

**NANCHANG UNIVERSITY**

**C++语言程序设计**



**题 目:** 学生成绩管理系统设计

**学 院：** 信息工程学院

**专业班级：** 23级新一代电子信息

**姓 名：** 赖宝林

**学 号：** 416100230006

**任课教师：** 王昊

**完成时间：** 2024.1.12

目录

[摘要 2](#_Toc163850138)

[一、开发环境 3](#_Toc163850139)

[二、项目需求分析 3](#_Toc163850140)

[2.1项目背景及简介 3](#_Toc163850141)

[2.2功能需求 4](#_Toc163850142)

[三、系统各部分模块设计与仿真结果 5](#_Toc163850143)

[3.1成绩显示 5](#_Toc163850144)

[3.2成绩查询 7](#_Toc163850145)

[3.3成绩录入 8](#_Toc163850146)

[3.4成绩修改 10](#_Toc163850147)

[3.5成绩删除 12](#_Toc163850148)

[3.6 统计成绩 13](#_Toc163850149)

[五、结论与心得 17](#_Toc163850150)

[六、附完整代码 18](#_Toc163850151)

# 摘要

C++是一种面向对象的编程语言，它的设计目标是提供高效、可移植、面向对象的编程环境。C++继承了C语言的许多特性，同时添加了许多新的功能，如类和对象、继承和多态、模板和泛型编程等，使得它更适合大型项目的开发和管理。学习一门语言最有效的方法就是设计实现一个完整的功能模型。而学生管理系统设计就是一个实用且难度适中的编程任务。

本论文设计并实现了一个成绩管理系统，旨在帮助学校、教育机构或个人方便地管理学生的成绩信息。该系统基于C++编程语言开发，采用面向对象的设计方法，包括类和对象、继承和多态等特性，同时利用C++标准模板库（STL）提供的数据结构和算法，实现了高效、可靠的成绩管理功能。系统的主要功能包括学生信息的录入、成绩的查询和统计、成绩单的生成和输出等。通过简单直观的用户界面，用户可以方便地添加、修改和删除学生信息，查询特定学生或特定科目的成绩，生成学生的成绩单并将其输出到文件或打印。

在系统设计和实现过程中，充分考虑了系统的可扩展性、易用性和稳定性。通过良好的模块化设计和合理的数据结构选择，使得系统具有良好的性能和可维护性，可以适应不同规模和需求的成绩管理任务。

本论文还对系统的设计思路、关键功能模块和实现细节进行了详细介绍和分析，并通过实际案例和测试验证了系统的有效性和实用性。最后，对系统存在的不足之处进行了讨论，并提出了改进和扩展的建议，以进一步完善和优化成绩管理系统的功能和性能。

通过本论文的研究和实践，对于成绩管理系统的设计和开发有了更深入的理解和掌握，为学校和教育机构提供了一个可靠、高效的成绩管理工具，具有一定的实际应用和推广价值。

# 一、开发环境

Microsoft Visual studio 2022

# 二、项目需求分析

### 2.1项目背景及简介

该设计要求学生以学生管理业务为背景,对“学生信息管理系统”软件进行分析和设计。

通过该题目的分析和设计，使学生初步得到软件工程的训练，全面培养软件开发过程中的分析、设计、编码、测试及文档规范书写的能力，得到软件工程的综合训练，提高解决实际问题的能力。

### 2.2功能需求

学生信息管理系统的总体目标是:在计算机网络，数据库和C语言的开发平台上，利用现有的软件，配置一定的硬件，分析和设计一个具有开放体系结构的、易扩充的、易维护的、具有良好人机交互界面的学生信息管理系统，为管理学生信息的决策者和管理者提供充足的信息、快捷的查询和有效的管理方式,减少不必要的损失和浪费,提高学生信息管理的效率

根据可行性研究的结果和系统的要求，分析现有情况及问题，采用C/S或B/S结构，将学生信息管理系统划分为:学生基本资料管理、成绩管理、统计查询管理等。其中具体要求如下:

1.学生成绩管理系统中不同使用单位（用户）的学生人数事先无法确定。

2.该学生成绩管理系统要求有学生成绩录入，查询，修改，删除，统计，保存文件。

3.系统使用文字菜单，用户通过选择菜单项的编号，实现系统对子模块的调用。也可用windows界面下的菜单栏、弹出式菜单、下拉菜单。

（1）查询

根据提供的cj.txt文件中的数据，通过姓名或学号可以查询各课程成绩。

（2）成绩录入

如果cj.txt文件中，学生信息不存在，请添加学生的信息及各课程成绩。

如果学生成绩存在空缺，请修改或添加该课程的成绩到cj.txt文件中。

1）单个学生成绩录入

2）通过读取tjxx.txt文件批量录入学生成绩

（3）数据修改

通过学号或姓名修改cj.txt文件中的任何数据，要考虑姓名可能存在重名的情况，修改学号，姓名，各课程成绩数据中的一项或多项，比如要修改同学的多门课程成绩。

1）单个学生成绩修改

2）通过读取xgxx.txt文件批量修改学生成绩

（4）数据删除

通过学号或姓名删除cj.txt文件中的任何数据，要考虑姓名可能存在重名的情况，删除学号，姓名，各课程成绩数据中的一项或多项，比如要删除同学的所有成绩，或删除该同学的所有数据信息。

