C++课程

课程设计报告

**班级：23Z202**

**学号：20201000128**

**姓名：刘瑾瑾**

**2021.06.22**

1. **课程设计内容和要求**

学生成绩管理是高等学校教务管理的重要组成部分，主要包括学生成绩的录入、删除、查找及修改、统计分析等等。请设计一个系统实现对学生成绩的管理。系统要求实现以下功能：

（一）基本要求：

1. 删除一个学生的信息：可以先查找姓名，再删除。删除前，要求用户确认。
2. 成绩录入：要求从文件中读取。
3. 成绩修改：若输入错误可进行修改，可以先查找姓名，再修改某一门课程的成绩。
4. 增加科目：允许用户为某一学生新增科目名称和成绩。
5. 查找：请实现一下4种查找功能。
6. 根据姓名查找某个学生的所有课程成绩信息。
7. 根据班号查找某个班所有学生的已学课程的成绩信息。
8. 根据课程名查找选修该门课所有学生的成绩信息。
9. 查找所有学生的已选课程的成绩信息。
10. 排序功能：查找后的结果默认按学号排序，对根据班号查找和查找所有学生信息可按已修学分从高到低进行排序，对课程名查找后可选择按照成绩从高到低进行排序。
11. 统计分析：对某个班级学生的单科成绩进行统计，求出平均成绩，显示该门课程成绩不及格的学生，求出该门课程的标准差和合格率；统计每个学生完成的总学分、不及格的课程数和未修学分。
12. 文件操作：可以打开文件，显示班级的所有学生信息；可以将增加或修改后的成绩重新写入文件；可以将排序好的信息写入新的文件。
13. 界面设计：所有操作都要通过交互界面来操作和显示。

（二）较高要求：

查找可以实现模糊查询，即输入名字的一部分，可以列出满足条件的所有记录。再从这个记录中进行二次选择。

1. **需求分析**
2. 问题描述：设计一个学生管理系统，实现对学生成绩的管理，使其可以实现录入成绩信息、删除学生信息、修改学生成绩、增加学生科目和成绩、查找信息、排序、统计信息、实现相应的文件操作等基础功能以及模糊查找的高级功能。
3. 系统环境：Visual Studio 2019
4. 运行要求：对测试数据可以安全高效的实现上述功能。
5. **概要设计**
6. 系统流程设计

2、系统模块设计

**学生成绩管理系统**

**1、录入学生成绩**

**2、删除学生信息**

**3、修改学生成绩**

**4、增加学生科目和成绩**

**5、查找学生信息**

**6、统计学生信息**

**7、模糊查询信息**

**8、将修改后的信息重新写入文件**

**9、安全退出**

**1、按姓名查找信息**

**2、按班号查找信息**

**3、按课程名查找信息**

**4、查找所有学生信息**

**5、退出查找**

**1、对班级的成绩进行统计**

**2、对学生成绩进行统计**

**1、成绩录入模块**

**2、删除学生信息模块**

**3、修改学生成绩模块**

**4、增加学生科目和成绩模块**

**5、默认排序模块**

**6、查找学生信息模块**

**7、统计信息模块**

**8、模糊查询模块**

**9、重新写入文件模块**

**主函数模块**

1. **详细设计**
   1. 设计三个类：课程类(course)、学生类（student）、系统管理类(manage)。
2. 课程类包括学生姓名、学号、课程编号、课程名、课程分数、课程学分。
3. 学生类包括学生姓名、学号、班号、专业、需完成的总学分、已完成学分、未完成学分和成绩不及格科目以及一个course类型的容器（vector<course>m\_cs）。
4. 系统管理类包括一个student类型的容器（vector<student>m\_a）。另外，系统管理类包括实现相关操作的函数：成绩录入函数（input），删除函数（remove）,

修改学生成绩函数（change）,增加学生科目和成绩函数（add）,根据姓名查找

函数（search1），根据班号查找函数（search2）,根据课程名查找函数（search3）,

查找所有信息函数（search4）,统计班级信息函数（statistic1），统计学生信息函数（statistic2）,默认排序函数（rank）,模糊查询函数（fuzzy\_search）,

重新写入文件函数（write\_in）。

（4）以上三个类互为友元，可互相访问，类的成员以及成员函数默认为私有，在一定程度上保障信息的安全性

2、主函数显示功能菜单，提供给使用者选择操作。操作之前，会显示菜单。每次模块调用结束，也会显示菜单，以便下一步操作。定义manage类对象grades,通过该对象调用相关函数。

3、（1）成绩录入模块（input）：

第一步：打开文件student.txt，通过student类型vector录入学生基本信息，并利用迭代器将其输出，在界面上显示。

第二步：打开文件score.txt,利用学生的学号判断课程信息是否属于相应学生，再通过相应学生的course类型vector录入相应成绩信息。

第三步：打开文件module.txt，利用课程类的课程名判断课程信息是否相符，将相应课程的学分录入每个学生的每一门课程。

第四步：为方便后续排序以及统计等处理，在相关信息录入完成后，直接计算已完成的学分，未完成的学分，不及格科目。

（2）删除学生信息模块（remove）：

输入学生名字，再次确认后，通过for循环查找学生，找到后，通过for循环交换该学生和他后面的学生，直至该学生被换到容器最后一个，利用pop\_back()函数将其移除容器m\_a。

