (int j = i + 1; j < m\_stuNum; j++){

//学号降序

if (select == 2) {

if (m\_student[MinOrMax]->m\_id < m\_student[j]->m\_id){

MinOrMax = j;

}

}

//分数1升序

if (select == 3) {

if (m\_student[MinOrMax]->m\_score1 > m\_student[j]->m\_score1) {

MinOrMax = j;

}

}

//分数1降序

if (select == 4) {

if (m\_student[MinOrMax]->m\_score1 < m\_student[j]->m\_score1) {

MinOrMax = j;

}

}

//分数2升序

if (select == 5) {

if (m\_student[MinOrMax]->m\_score2 > m\_student[j]->m\_score2) {

MinOrMax = j;

}

}

//分数2降序

if (select == 6) {

if (m\_student[MinOrMax]->m\_score2 < m\_student[j]->m\_score2) {

MinOrMax = j;

}

}

//学号升序

else {

if (m\_student[MinOrMax]->m\_id > m\_student[j]->m\_id) {

MinOrMax = j;

}

}

}

if (i != MinOrMax){

Student\* temp = m\_student[i];

m\_student[i] = m\_student[MinOrMax];

m\_student[MinOrMax] = temp;

}

}

saveFile();

cout << "排序成功,排序后结果为：" << endl;

showStudent();

}

}

##### saveFile()函数

保存文件函数。

void Manager::saveFile() {

ofstream ofs;

ofs.open(FILENAME, ios::out);

for (int i = 0; i < this->m\_stuNum; i++){

ofs << m\_student[i]->m\_name << "\t"

<< m\_student[i]->m\_sex << "\t"

<< m\_student[i]->m\_age << "\t"

<< m\_student[i]->m\_id << "\t"

<< m\_student[i]->m\_college << "\t"

<< m\_student[i]->m\_major << "\t"

<< m\_student[i]->m\_score1 << "\t"

<< m\_student[i]->m\_score2 << endl;

}

ofs.close();

}

##### cleanFile()函数

清空文件函数。

void Manager::cleanFile() {

cout << "确认清空？" << endl;

cout << "1、确认" << endl;

cout << "2、返回" << endl;

int select = 0;

cin >> select;

if (select == 1){

//打开模式 ios::trunc 如果存在删除文件并重新创建

ofstream ofs(FILENAME, ios::trunc);

ofs.close();

if (m\_student != NULL){

for (int i = 0; i < this->m\_stuNum; i++){

if (m\_student[i] != NULL){

delete this->m\_student[i];

}

}

m\_stuNum = 0;

delete[] m\_student;

m\_student = NULL;

m\_FileIsEmpty = true;

}

cout << "清空成功！" << endl;

}

system("pause");

system("cls");

}

##### showMenuAdministrator()函数

显示二级管理员登录菜单函数。

void Manager::showMenuAdministrator() {

system("cls");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 管理员登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 1.重置学生密码 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 2.重置教师密码 \*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\* 0.返回上一级 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

}

##### loginAdministrator()函数

管理员登录函数。

void Manager::loginAdministrator() {

system("cls");

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 管理员登录 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

string key;

cout << "请输入管理员操作密码：";

cin >> key;

if (key == keyAdministrator) {

int choice = 0;

while (true) {

showMenuAdministrator();

cout << "请输入您的选择：";

cin >> choice;

//返回上一级

if (choice == 0)

break;

switch (choice) {

//重置学生密码

case 1:

resetStudentKey();

break;

//重置教师密码

case 2:

resetTeacherKey();

break;

default:

system("cls");

break;

}

}

}

else {

cout << "密码错误！" << endl;

system("pause");

}

}

##### resetStudentKey()函数

管理员重置学生密码函数。

void Manager::resetStudentKey() {

if (m\_FileIsEmpty) {

cout << "文件不存在或记录为空！" << endl;

}

else {

cout << "请输入重置密码的学生的学号：" << endl;

string id;

cin >> id;

int ret = isExist(id);

if (ret != -1) {

//查找到编号的学生

cout << "查到" << id << "，学生：" << m\_student[ret]->m\_name << endl;

m\_student[ret]->m\_key = "123456";

cout << "重置密码成功！新密码：" << m\_student[ret]->m\_key << endl;

}

else {

cout << "重置失败，查无此人!" << endl;

}

}

system("pause");

system("cls")

}

##### resetTeacherKey()函数

管理员重置学生密码函数。

void Manager::resetTeacherKey() {

if (m\_FileIsEmpty) {

cout << "文件不存在或记录为空！" << endl;

}

else {

string id;

cout << "请输入教师工号：";

cin >> id;

if (teacher1.m\_id == id) {

teacher1.m\_key = "123456";

cout << "重置密码成功！新密码" << teacher1.m\_key << endl;

system("pause");

}

else {

cout << "重置失败，查无此人!" << endl;

system("pause");

}

}

system("cls");

}

##### exitSystem()函数

退出系统函数。

void Manager::exitSystem() {

cout << "欢迎下次使用!" << endl;

system("pause");

exit(0);

}

## 主要功能的核心源程序

### 程序预处理

包括加载头文件、定义常量。

#include <iostream>

#include "Manager.h"

using namespace std;

宏常量在Manager.h文件中定义。

#define FILENAME "studentFile.txt"

### 主函数main()

main()函数主要实现了对整个程序的运行控制，以及相关功能模块的调用。详细工作分析请参考图3.4-1。

int main(){

Manager m;

int choice = 0;

while (true){

m.showMenu();//展示菜单

cout << "请输入您的选择：";

cin >> choice;

switch (choice){

case 0: //退出系统

m.exitSystem();

break;

case 1: //学生登录

m.loginStudent();

break;

case 2: //教师登录

m.loginTeacher();

break;

case 3://管理员登录

m.loginAdministrator();

break;

default:

cout << "输入有误！" << endl;

system("pause");

break;

}

}

system("pause");

return 0;