1）单个学生成绩删除

2）通过读取scxx.txt文件批量删除学生成绩

（5）统计成绩

1）生成一个xxcj.txt文件，里面统计包括姓名，学号，总成绩（所有课程成绩之和），平均成绩信息。

2）生成以课程名拼音命名的txt文件，里面统计该课程不及格同学的学号，姓名，课程名，成绩的信息。b.

3）生成一个log.txt文件，里面记录添加，修改过的学生数据的原始和修改记录。

# 三、系统各部分模块设计与仿真结果

### 3.1成绩显示

通过制表符或者连续数量的空格从txt文件中读取出数据存入结构体，成绩缺失的元素用-1表示。

函数模块：  
void initializeStudentsFromFile(string filename) {

std::ifstream infile(filename);

if (!infile) {

std::cerr << "Error: Unable to open file!" << std::endl;

return;

}

std::string line;

std::getline(infile, line); // 跳过表头

while (std::getline(infile, line)) {

int score;

std::regex pattern("\\t\\t"); // 匹配一个或多个连续的空格&&制表符

std::string result = std::regex\_replace(line, pattern, " -1 ");

std::regex pattern1("\\s"); // 匹配一个或多个连续的空格&&制表符

std::istringstream ss(result);

Student student;

//// 假设学号和姓名的宽度分别为10和20

ss >> student.id >> student.name;

if (student.id != "-1")

{

//假设每个成绩的宽度为5

for (int i = 0; i < 4; ++i) {

//cout << line << endl;

ss >> std::setw(5);

if (ss >> score)

{

student.scores.push\_back(score);

}

else

{

student.scores.push\_back(-1); // 假设-1代表空白成绩

ss.clear(); // 清除错误状态

}

}

students.push\_back(student);

}

}

infile.close();

}

##### 仿真结果：

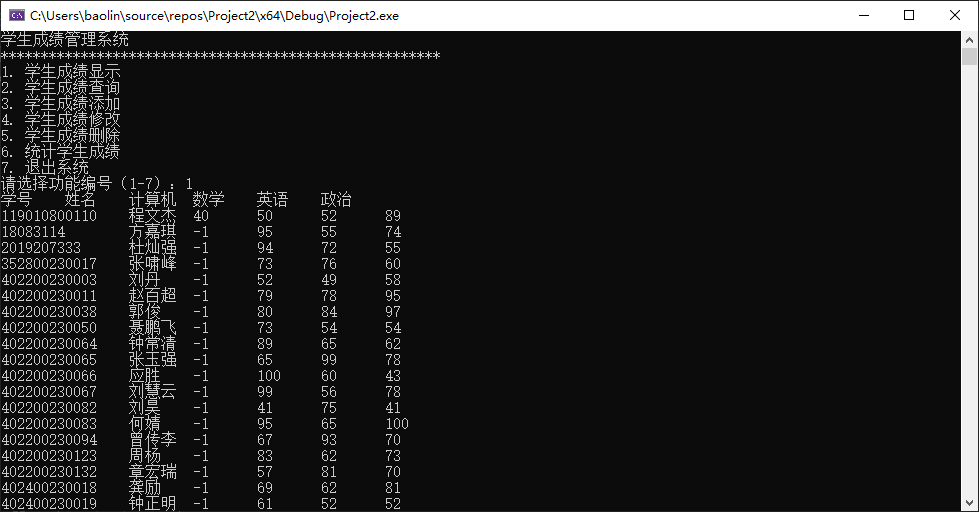


图3.1 成绩显示打印效果（命令行）

### 3.2成绩查询

##### 函数模块：

// 要求一：通过姓名或者学号查询students数组里的课程成绩信息

void queryScoreByNameOrId() {

string input;

cout << "请输入姓名或学号进行查询：";

cin >> input;

bool found = false;

for (const auto& student : students)

{

if (student.name == input || student.id == input) {

cout << "姓名：" << student.name << "\t学号：" << student.id << endl;

cout << "各科成绩：";

for (const auto& score : student.scores) {

cout << score << " ";

}

cout << endl;

found = true;

break;

}

}

if (!found) {

cout << "未找到匹配的学生信息！" << endl;