（３）修改学生成绩模块(change)：

首先输入需要修改信息的学生的名字，其次输入需要修改的课程，通过for循环在ｍ\_ａ容器中查找学生，找到储存学生课程信息的容器m\_cs,再通过for循环找到需要修改的课程，经过界面提示，输入修改后的成绩即可修改。如果想继续修改，可输入Y／ｙ，不想继续修改，可输入N／ｎ。

（４）增加学生科目和成绩模块（add）：

首先输入需要增加科目的学生名字，通过for循环在m\_a容器中找到该学生信息，定义一个course类型对象a,将输入的课程以及成绩储存到a中。其次，打开module.txt 文件，查找新增科目的学分，获取课程学分。然后，完善a的其他成员，调用push\_back()将其插入容器m\_cs的末尾。最后，在已有学生信息的基础上，再次计算该学生相应已完成学分，未完成学分，不及格科目。通过两次do while循环实现对学生科目和成绩的持续增加，不想继续增加可退出。

（5）默认排序模块（rank）：

利用冒泡排序法对学生信息按照学号从小到大进行默认排序。

（6）查找学生信息模块（search）：

在进行查询之前，调用默认排序函数，将所有学生信息按照已完成学分从高到低排序。

根据姓名查找函数模块（search1）：

输入需要查找的学生姓名，通过for循环查找，找到后,将其课程成绩信息输出，在界面显示。

根据班号查找函数（search2）：

首先，定义student类型容器vi,通过for循环根据班号进行查找后，将符合条件的student元素m\_a[i]放入vi中。其次，将vi中元素按照完成的学分通过冒泡法按照从高到低的顺序进行排序。最后，打开文件student2.txt,将排序好的信息在界面进行显示，并通过ios::app输入文件student2.txt。

根据课程名查找函数（search3）：

首先，定义course类型容器a，通过for循环根据课程名查找后，将符合条件的course元素放入a中。然后，将a中元素按照课程成绩通过冒泡法按照从高到低的顺序进行排序。最后，打开文件student2.txt,将排序好的信息在界面进行显示，并通过ios::app输入文件student2.txt。

查找所有信息函数（search4）：

将容器中所有student元素通过冒泡排序法按照已完成学分从高到低进行排序。然后，打开文件student2.txt,将排序好的信息在界面进行显示，并通过ios::app输入文件student2.txt。

(7)统计信息模块（statistic）：

统计班级信息函数（statistic1）：

第一步：定义average（平均数）、dev（标准差）、pass（合格率）等对象，定义string类型的容器a。

第二步：打开文件module.txt，将其中的各个课程放入容器a中。

第三步：输入需要查找的班级，将符合的学生信息筛选出来，利用容器a，对该班级所有科目进行平均数、标准差、合格率和未通过学生名单的统计，并输出显示。同时，打开student2.txt文件，将统计后的信息通过ios::app写入。

统计学生信息函数（statistic2）：

调用默认排序函数，进行排序后在界面显示。同时，打开文件student2.txt，将统计好的信息通过ios::app写入文件。

（8）模糊查询模块（fuzzy\_search）：

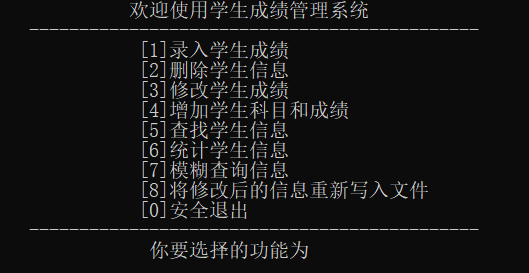
定义student类型容器vi，定义string类型对象c，用来获取输入的学生姓名的一部分。第一次选择：通过m\_a[i].m\_name.fine(c)!=string::npos判断，筛选出合适的元素，将其显示在界面上，并将其放入容器vi中。

第二次选择：通过vi[i].m\_name.find(c)!=string::npos判断，筛选出合适的元素，将其显示在界面上。

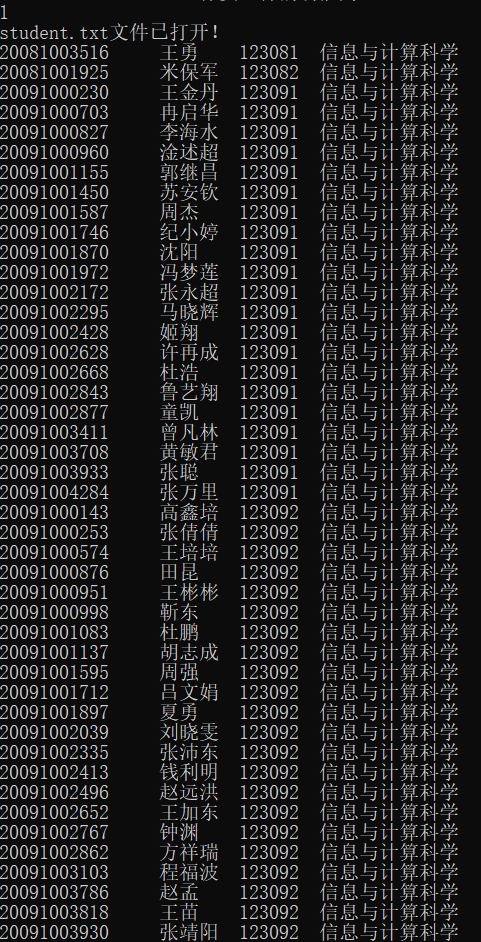
（9）重新写入文件函数模块（write\_in）：

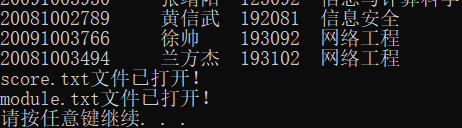
打开score.txt 文件将经过操作的数据（学生成绩）通过ios::out写入文件，覆盖原有数据。