}

## 运行结果

### 系统初始状态

当用户刚进入系统时，其初始状体如图4.4-1所示。此时，系统等待用户从键盘上输入有效数字进入二级菜单。

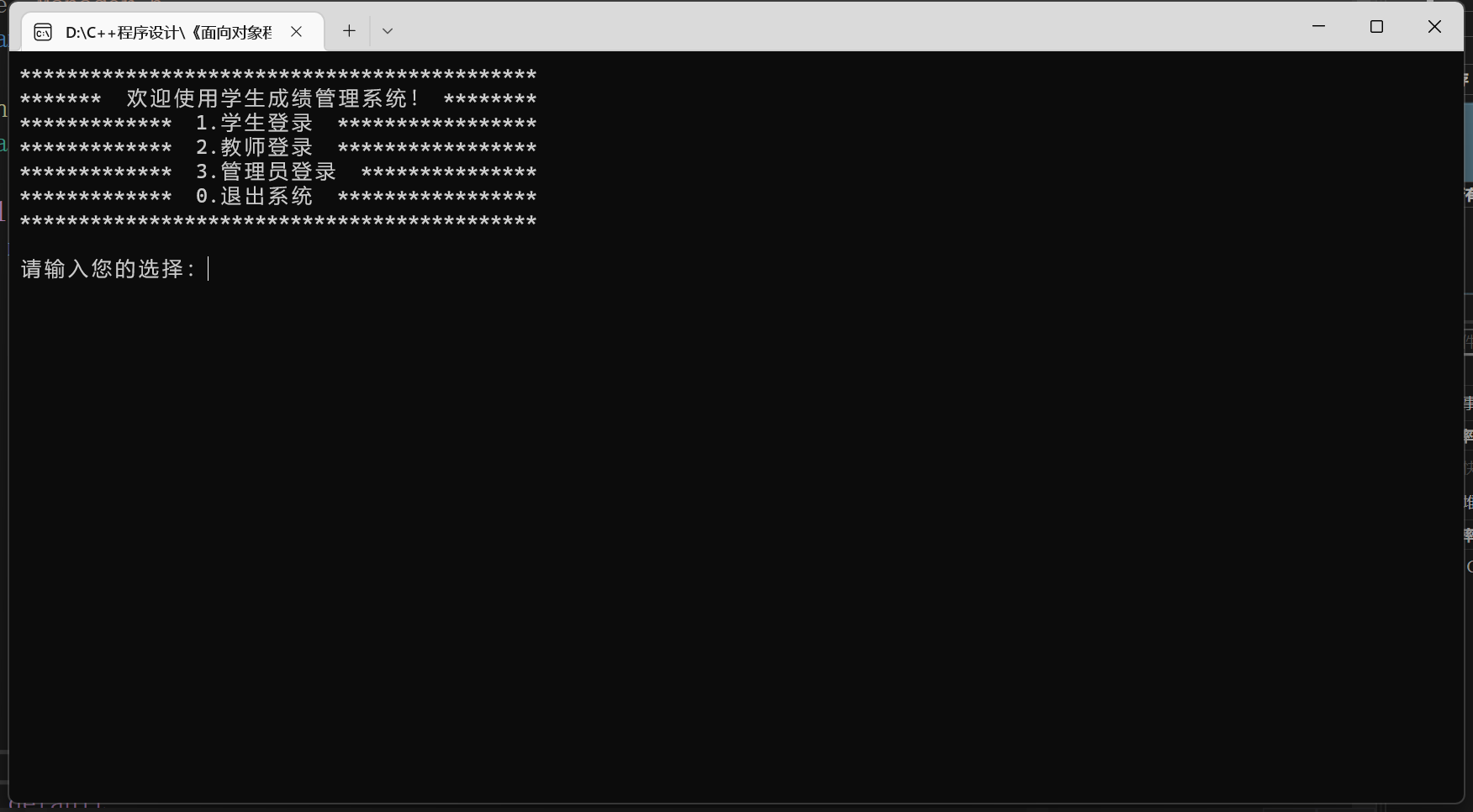


图 4.4-1 系统初始状态

### 学生登录

当用户在主界面输入“1”时，系统进入学生登录界面。首先要输入账号和密码，如图4.4-2所示。

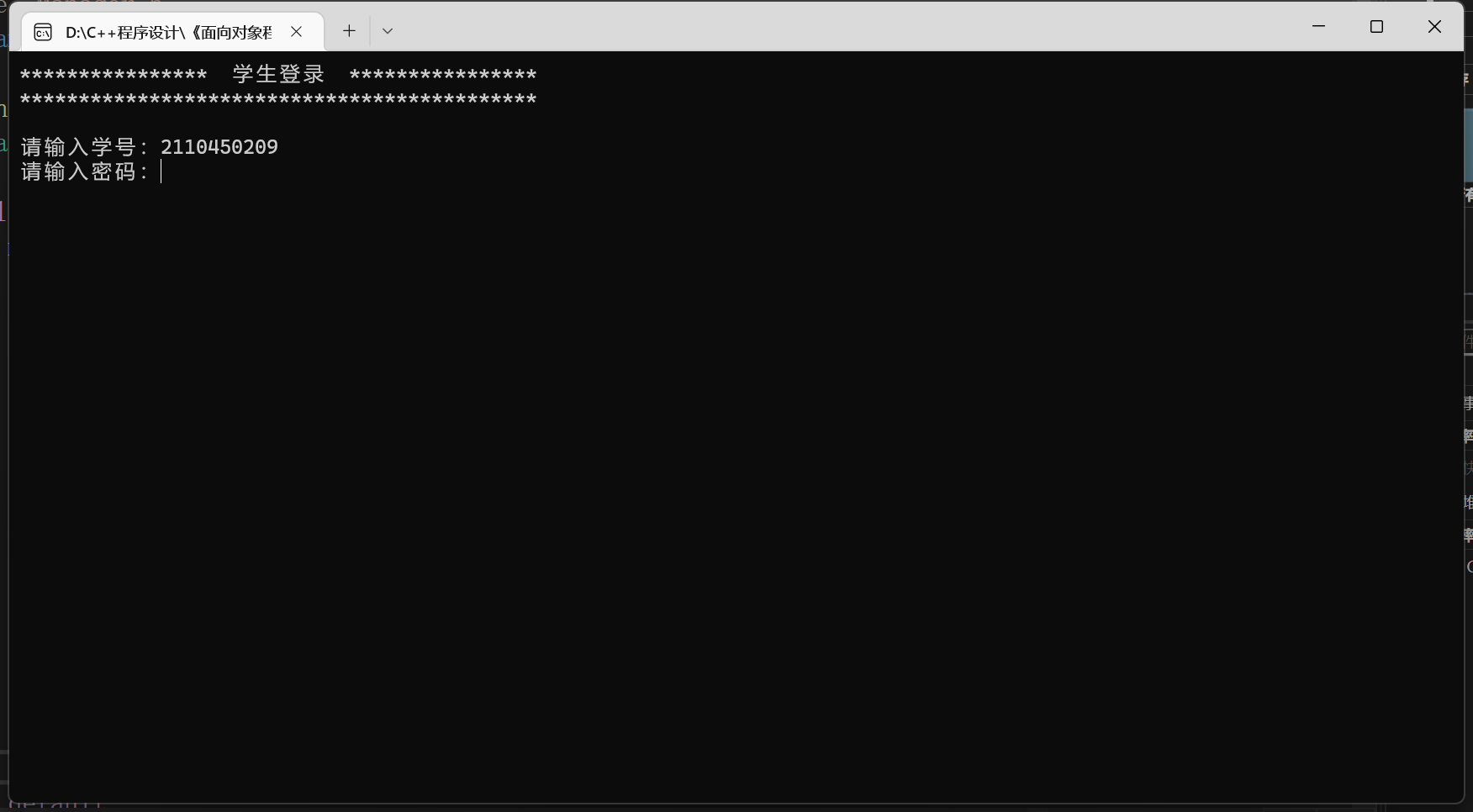


图 4.4-2 学生登录界面输入账号和密码

若账号不存在或者密码输入错误时，系统将提示错误，如图4.4-3(1)和图4.4-3(2)所示，并回到主界面。当账号和密码输入正确后，系统显示二级学生登录界面，如图4.4-4所示。此时，系统等待用户输入。