}

}

##### 仿真结果：

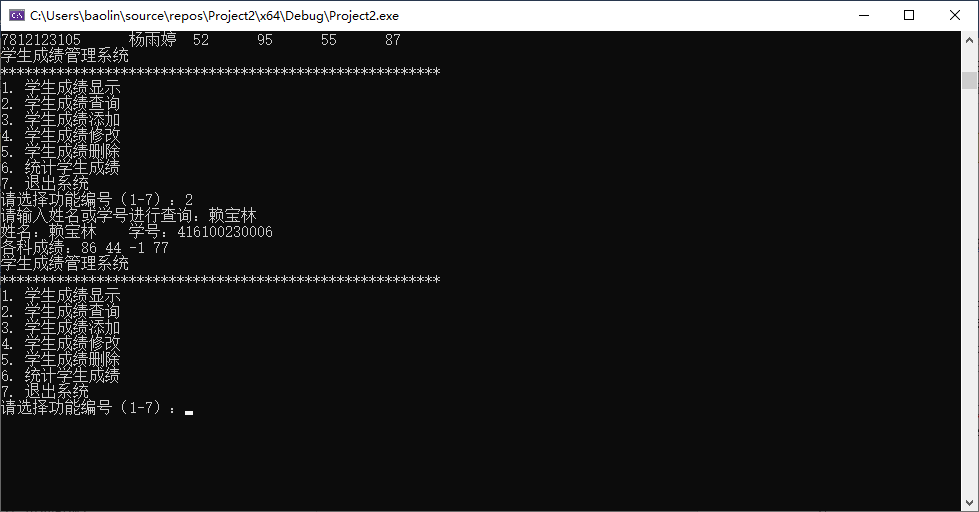


图3.2 成绩查询效果（命令行）

### 3.3成绩录入

##### 函数模块：

//要求2:成绩录入，分为单条录入和通过文件批量导入，若学生信息不在students数组,则新增，若存在则修改

void addScore() {

bool exitLoop = false;

int choice;

do {

cout << "学生成绩管理系统" << endl;

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

cout << "1. 单条信息录入" << endl;

cout << "2. 按文件批量导入" << endl;

cout << "3. 返回上一级目录" << endl;

cout << "请选择功能编号：";

cin >> choice;

switch (choice) {

case 1: {

Student student;

cout << "请输入学生姓名：";

cin >> student.name;

cout << "请输入学生学号：";

cin >> student.id;

cout << "请输入各科成绩（用空格分隔）：";

for (int i = 0; i < 3; ++i) {

int score;

cin >> score;

student.scores.push\_back(score);

}

bool found = false;

for (auto& s : students) {

if (s.name == student.name && s.id == student.id) {

s = student; // 更新学生成绩信息

found = true;

break;

}

}

if (!found) {

students.push\_back(student); // 新增学生成绩信息

}

cout << "成绩录入成功！" << endl;

break;

}

case 2: {

initializeStudentsFromFile("cj.txt");

initializeStudentsFromFile("tjxx.txt");

cout << "批量导入成功！" << endl;

break;

}

case 3:

cout << "返回上一级目录" << endl;

exitLoop = true; // 设置退出循环的条件

break;

default:

cout << "无效的选择，请重新输入！" << endl;

break;

}

} while (!exitLoop);