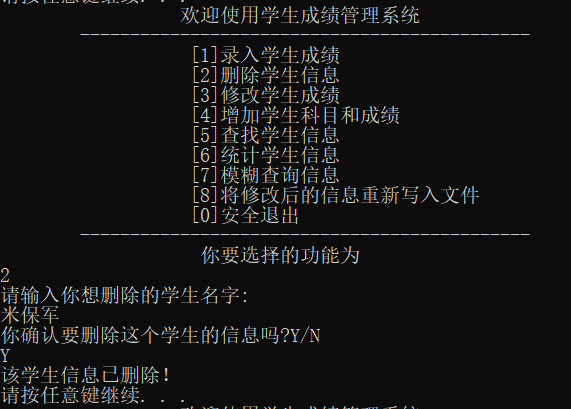
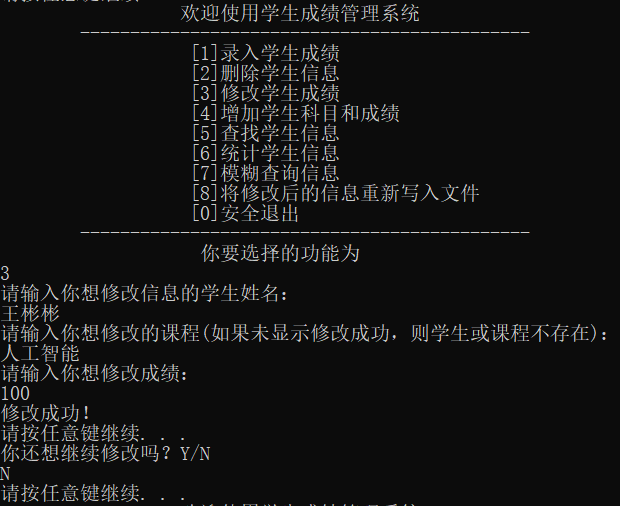
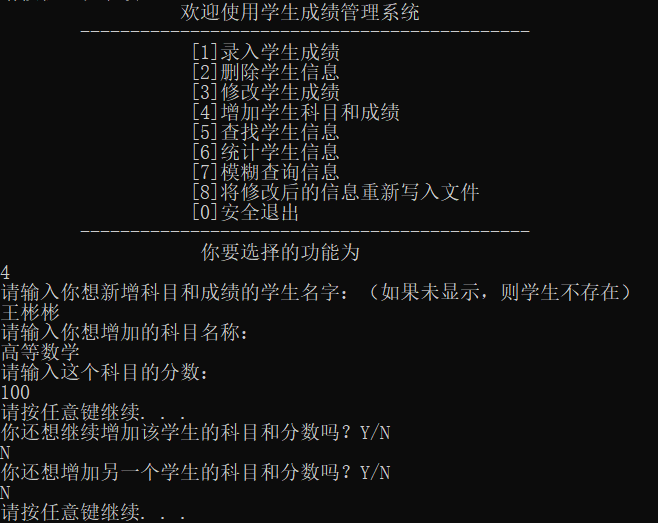
1. **测试**