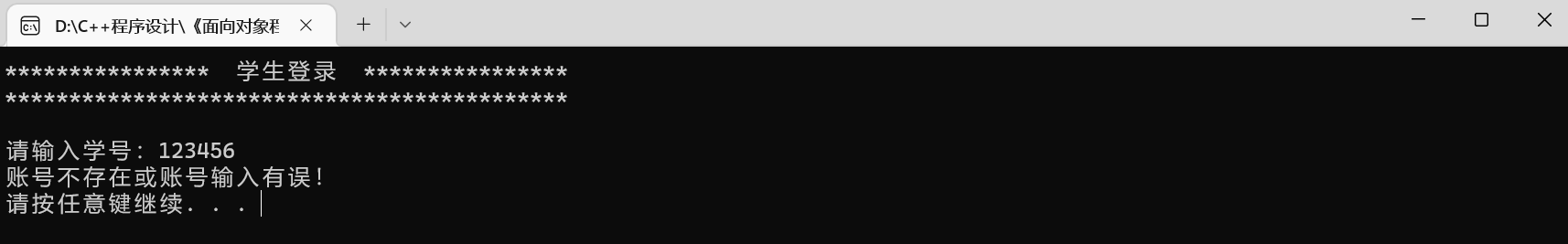


图 4.4-3(1) 学生登录账号不存在或输入有误



图 4.4-3(2) 学生登录账号输入密码错误

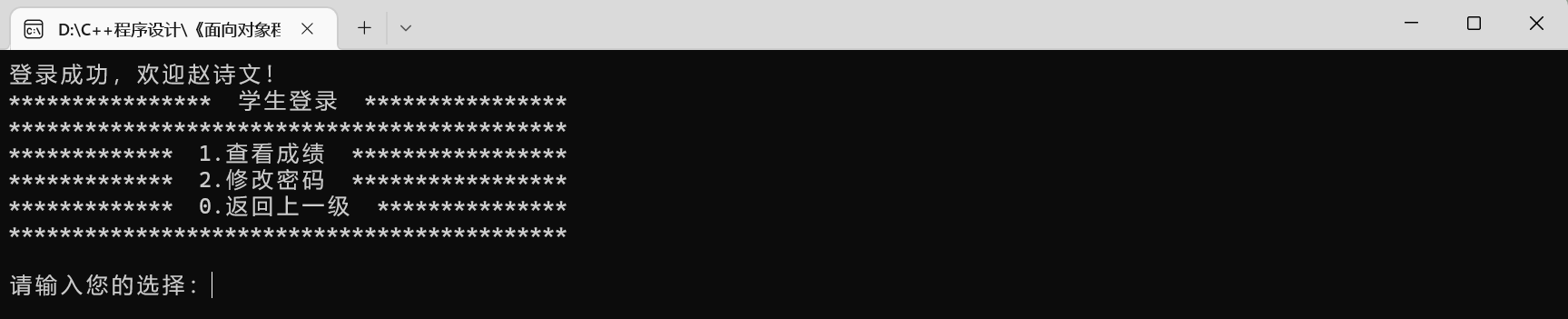


图 4.4-4 学生登录界面

用户输入“1”为查看成绩，如图4.4-5所示。学生查看完成绩后按任意键继续将回到学生登录界面。



图 4.4-5 学生查看成绩

用户输入“2”为修改密码，此时用户根据系统提示修改密码即可，如图4.4-6所示。修改密码成功后按任意键继续将回到学生登录界面。



图 4.4-6 学生修改密码成功

若输入的两次的新密码不同，系统将提示修改密码失败。如果新密码与旧密码相同也无法完成修改密码操作，如图4.4-7和图4.4-8所示以上提示后，系统都将回到学生端的界面。

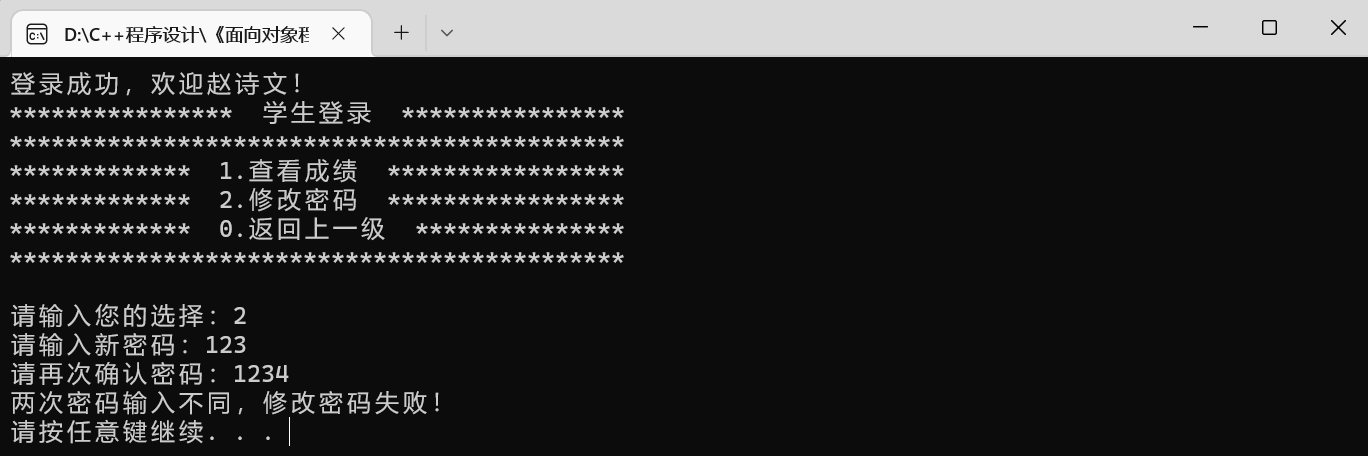


图 4.4-7 两次输入新密码不同修改密码失败

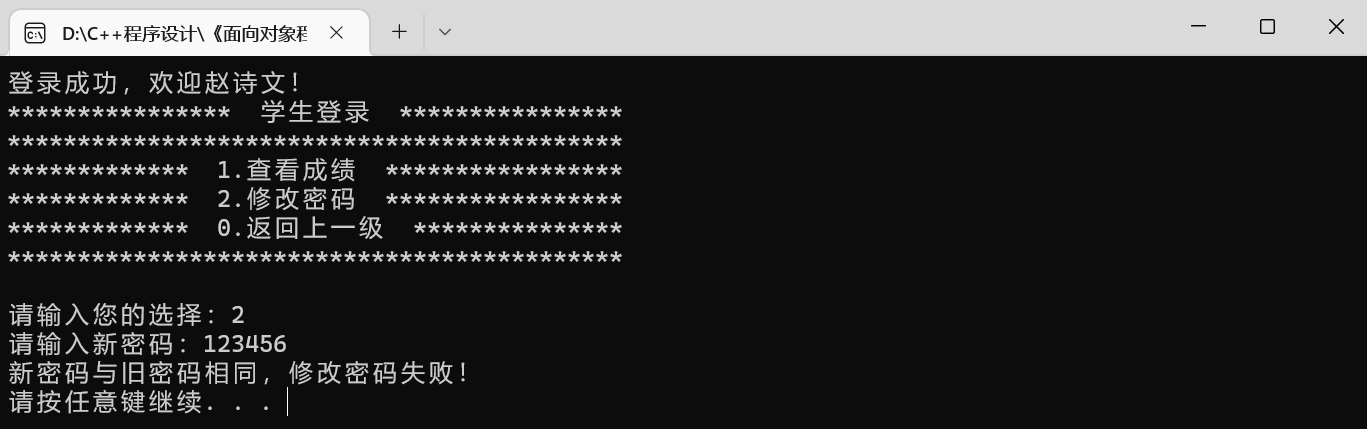


图 4.4-8 新旧密码相同修改密码失败

用户输入“0”则将返回上一级，即主界面。若用户输入无效字符，系统将直接回到学生登录界面重新等待用户输入。

### 教师登录

用户在主界面输入“2”，进入教师登录界面，如图4.4-9所示。系统提示输入账号和密码，若密码正确，系统显示二级教师登录界面，等待用户继续输入。如图4.4-10所示。若账号不存在或者密码输入错误时，系统将提示错误，如图4.4-11(1)和图4.4-11(2)所示，并回到主界面。