}

##### 仿真结果：

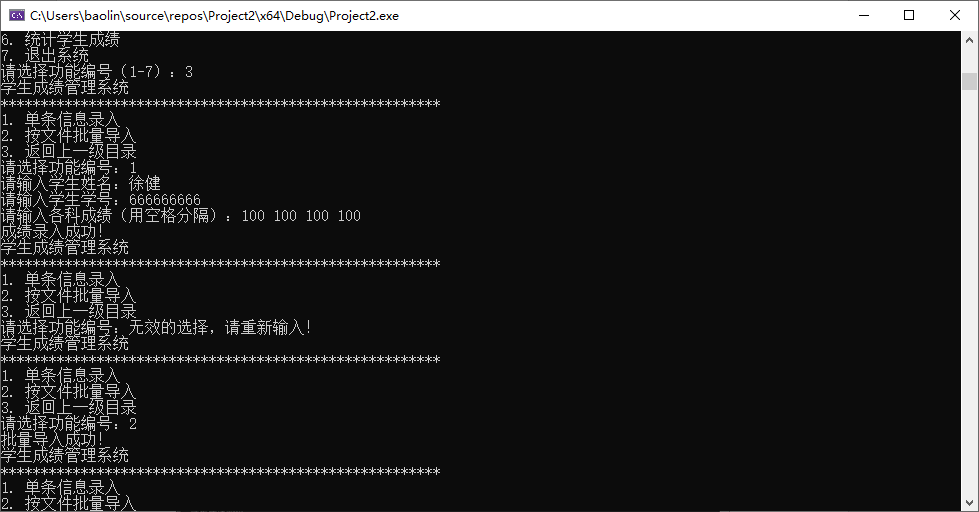


图3.3 成绩录入效果（命令行）

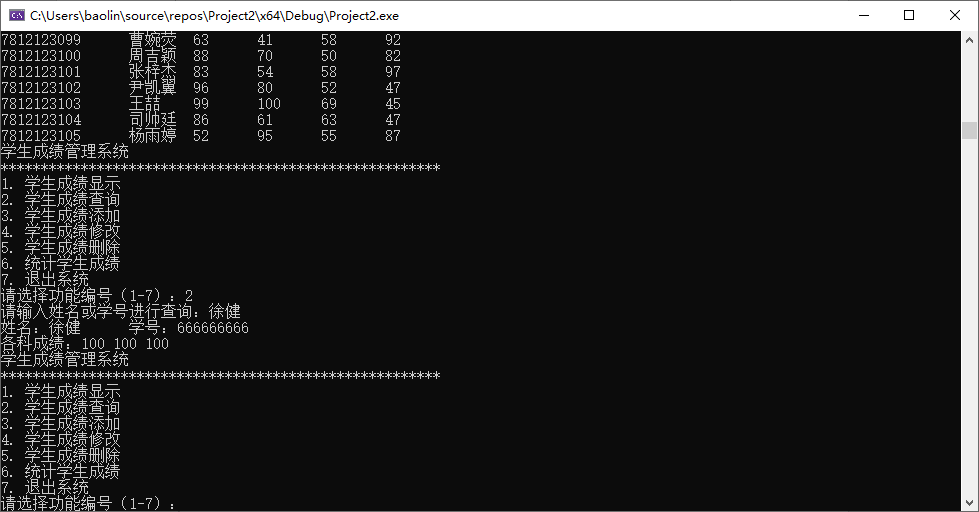


图3.4 成绩录入效果（命令行）

### 3.4成绩修改

##### 函数模块：

//要求4:学生成绩修改，此功能有子目录，包含单条数据修改和读取txt文件批量修改两种形式，单条数据修改：通过输入学号或者

//Id找出students数组中的元素，然后提示输入正确批量修改：

void modifyScore() {

string input;

cout << "请输入要修改的学生姓名或学号：";

cin >> input;

bool found = false;

for (auto& student : students) {

if (student.name == input || student.id == input) {

cout << "找到学生：" << student.name << " 学号：" << student.id << endl;

cout << "请输入要修改的信息：" << endl;

cout << "1. 姓名" << endl;

cout << "2. 学号" << endl;

cout << "3. 各科成绩" << endl;

int choice;

cin >> choice;

if (choice == 1) {

cout << "请输入新的姓名：";

cin >> student.name;

}

else if (choice == 2) {

cout << "请输入新的学号：";

cin >> student.id;

}

else if (choice == 3) {

cout << "请输入新的各科成绩（用空格分隔）：";

for (int i = 0; i < 4; ++i) {

int score;

cin >> score;

student.scores[i] = score;

}

}

else {

cout << "无效的选项！" << endl;

}

cout << "学生信息修改成功！" << endl;

found = true;

break;

}

}

if (!found) {

cout << "未找到匹配的学生信息！" << endl;

}

}

##### 仿真结果：



图3.5 成绩修改效果（命令行）

### 3.5成绩删除

##### 函数模块：

//数据删除

void deleteStudent() {

string input;

cout << "请输入要删除的学生姓名或学号：";

cin >> input;

auto it = students.begin();

bool found = false;

while (it != students.end()) {

if ((\*it).name == input || (\*it).id == input) {

it = students.erase(it); // 删除匹配的学生信息

cout << "学生信息删除成功！" << endl;

found = true;

}

else {

++it;

}

}

if (found == false) {

cout << "查无此人！" << endl;

}

}

##### 仿真结果：

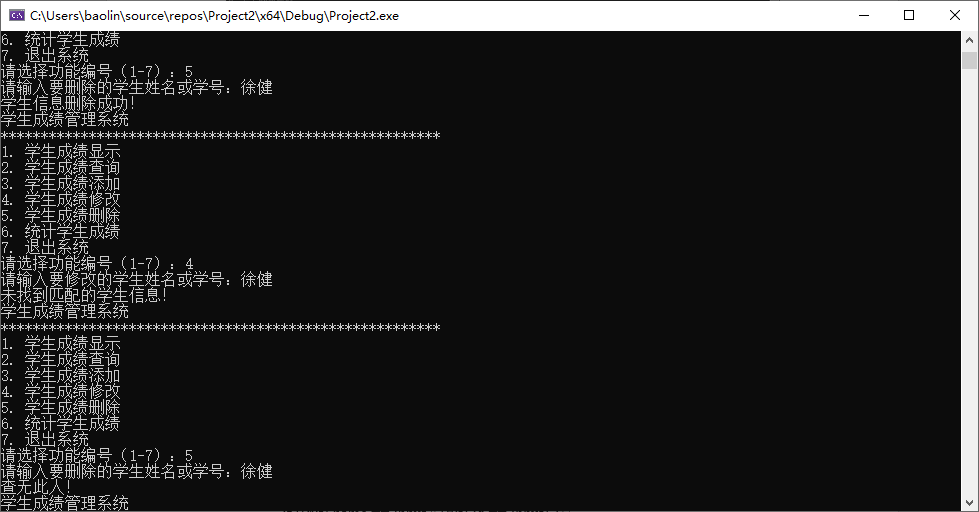


图3.6 成绩删除效果（命令行）

### 3.6 统计成绩

##### 函数模块：

//统计成绩

void generateTotalAndAverageScoreFile()