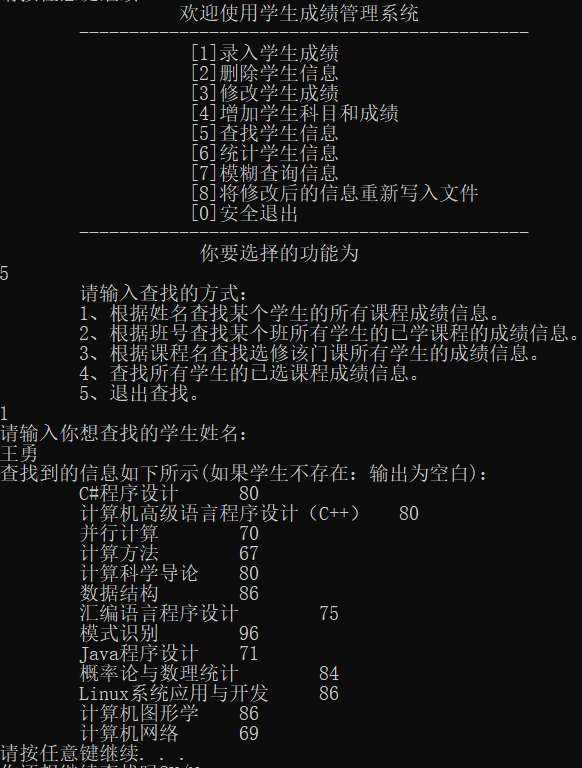


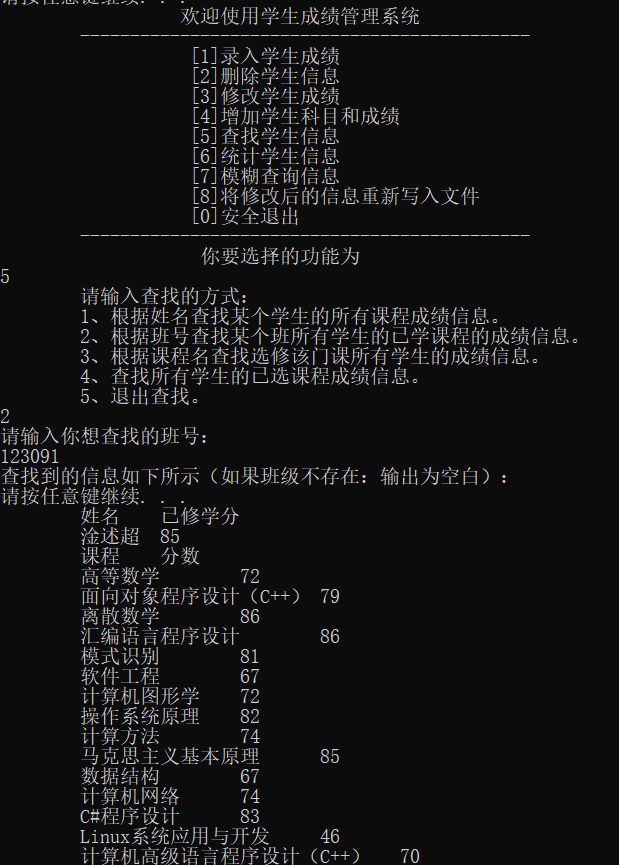
* 1. 录入学生成绩（input）：

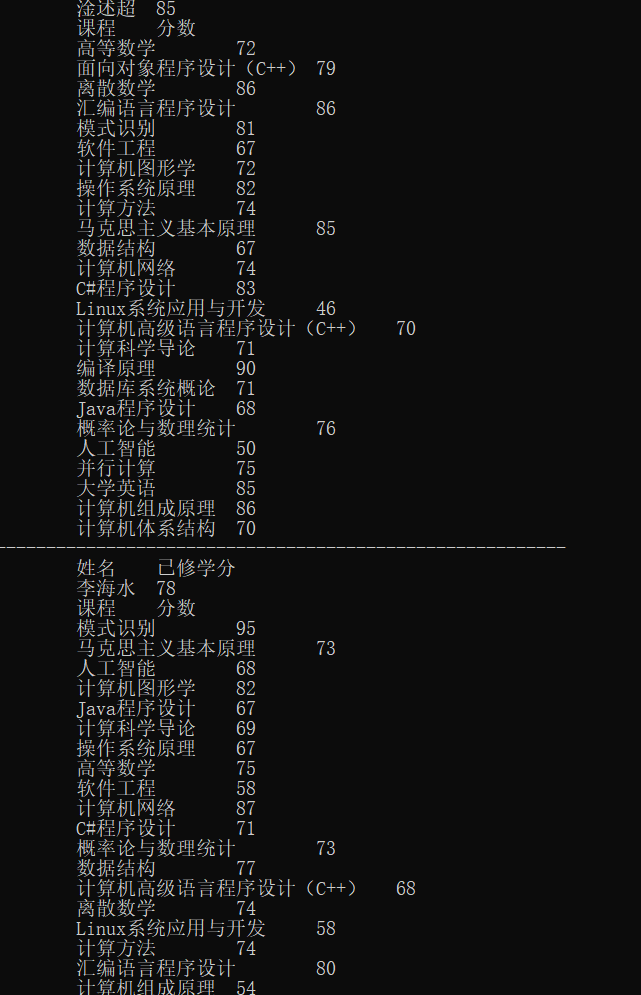




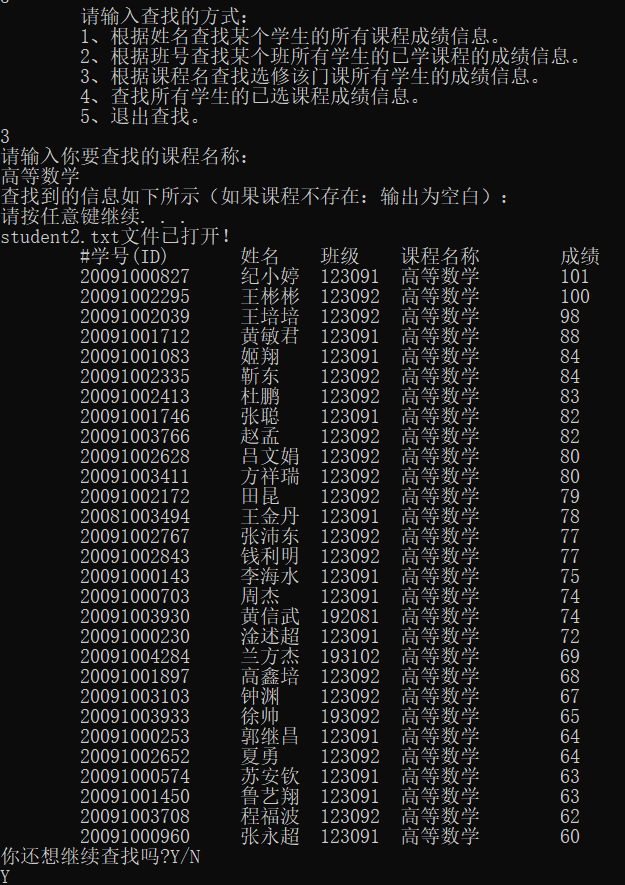
* 1. 删除学生信息（remove）：
  2. 修改学生成绩（change）：
  3. 增加学生科目和成绩（add）：
  4. 查找学生信息