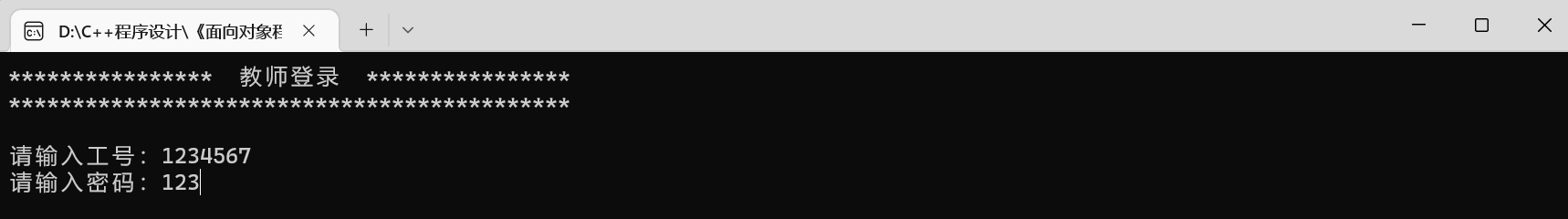


图 4.4-9 教师登录界面输入账号和密码

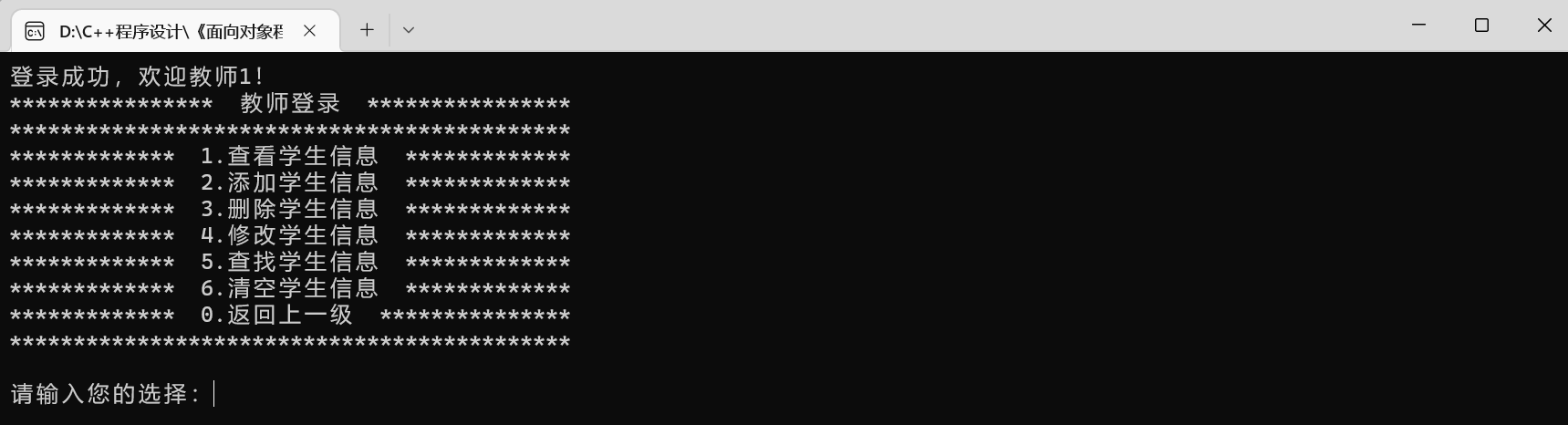


图 4.4-10 教师登录界面

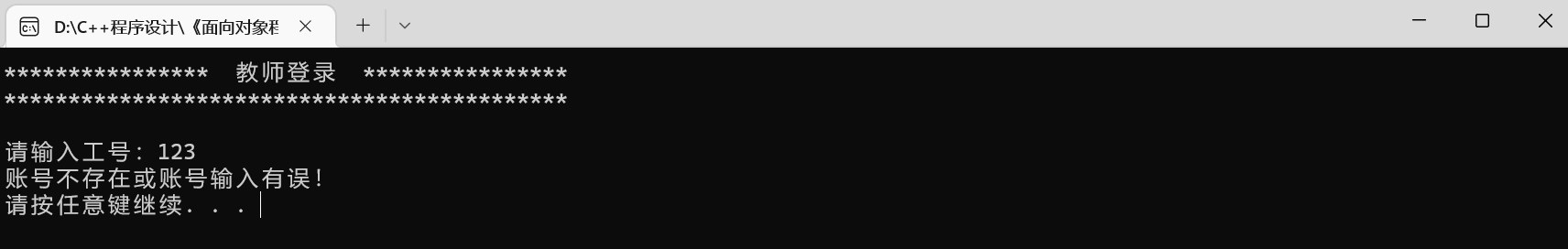


图 4.4-11(1) 教师登录账号不存在或输入有误

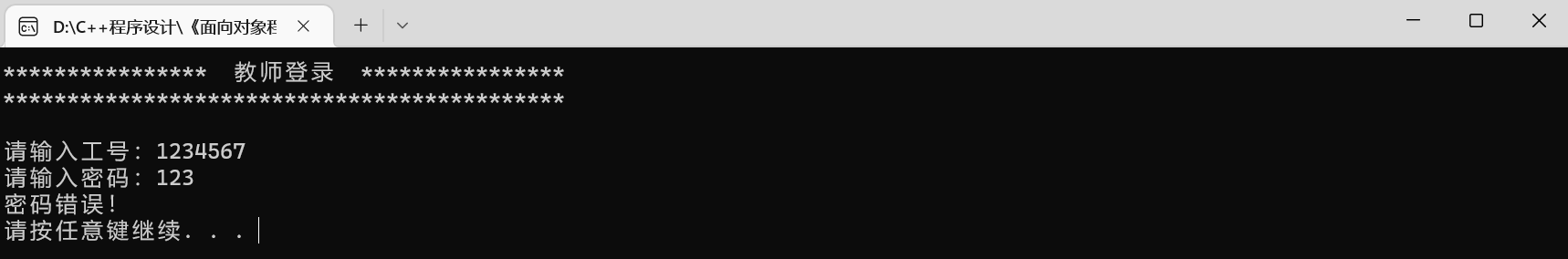


图 4.4-11(2) 教师登录账号输入密码错误

用户继续输入“1”，系统将进入查看学生信息模块，如图4.4-11所示。信息显示后，用户可以根据系统提示选择是否排序查看成绩，若输入“1”代表“是”，系统将进入排序查看模块，此时用户可以再根据提示选择排序方式，如图4.4-12所示。若不选择排序，则按任意键系统返回教师登录模块。