{

ofstream outfile("xxcj.txt");

if (!outfile) {

cerr << "Error: Unable to create file!" << endl;

return;

}

for (const auto& student : students) {

int totalScore = 0;

for (int score : student.scores) {

if (score != -1){

totalScore += score;

}

}

double averageScore = static\_cast<double>(totalScore) / student.scores.size();

outfile << "姓名：" << student.name << "\t学号：" << student.id << "\t总成绩：" << totalScore << "\t平均成绩：" << averageScore << endl;

}

outfile.close();

cout << "xxcj.txt 文件生成成功！" << endl;

}

void generateCourseScoreFile(const string& courseName) {

ofstream outfile(courseName + "未及格名单.txt");

if (!outfile) {

cerr << "Error: Unable to create file!" << endl;

return;

}

for (const auto& student : students) {

for (int i = 0; i < student.scores.size(); ++i) {

if (student.scores[i] < 60) {

outfile << "姓名：" << student.name << "\t学号：" << student.id << "\t课程名：" << courseName << "\t成绩：" << student.scores[i] << endl;

break; // 只需要记录不及格的一门课程

}

}

}

outfile.close();

cout << courseName << ".txt 文件生成成功！" << endl;

}

void generateLog() {

ofstream outfile("log.txt", ios::app);

if (!outfile) {

cerr << "Error: Unable to create file!" << endl;

return;

}

for (const auto& student : students) {

outfile << "姓名：" << student.name << "\t学号：" << student.id << "\t成绩：";

for (int score : student.scores) {

outfile << score << " ";

}

outfile << endl;

}

outfile.close();

cout << "log.txt 文件生成成功！" << endl;