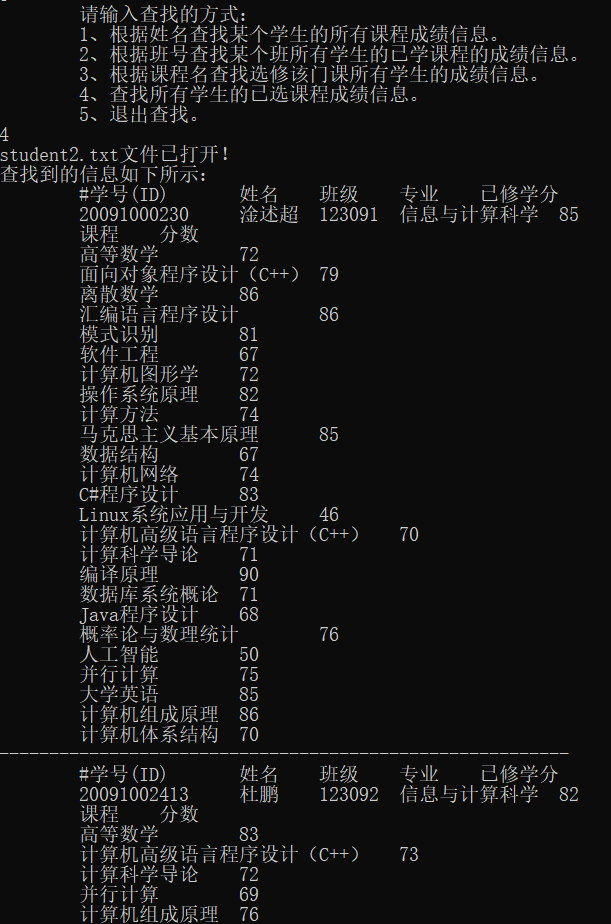
1. 根据姓名查找函数（search1）：
2. 根据班号查找函数（search2）：显示一部分结果。