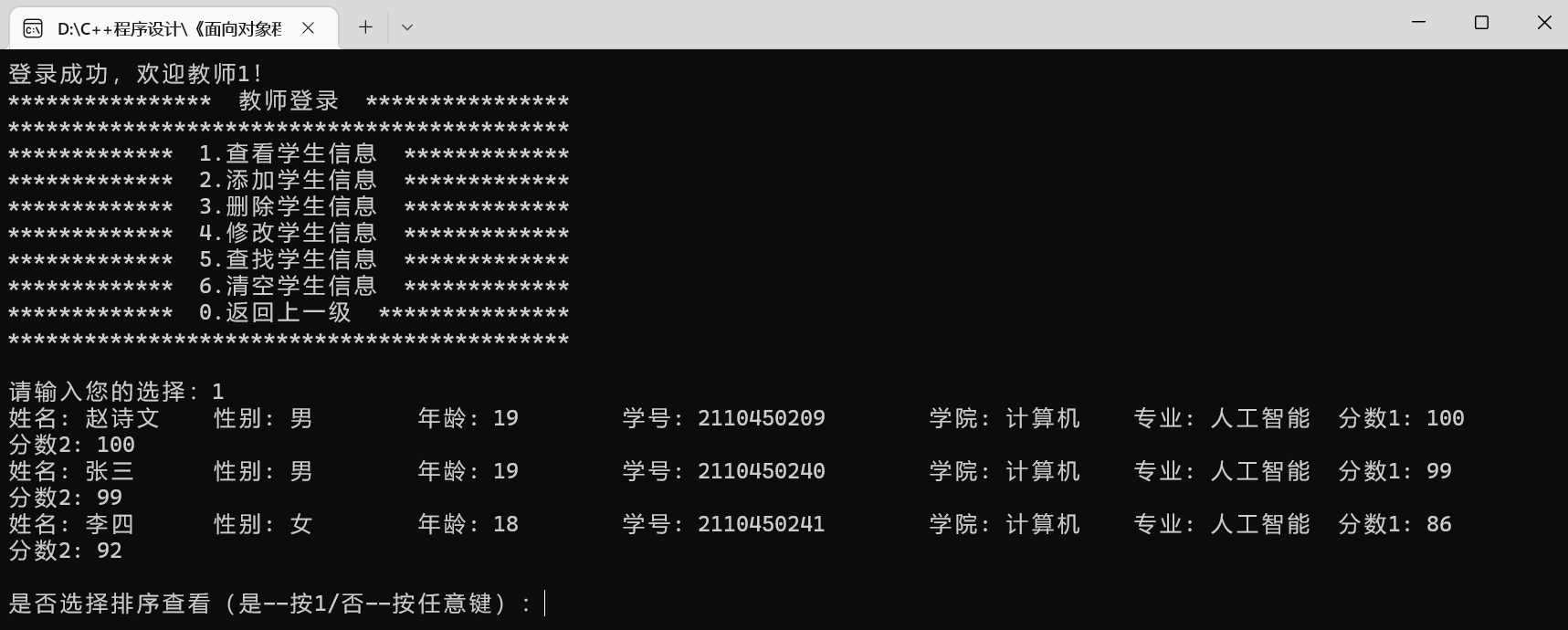


图 4.4-11 教师查看学生信息

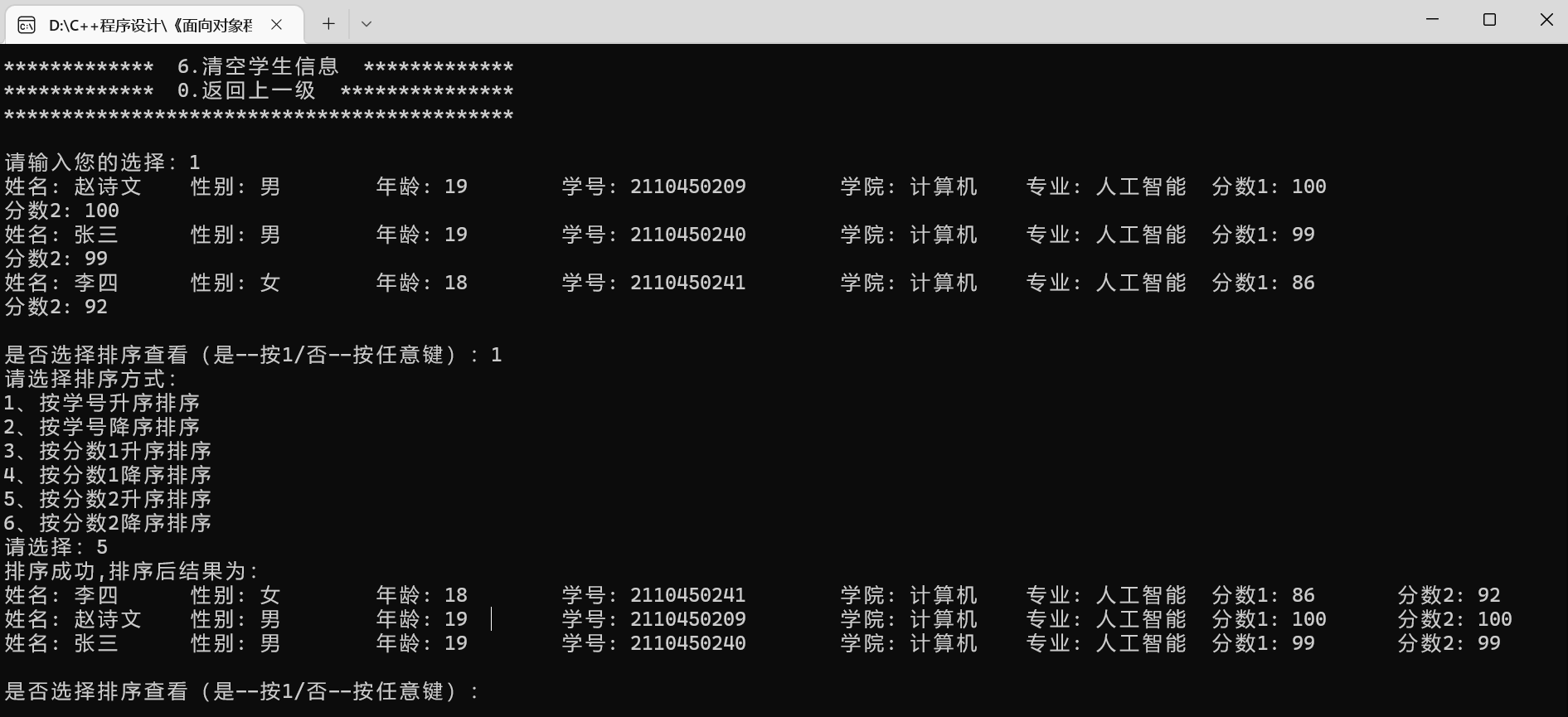


图 4.4-12 选择排序查看成绩

用户输入“2”，系统进入添加学生信息模块，用户可根据提示添加学生信息，如图4.4-13所示。添加学生王五信息成功后，系统将返回教师登录模块。

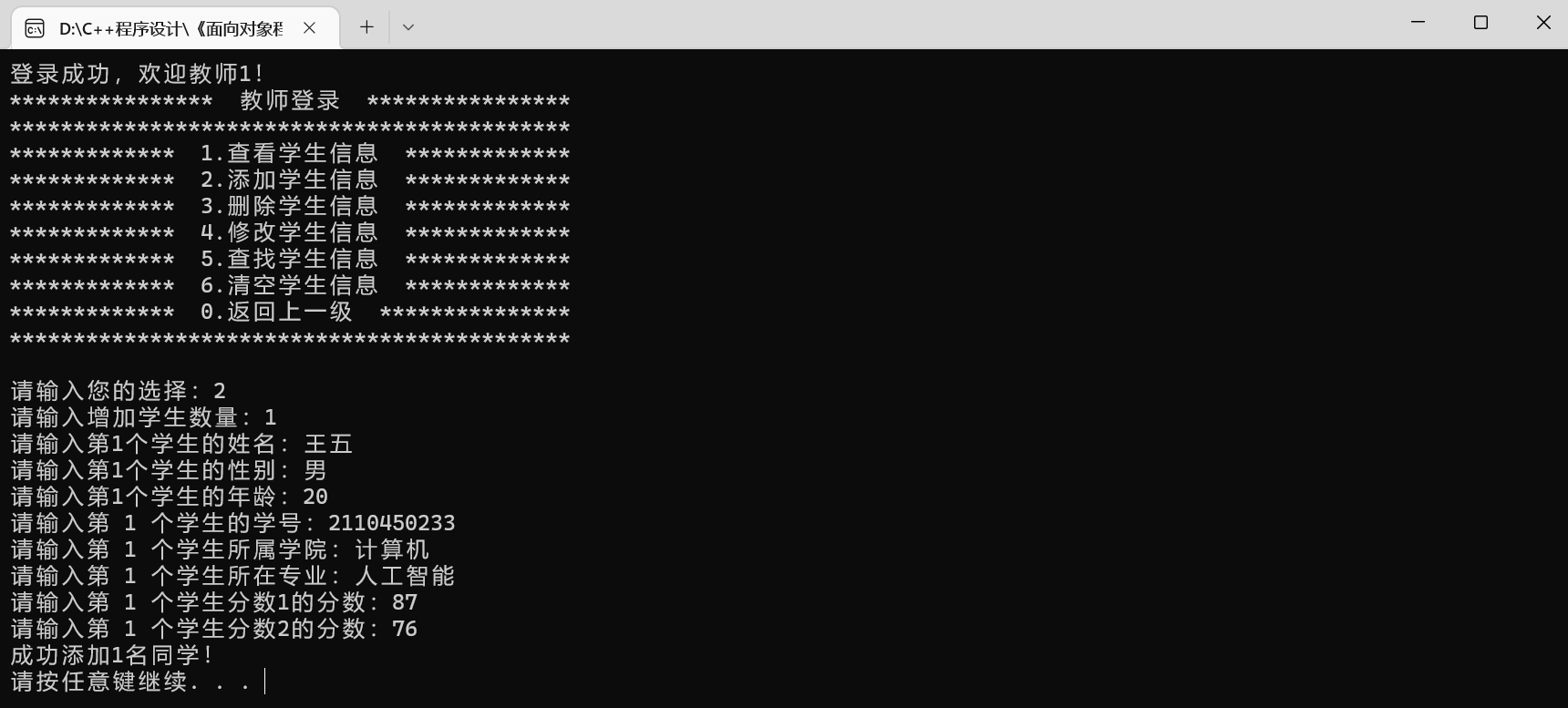


图 4.4-13 教师添加学生信息模块

用户输入“3”，系统进入删除学生信息模块，用户可根据提示删除学生信息，如图4.4-14所示。删除学生信息成功后，系统将返回教师登录模块。