}

##### 仿真结果：

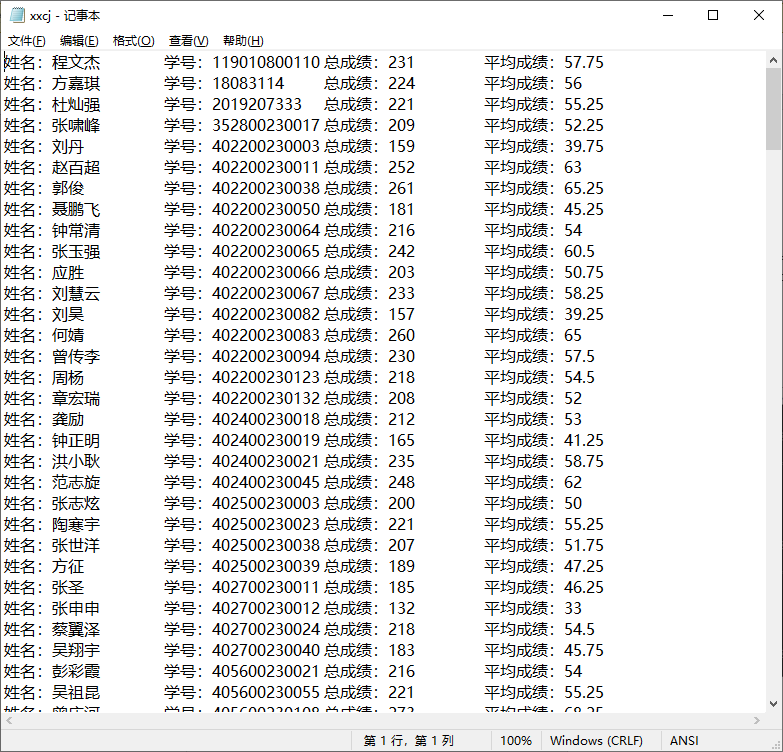


图3.7 平均分统计（文档）

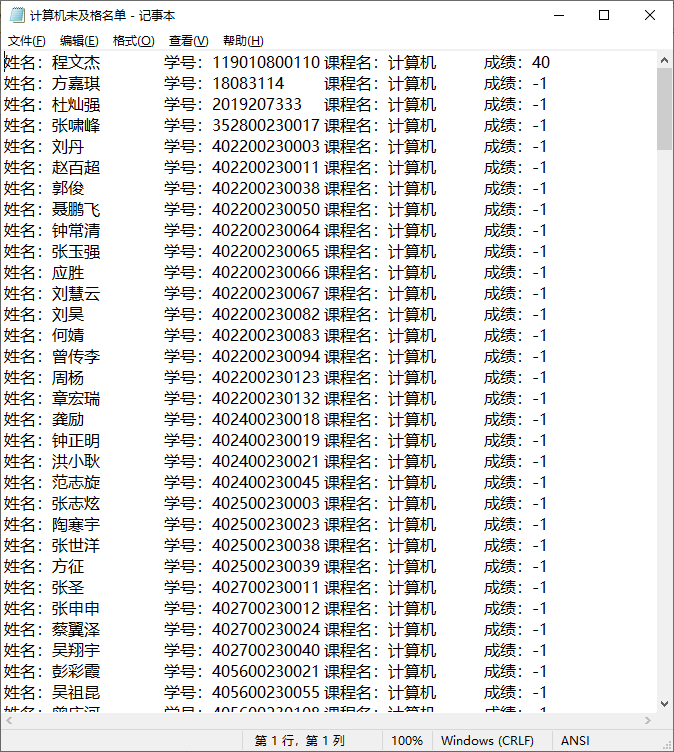


图3.8 不及格统计（文档）

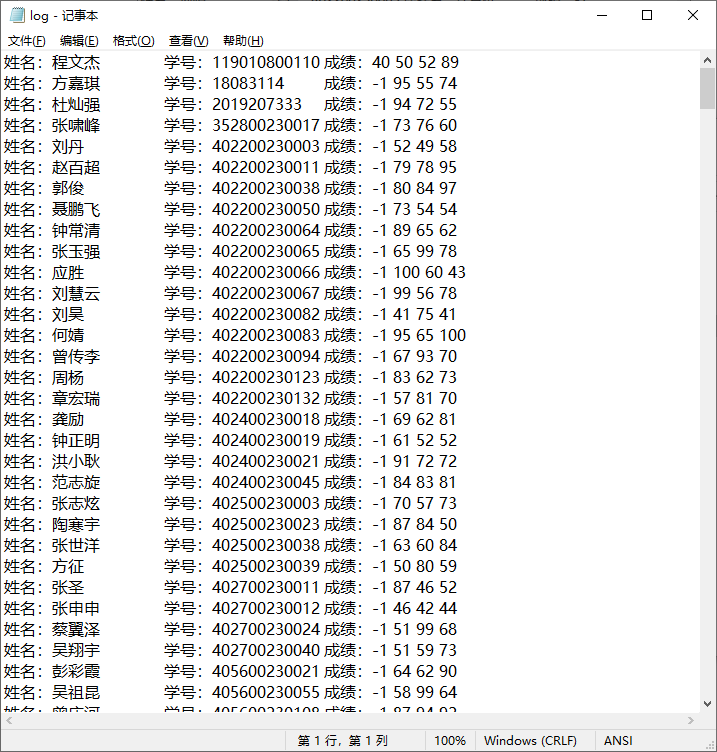


图3.9 操作记录（文档）

# 四、结论与心得

通过此次实验，我的C++语言编程的能力有了提高，特别是在数组、函数和结构体方面有了很大提高，另外还学习了课本没有讲到的知识，学会了使用模块化编程的简单方法，另外在对话框的编程要求美观、方面也有了体会。

数组在程序中有着很重要的作用，对数据的贮存、输入、读取起着很重要的作用，本程序几乎所有数据都以数组进行输入、贮存、读取方便了对数据的处理和输入，数组的使用在本程序中有很大的好处。

C++作为一种面向对象的编程语言，具有丰富的功能和强大的扩展性，能够满足成绩管理系统的需求。通过面向对象的设计方法，将系统划分为不同的模块和类，提高了代码的复用性和可维护性。

其次，借助C++标准模板库（STL）提供的数据结构和算法，实现了高效的成绩管理功能。例如，利用向量容器存储学生对象，通过迭代器进行遍历和操作，大大简化了代码的编写和管理。

在系统开发过程中，遇到了一些挑战和困难，如数据存储和管理的设计、用户界面的友好性等。通过不断的学习和调试，逐步解决了这些问题，并对C++语言和面向对象编程有了更深入的理解和掌握。

总的来说，通过本论文的研究和实践，我对C++编程语言和面向对象的设计思想有了更深入的认识和理解，同时也掌握了开发成绩管理系统的基本方法和技巧。在今后的学习和工作中，我将继续努力提升自己的编程能力，不断完善和优化成绩管理系统，为教育工作和学生管理提供更好的服务。

函数是实现模块化编程的基础，它使程序变得方面易懂，对修改起了很大作用，也降低了编程的难度，一个函数就是一个功能，它使各个功能之间有了一定的独立性，一个函数的错误，不会导致整个程序的失败(主函数除外)，对函数的良好应用时在本程序中又以深刻的体会。也有了很大提高。

结构体的使用，为本程序提供了很大方便，也提高了本程序的可读性，结构数组方便了对数据的管理。

文件的读取与存贮使数据可以保存下来，这也是在学校的学习过程中没有掌握的内容，在开始时遇到了不小的困难。

另外、感觉到在学校编的程序时那样的简短、自己需要在编程方面进一步提高，也提高了自己对编程的兴趣，使自己对编程有了更深的理解，体会到编程的乐趣。

# 五、附完整代码

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

#include <algorithm>

#include<sstream>

#include <iomanip>

#include <regex>

using namespace std;

struct Student {

string name;

string id;

vector<int> scores;