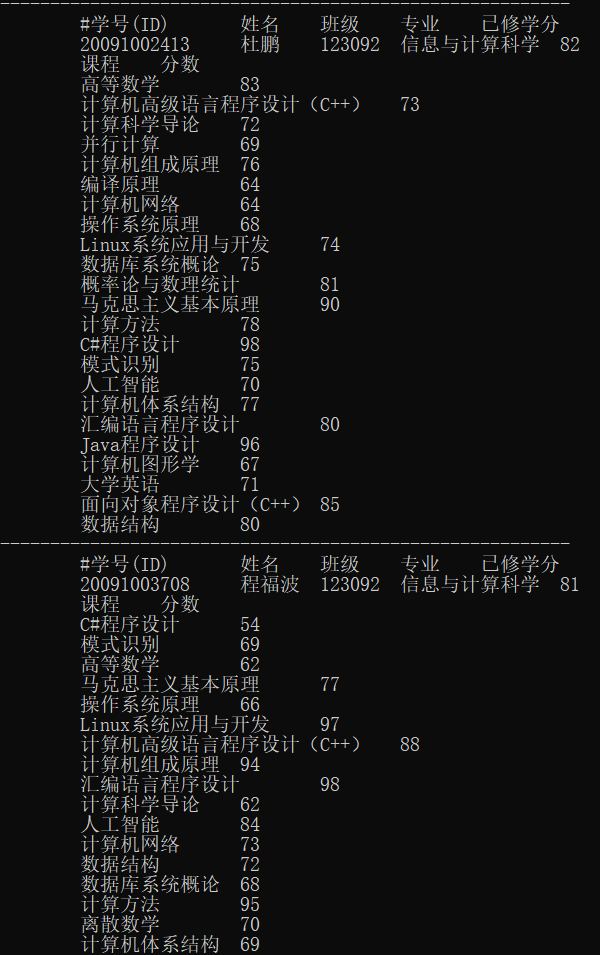




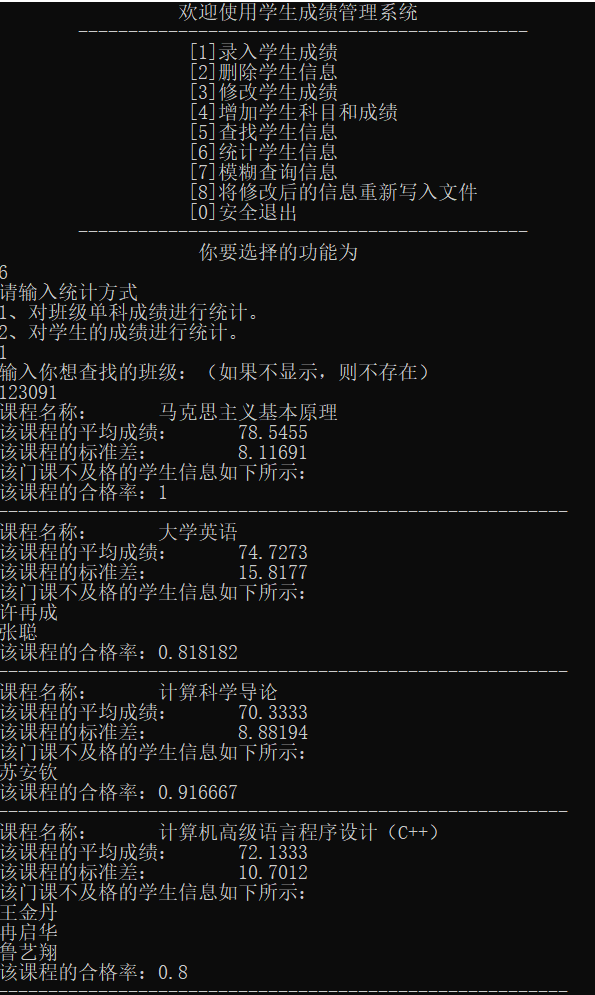
1. 根据课程名查找函数（search3）：

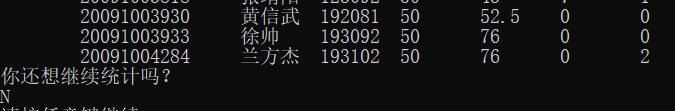


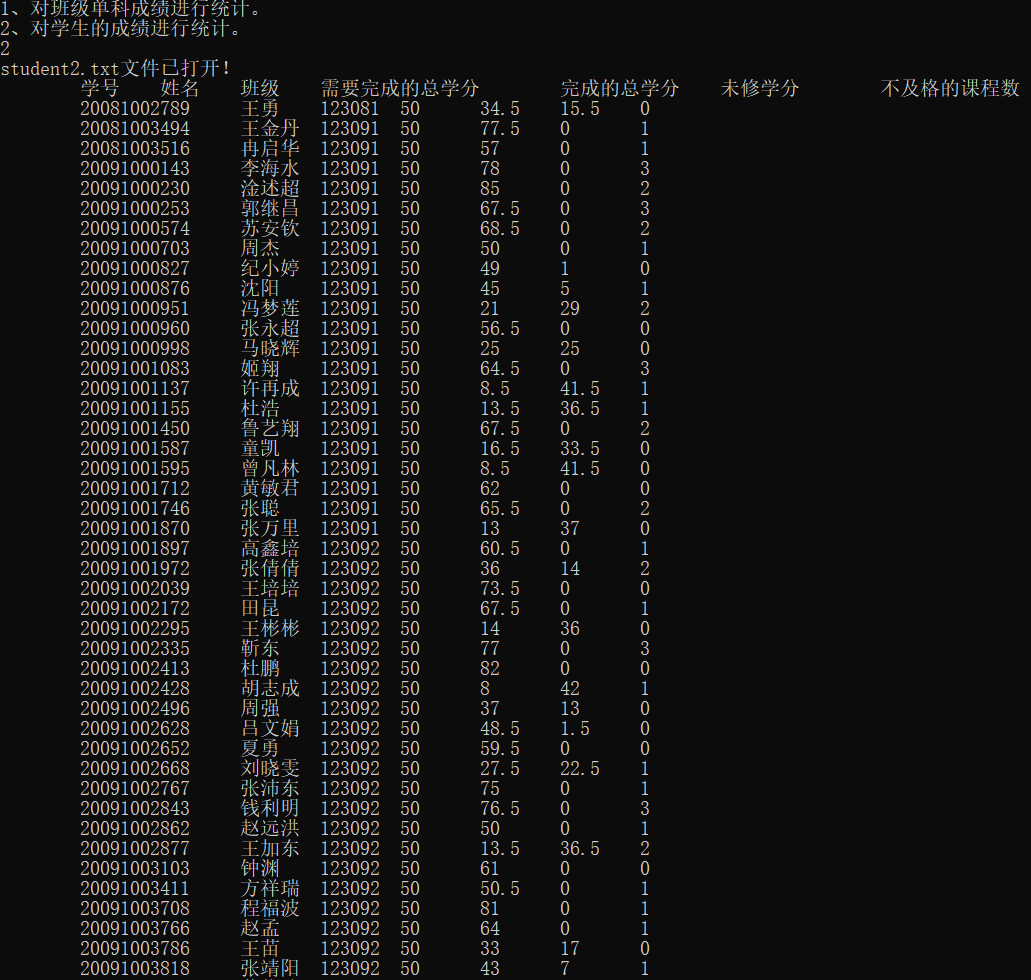
1. 查找所有信息函数（search4）：显示一部分结果。

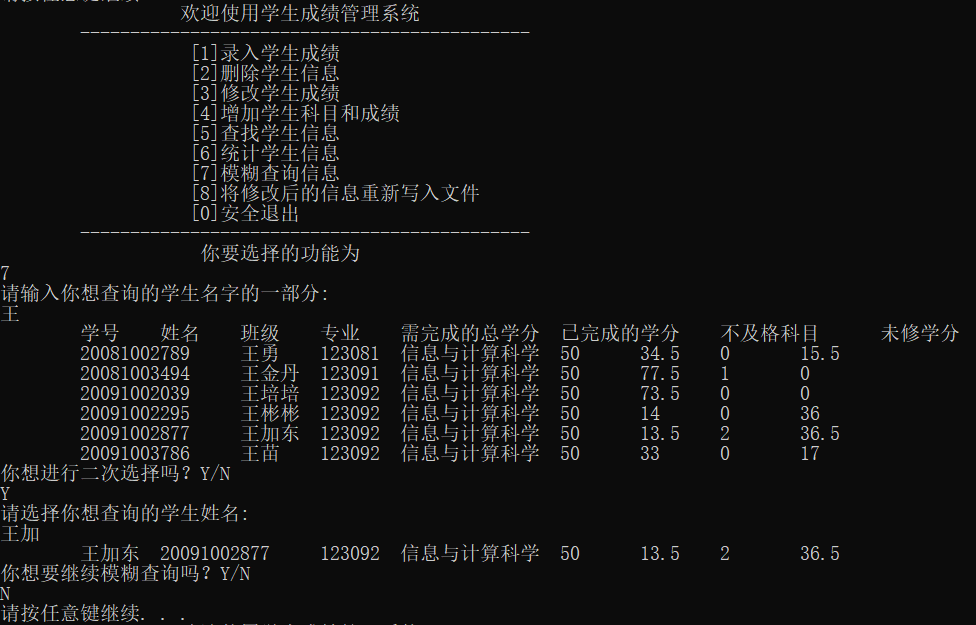


6、统计学生信息

* 1. 统计班级信息函数（statistic1）：显示一部分结果。

（2）统计学生信息函数（statistic2）：



7、模糊查询信息（fuzzy\_search）：

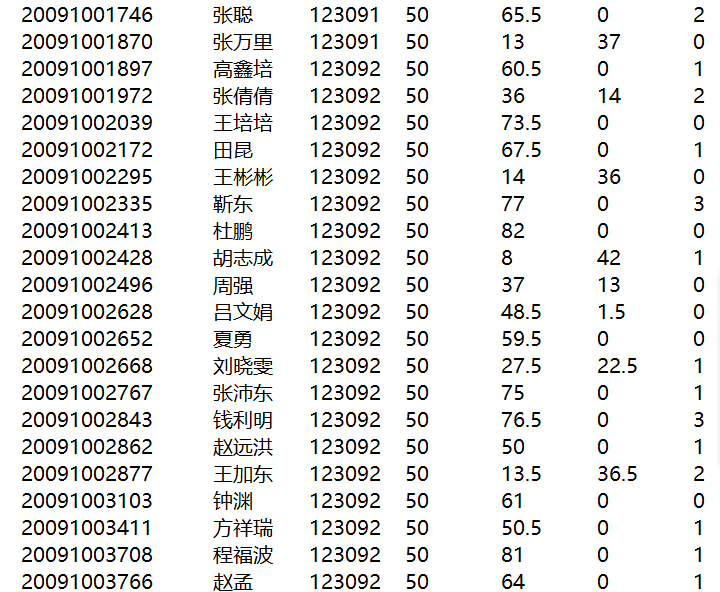
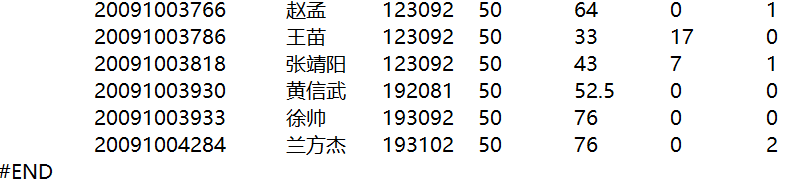