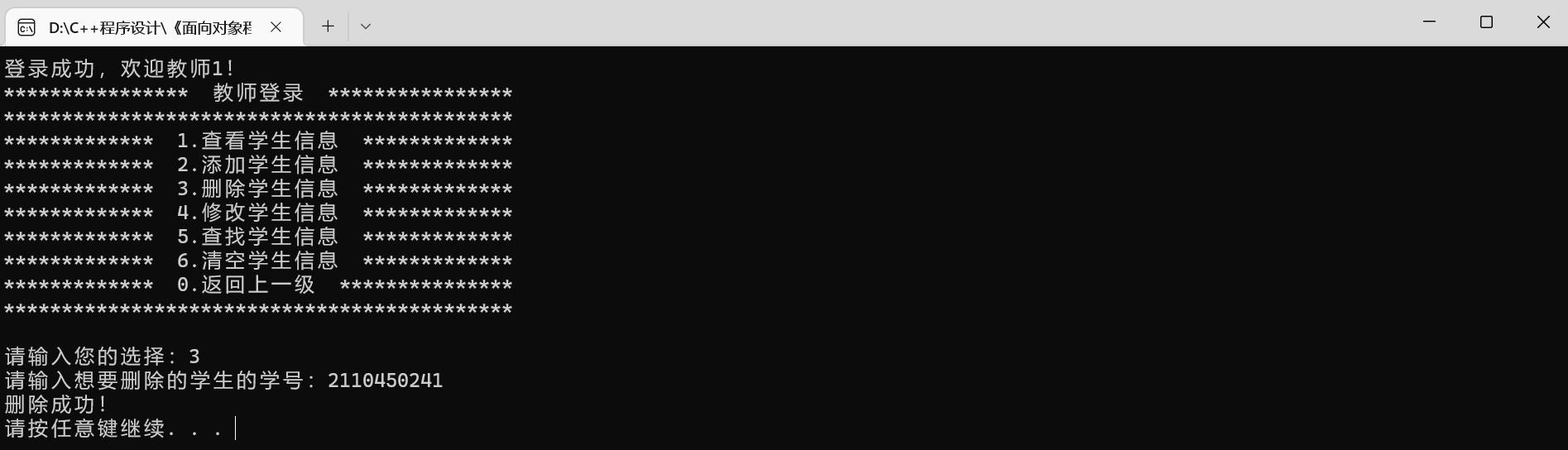


图 4.4-14 教师删除学生信息模块

用户输入“4”，系统进入修改学生信息模块，用户可根据提示修改学生信息，如图4.4-15所示。修改学生信息成功后，系统将返回教师登录模块。

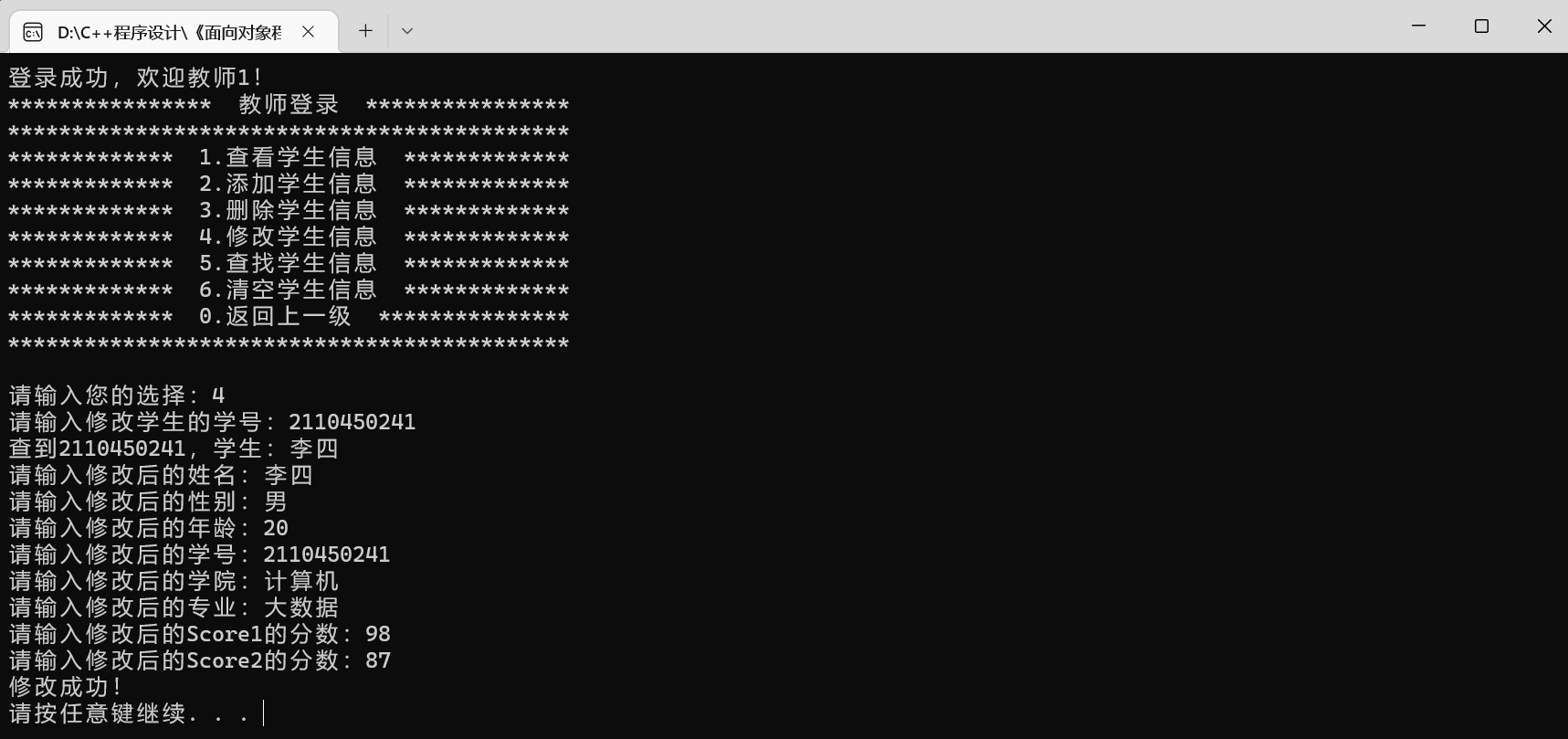


图 4.4-15 教师修改学生信息模块

用户输入“5”，系统进入查找学生信息模块，用户可根据提示查找学生信息，如图4.4-16所示。查找学生信息成功后，系统将返回教师登录模块。

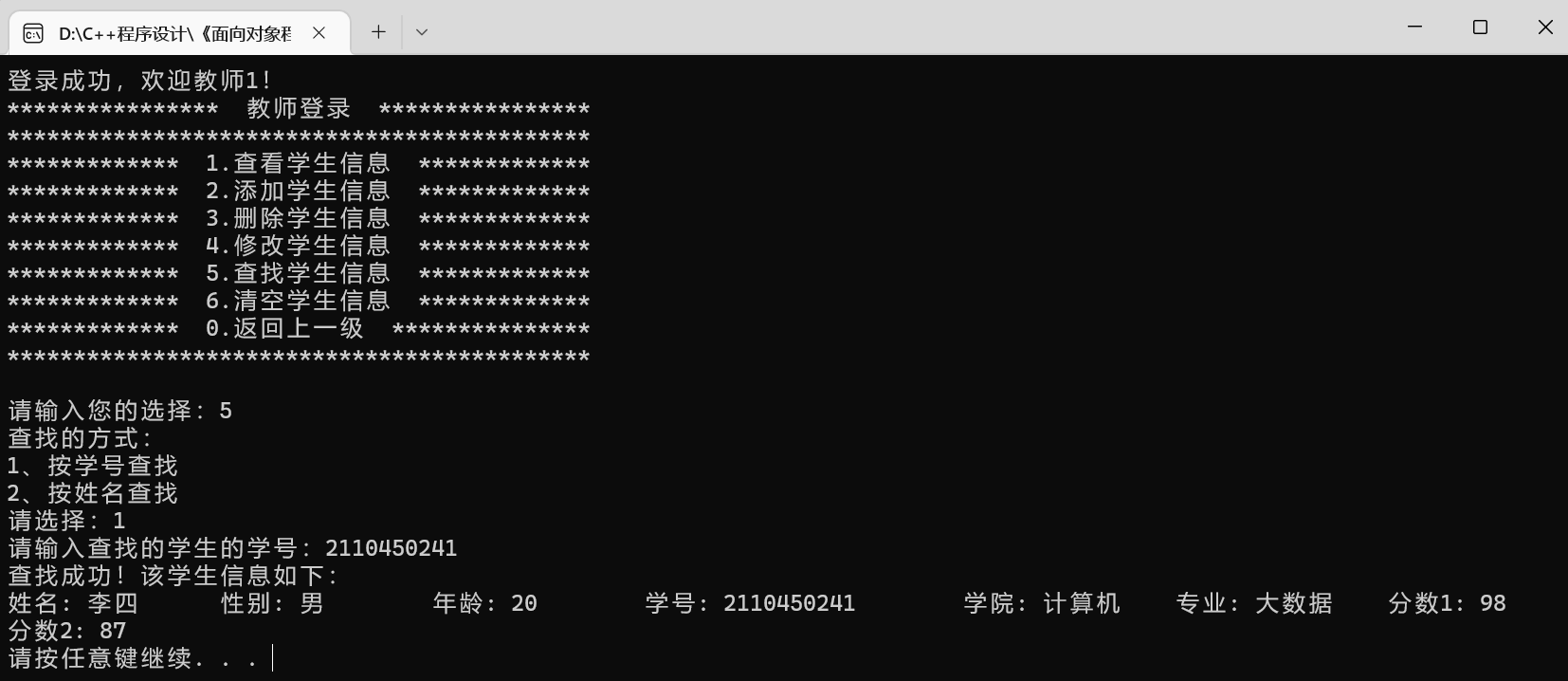