};

std::vector<Student> students;

// 初始化students变量，从cj.txt文件读取数据

void initializeStudentsFromFile(string filename) {

std::ifstream infile(filename);

if (!infile) {

std::cerr << "Error: Unable to open file!" << std::endl;

return;

}

std::string line;

std::getline(infile, line); // 跳过表头

while (std::getline(infile, line)) {

int score;

std::regex pattern("\\t\\t"); // 匹配一个或多个连续的空格&&制表符

std::string result = std::regex\_replace(line, pattern, " -1 ");

std::regex pattern1("\\s"); // 匹配一个或多个连续的空格&&制表符

std::istringstream ss(result);

Student student;

//// 假设学号和姓名的宽度分别为10和20

ss >> student.id >> student.name;

if (student.id != "-1")

{

//假设每个成绩的宽度为5

for (int i = 0; i < 4; ++i) {

//cout << line << endl;

ss >> std::setw(5);

if (ss >> score)

{

student.scores.push\_back(score);

}

else

{

student.scores.push\_back(-1); // 假设-1代表空白成绩

ss.clear(); // 清除错误状态

}

}

students.push\_back(student);

}

}

infile.close();

}

//初始界面

void displayMenu() {

cout << "学生成绩管理系统" << endl;

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

cout << "1. 学生成绩显示" << endl;

cout << "2. 学生成绩查询" << endl;

cout << "3. 学生成绩添加" << endl;

cout << "4. 学生成绩修改" << endl;

cout << "5. 学生成绩删除" << endl;

cout << "6. 统计学生成绩" << endl;

cout << "7. 退出系统" << endl;

cout << "请选择功能编号（1-7）：";

}

// 要求一：通过姓名或者学号查询students数组里的课程成绩信息

void queryScoreByNameOrId() {

string input;

cout << "请输入姓名或学号进行查询：";

cin >> input;

bool found = false;

for (const auto& student : students)

{

if (student.name == input || student.id == input) {

cout << "姓名：" << student.name << "\t学号：" << student.id << endl;

cout << "各科成绩：";

for (const auto& score : student.scores) {

cout << score << " ";

}

cout << endl;

found = true;

break;

}

}

if (!found) {

cout << "未找到匹配的学生信息！" << endl;

}

}

//要求2:成绩录入，分为单条录入和通过文件批量导入，若学生信息不在students数组,则新增，若存在则修改

void addScore() {

bool exitLoop = false;

int choice;

do {

cout << "学生成绩管理系统" << endl;

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

cout << "1. 单条信息录入" << endl;

cout << "2. 按文件批量导入" << endl;

cout << "3. 返回上一级目录" << endl;

cout << "请选择功能编号：";

cin >> choice;

switch (choice) {

case 1: {

Student student;

cout << "请输入学生姓名：";

cin >> student.name;

cout << "请输入学生学号：";

cin >> student.id;

cout << "请输入各科成绩（用空格分隔）：";

for (int i = 0; i < 3; ++i) {

int score;

cin >> score;

student.scores.push\_back(score);

}

bool found = false;

for (auto& s : students) {

if (s.name == student.name && s.id == student.id) {

s = student; // 更新学生成绩信息

found = true;

break;

}

}

if (!found) {

students.push\_back(student); // 新增学生成绩信息

}

cout << "成绩录入成功！" << endl;

break;

}

case 2: {

initializeStudentsFromFile("cj.txt");

initializeStudentsFromFile("tjxx.txt");

cout << "批量导入成功！" << endl;

break;

}

case 3:

cout << "返回上一级目录" << endl;

exitLoop = true; // 设置退出循环的条件

break;

default:

cout << "无效的选择，请重新输入！" << endl;

break;

}

} while (!exitLoop);

}

//要求3:选1时以数据表的格式显示出cj.txt文件里的成绩数据

void showGradesData() {

//ifstream infile("cj.txt");

//if (!infile) {

// cerr << "Error: Unable to open file!" << endl;

// return;

//}

//Student student;

//while (infile >> student.id >> student.name) {

// for (int i = 0; i < 3; ++i) {

// int score;

// if (!(infile >> score)) {

// break; // 如果读取成绩失败，则跳出内部循环

// }

// student.scores.push\_back(score);

// }

// students.push\_back(student);

//}

//students.erase(students.end() - 1);

//infile.close();

// 打印表头

cout << "学号\t姓名\t计算机\t数学\t英语\t政治" << endl;

// 打印数据

for (const auto& student : students) {

cout << student.id << "\t" << student.name << "\t";

for (const auto& score : student.scores) {

cout << score << "\t";

}

cout << endl;