* 1. 将修改后的信息重新写入文件（write\_in）：显示一部分结果。

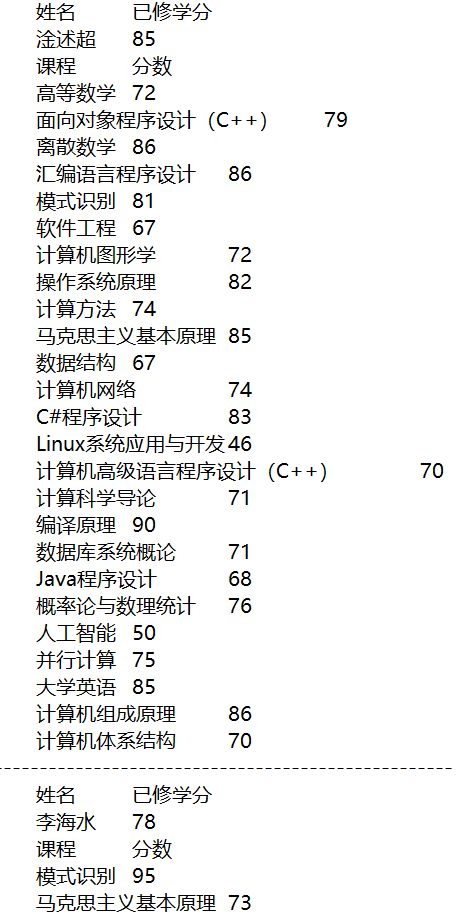


第一幅图片是写入之前，米保军相关信息存在，运行过程中，删除了米保军相关信息，重新写入文件后，如第二幅图片所示。

1. 文件操作：显示一部分结果。

班级统计后的数据写入文件student2.txt

学生成绩统计后写入文件student2.txt

排序好的信息写入文件student2.txt

**六、结论**

对系统开发的总结：从这次系统开发的过程中，我受到了很多启发。首先开发系统需要明确的思路，对于系统的框架要首先搭建好，先写出主函数和三个类，之后再对类的成员和模块进行填充，最后将内容逐步完善。其次是函数调用，比如排序模块，若不将默认排序函数写出来，代码是会变得很长，且会有很多地方会重复;而将默认排序函数写出来后，在需要的时候调用即可，解决了代码冗长的问题。最后是安全性问题，设计的三个类成员以及成员函数默认为私有，保障了数据安全性,将它们互相声明为友元也方便了三个类之间的访问。