图 4.4-16 教师查找学生信息模块

用户输入“6”，系统进入清空学生信息模块，用户可根据提示清空学生信息，如图4.4-17所示。清空学生信息成功后，系统将返回教师登录模块。

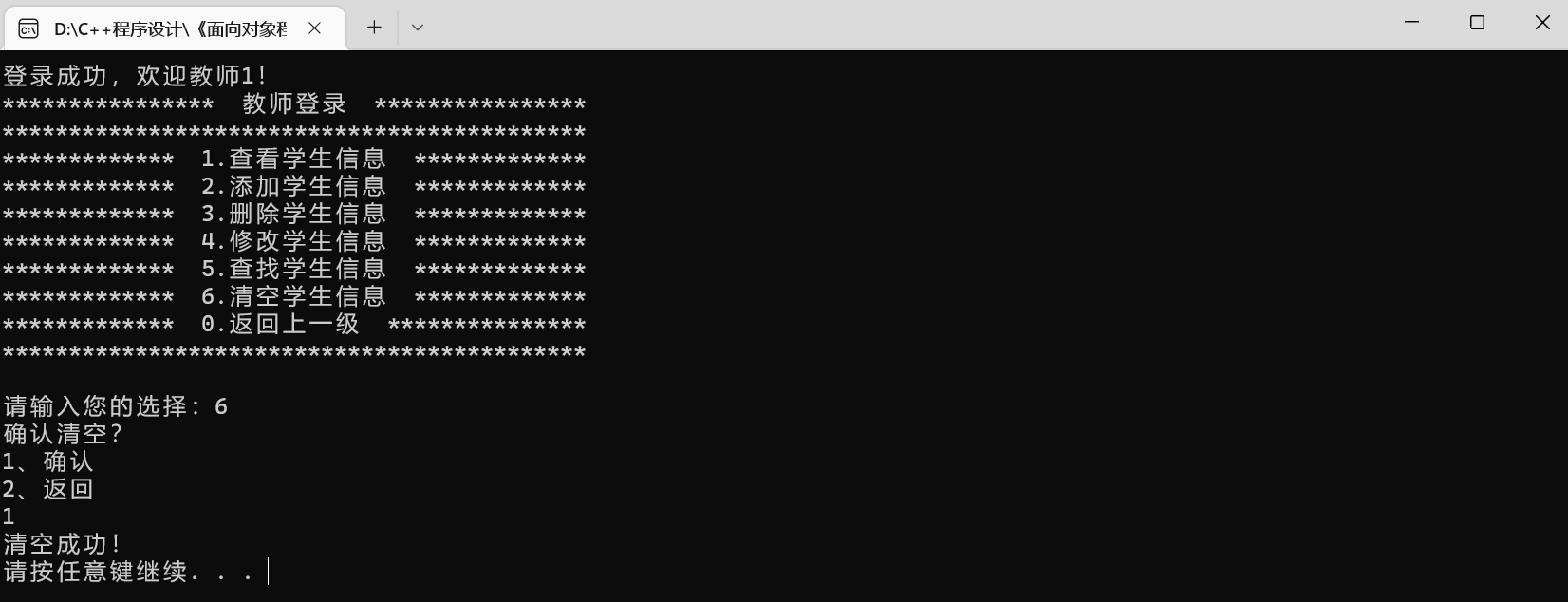


图 4.4-17 教师清空学生信息模块

用户输入“0”，系统将返回主界面。

### 管理员登录

用户在主界面输入“3”，进入管理员登录界面，如图4.4-18所示。系统提示输入操作密码，若密码正确，系统显示二级管理员登录界面，等待用户继续输入。如图4.4-19所示。若密码输入错误时，系统将提示错误，如图4.4-20所示，并回到主界面。