}

}

//要求4:学生成绩修改，此功能有子目录，包含单条数据修改和读取txt文件批量修改两种形式，单条数据修改：通过输入学号或者

//Id找出students数组中的元素，然后提示输入正确批量修改：

void modifyScore() {

string input;

cout << "请输入要修改的学生姓名或学号：";

cin >> input;

bool found = false;

for (auto& student : students) {

if (student.name == input || student.id == input) {

cout << "找到学生：" << student.name << " 学号：" << student.id << endl;

cout << "请输入要修改的信息：" << endl;

cout << "1. 姓名" << endl;

cout << "2. 学号" << endl;

cout << "3. 各科成绩" << endl;

int choice;

cin >> choice;

if (choice == 1) {

cout << "请输入新的姓名：";

cin >> student.name;

}

else if (choice == 2) {

cout << "请输入新的学号：";

cin >> student.id;

}

else if (choice == 3) {

cout << "请输入新的各科成绩（用空格分隔）：";

for (int i = 0; i < 4; ++i) {

int score;

cin >> score;

student.scores[i] = score;

}

}

else {

cout << "无效的选项！" << endl;

}

cout << "学生信息修改成功！" << endl;

found = true;

break;

}

}

if (!found) {

cout << "未找到匹配的学生信息！" << endl;

}

}

//数据删除

void deleteStudent() {

string input;

cout << "请输入要删除的学生姓名或学号：";

cin >> input;

auto it = students.begin();

bool found = false;

while (it != students.end()) {

if ((\*it).name == input || (\*it).id == input) {

it = students.erase(it); // 删除匹配的学生信息

cout << "学生信息删除成功！" << endl;

found = true;

}

else {

++it;

}

}

if (found == false) {

cout << "查无此人！" << endl;

}

}

//统计成绩

void generateTotalAndAverageScoreFile()

{

ofstream outfile("xxcj.txt");

if (!outfile) {

cerr << "Error: Unable to create file!" << endl;

return;

}

for (const auto& student : students) {

int totalScore = 0;

for (int score : student.scores) {

if (score != -1){

totalScore += score;

}

}

double averageScore = static\_cast<double>(totalScore) / student.scores.size();

outfile << "姓名：" << student.name << "\t学号：" << student.id << "\t总成绩：" << totalScore << "\t平均成绩：" << averageScore << endl;

}

outfile.close();

cout << "xxcj.txt 文件生成成功！" << endl;

}

void generateCourseScoreFile(const string& courseName) {

ofstream outfile(courseName + ".txt");

if (!outfile) {

cerr << "Error: Unable to create file!" << endl;

return;

}

for (const auto& student : students) {

for (int i = 0; i < student.scores.size(); ++i) {

if (student.scores[i] < 60) {

outfile << "姓名：" << student.name << "\t学号：" << student.id << "\t课程名：" << courseName << "\t成绩：" << student.scores[i] << endl;

break; // 只需要记录不及格的一门课程

}

}

}

outfile.close();

cout << courseName << ".txt 文件生成成功！" << endl;

}

void generateLog() {

ofstream outfile("log.txt", ios::app);

if (!outfile) {

cerr << "Error: Unable to create file!" << endl;

return;

}

for (const auto& student : students) {

outfile << "姓名：" << student.name << "\t学号：" << student.id << "\t成绩：";

for (int score : student.scores) {

outfile << score << " ";

}

outfile << endl;

}

outfile.close();

cout << "log.txt 文件生成成功！" << endl;

}

//将变量写入txt

void saveDataToFile() {

ofstream outfile("cj.txt");

if (!outfile) {

cerr << "Error: Unable to open file!" << endl;

return;

}

outfile << "学号\t姓名\t计算机\t数学\t英语\t政治" << endl;

for (const auto& student : students) {

outfile << student.id << "\t" << student.name << "\t";

for (const auto& score : student.scores) {

outfile << score << "\t";

}

outfile << endl;

}

outfile.close();

}

//主函数

int main() {

initializeStudentsFromFile("cj.txt");

int choice;

do {

displayMenu();

cin >> choice;

switch (choice) {

case 1:

showGradesData();

break;

case 2:

queryScoreByNameOrId();

break;

case 3:

addScore();

break;

case 4:

modifyScore();

break;

case 5:

deleteStudent();

break;

case 6:

cout << "学生成绩管理系统" << endl;

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

cout << "学生成绩管理系统" << endl;

cout << "1. 生成总成绩和平均成绩文件（xxcj.txt）" << endl;

cout << "2. 生成不及格学生文件" << endl;

cout << "3. 生成日志文件（log.txt）" << endl;

int option;

cin >> option;

if (option == 1) {

generateTotalAndAverageScoreFile();

}

else if (option == 2) {

string courseName;

cout << "请输入课程名（拼音）：";

cin >> courseName;

generateCourseScoreFile(courseName);

}

else if (option == 3) {

generateLog();

}

else {

cout << "无效的选项！" << endl;

}

break;

// 其他功能实现略

case 7:

saveDataToFile();

cout << "感谢使用，已退出程序！" << endl;

break;

default:

cout << "无效的选择，请重新输入！" << endl;

break;

}

} while (choice != 0);

return 0;

}