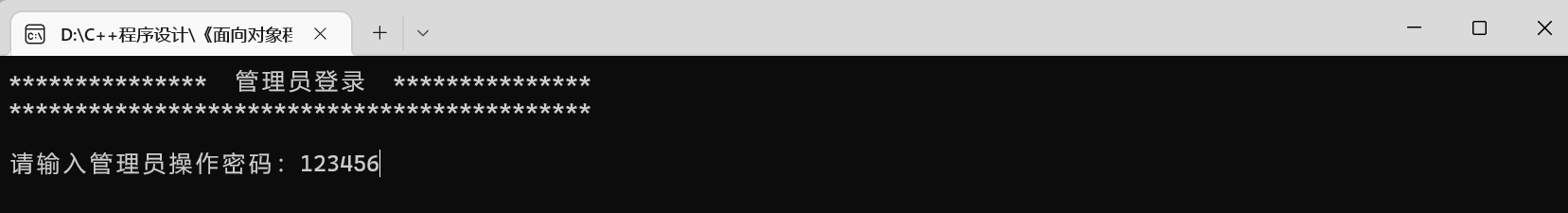


图 4.4-18 输入管理员操作密码

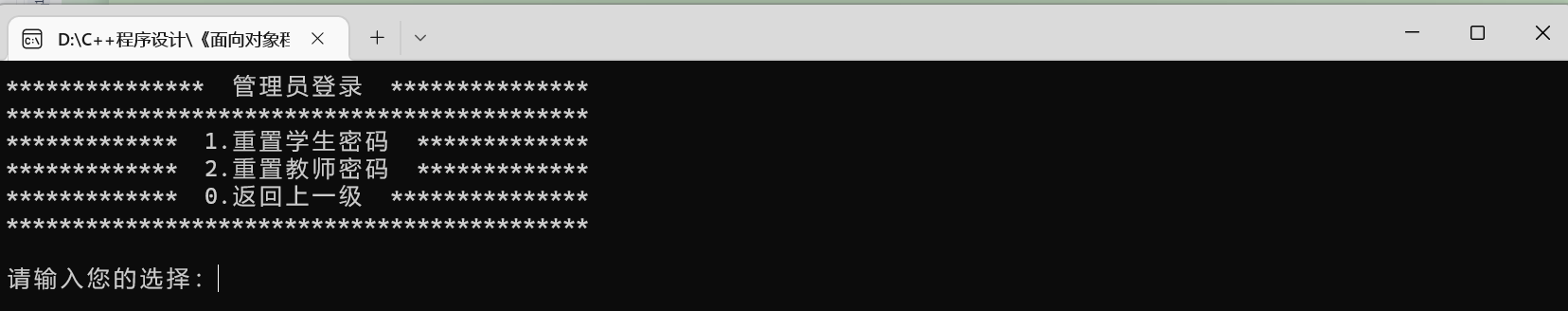


图 4.4-19 管理员登录界面

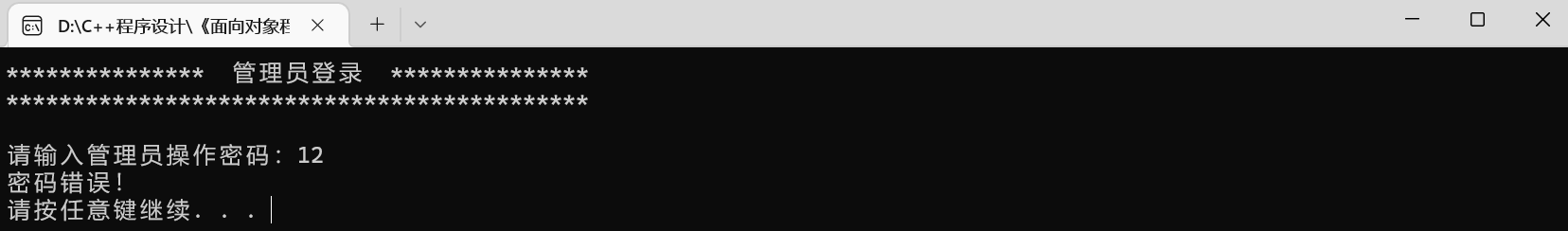


图 4.4-20 管理员操作密码输入错误

用户输入“1”，系统进入重置学生密码模块，用户可根据提示重置学生密码，如图4.4-21所示。重置学生密码成功后，系统将返回管理员登录模块。

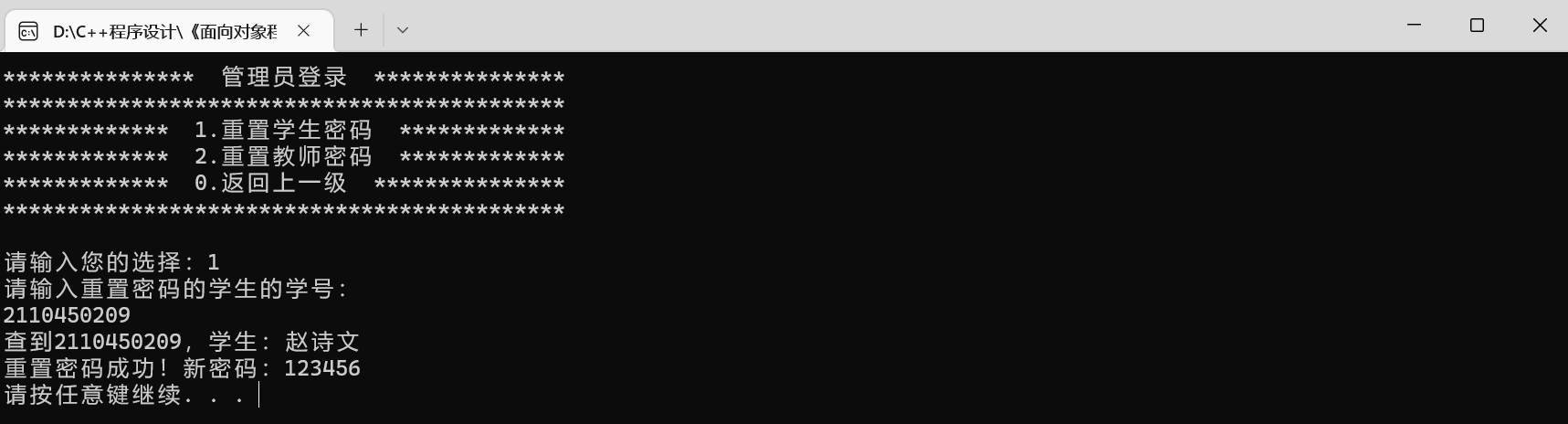


图 4.4-21 管理员重置学生密码模块

用户输入“2”，系统进入重置教师密码模块，用户可根据提示重置教师密码，如图4.4-22所示。重置教师密码成功后，系统将返回管理员登录模块。



图 4.4-22 管理员重置教师密码模块

用户输入“0”，系统将返回主界面。

### 系统退出

在主界面输入“0”，系统将退出，如图4.4-23所示。

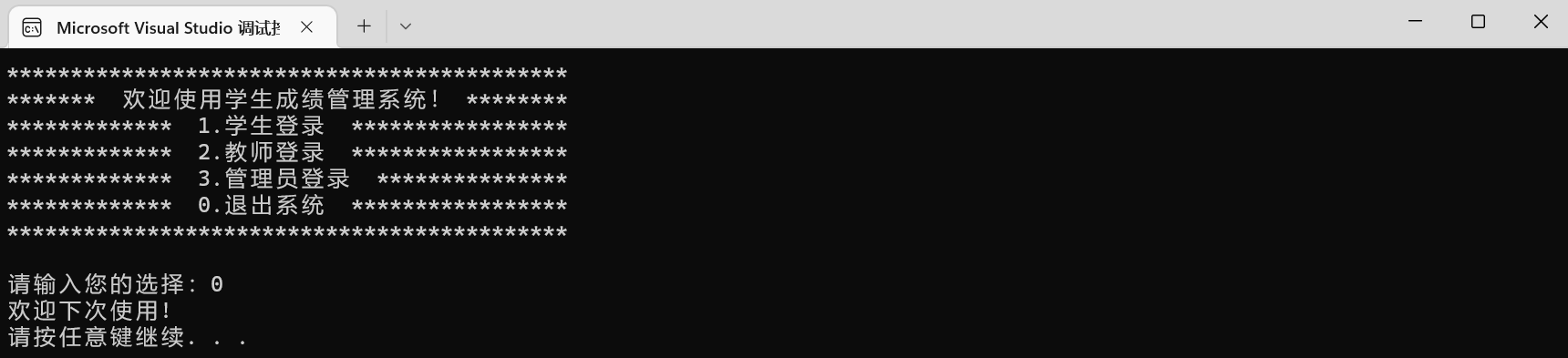


图 4.4-23 退出系统

# 遗留问题

对于学生成绩信息的管理，做到了使用动态数组存储数据信息，但对于教师和管理员信息的管理，则仅是使用默认创建方式，没用为他们提供可操作的内存管理他们各自的信息，也没有提供接口共用户创建或修改信息。

该系统在管理密码方面，没有使用读写文件存储密码的方法，导致了修改密码仅能在系统运行过程中有效，系统退出后，仍是原始密码。

# 设计体会

本次课程设计，使我对C++这门语言有了更深入的理解，更感叹于它的强大。语言学习是一门实践性较强的课程，为了学好这门课程，必须在掌握理论知识的同时，加强动手操作的能力。

在此次课设过程中，我意识到了先设计好再执行的重要性。起初的系统开发，是在我简单设计过后开始的，但由于简单设计的不严谨和设计错误等问题，导致过程中大量时间是在找漏洞，反复推翻之前的设计。其实像这样的问题如果在设计之初详细规划好了，是完全可以避免的。

通过两周课程设计，我认识到了自己的不足，也接触到了之前没有接触过的函数和语法。在对用户使用账号和密码登录、动态数组使用等我都遇到了不小的困难，甚至有过放弃的念头。但在自己的深入思考和查资料下，最终将问题解决。

相较于上一次课设，我小的语法问题少了，但大的方面比如动态数组使用，基本功还是不扎实，在程序设计上也不是很合理。我意识到自己现在的水平有限，以后更需要加大努力，多学习一些新知识，加强上机实验。