存在的不足：较高要求模糊查询存在问题。首先，假设一个学生的名字有三个字,分别编为一二三，输入一、二、三、一二、二三均可进行模糊查询，但输入一三不能查询。其次，只实现了对名字的模糊查询，没有实现对课程的模糊查询。

**七、附录**

1、course.h

#ifndef COURSE\_H

#defineCOURSE\_H

#include<iostream>

#include<stdlib.h>

usingnamespace std;

classcourse {

friendclassstudent;

friendclassmanage;

private:

string m\_ID;//学号

string m\_number;//课程编号

string m\_name;//姓名

string m\_course;//课程名

double m\_score;//课程分数

double m\_credit;//课程学分

};

#endif

2、student.h

#ifndef STUDENT\_H

#defineSTUDENT\_H

#include<iostream>

#include<vector>

#include<string>

#include"course.h"

usingnamespace std;

classstudent {

friendclasscourse;

friendclassmanage;

private:

string m\_name;

string m\_ID;//ID

string m\_class;//班号

string m\_major;

double m\_total = 50;

double m\_completed = 0;//已完成的学分

double m\_uncompleted = 0;//未完成的学分

double m\_failed = 0;//成绩不及格科目

vector<course> m\_cs;//用来获取每个同学的科目以及分数即m\_course\_score

};

#endif;

3、manage.h

#ifndef MANAGE\_H

#defineMANAGE\_H

#include"student.h"

usingnamespace std;

classmanage {

friendclassstudent;

public:

vector<student>m\_a;

void input();//录入函数

void remove();//删除函数

void change();//修改函数

void add();//增加函数

void search1();//根据姓名查找

void search2();//根据班号查找

void search3();//根据课程名查找

void search4();//查找所有学生已选课程信息

void statistic1();//根据班级统计

void statistic2();//根据学生统计

void rank();//按学号默认排序

void fuzzy\_search();//模糊查找函数

void write\_in();//重新写入文件函数

};

#endif

4、main.cpp

#include"student.h"

#include"course.h"

#include"manage.h"

#include<iostream>

usingnamespace std;

int main() {

int key;

int k;

manage grades;

while (1) {

cout <<"\t 欢迎使用学生成绩管理系统 \n";

cout <<"\t---------------------------------------------\n";

cout <<"\t [1]录入学生成绩 \n";

cout <<"\t [2]删除学生信息 \n";

cout <<"\t [3]修改学生成绩 \n";

cout <<"\t [4]增加学生科目和成绩 \n";

cout <<"\t [5]查找学生信息 \n";

cout <<"\t [6]统计学生信息 \n";

cout <<"\t [7]模糊查询信息 \n";

cout <<"\t [8]将修改后的信息重新写入文件 \n";

cout <<"\t [0]安全退出 \n";

cout <<"\t---------------------------------------------\n";

cout <<"\t 你要选择的功能为 \n";

cin >> key;

if (key == 0) break;

switch (key) {

case 1:

grades.input();

system("pause");

break;

case 2:

grades.remove();

system("pause");

break;

case 3:

grades.change();

system("pause");

break;

case 4:

grades.add();

system("pause");

break;

case 5:

grades.rank();

char c;

do {

cout <<"\t请输入查找的方式：\n";

cout <<"\t1、根据姓名查找某个学生的所有课程成绩信息。\n";

cout <<"\t2、根据班号查找某个班所有学生的已学课程的成绩信息。\n";

cout <<"\t3、根据课程名查找选修该门课所有学生的成绩信息。\n";

cout <<"\t4、查找所有学生的已选课程成绩信息。\n";

cout <<"\t5、退出查找。\n";

cin >> k;

switch (k) {

case 1:

grades.search1();

break;

case 2:

grades.search2();

break;

case 3:

grades.search3();

break;

case 4:

grades.search4();

break;

case 5:

cout <<"已成功退出！"<< endl;

break;

default:

cout <<"没有此选项，请重新选择!"<< endl;

break;

}

if (k == 5)break;

cout <<"你还想继续查找吗?Y/N"<< endl;

cin >> c;

} while (c == 'Y' || c == 'y');

system("pause");

break;

case 6:

char h;

int i;

do {

cout <<"请输入统计方式"<< endl;

cout <<"1、对班级单科成绩进行统计。"<< endl;

cout <<"2、对学生的成绩进行统计。"<< endl;

cin >> i;

switch (i) {

case 1:

grades.statistic1();

break;

case 2:

grades.statistic2();

break;

default:

cout <<"没有此选项！"<< endl;

break;

}

cout <<"你还想继续统计吗？"<< endl;

cin >> h;

} while (h == 'Y' || h == 'y');

system("pause");

break;

case 7:

grades.rank();

grades.fuzzy\_search();

system("pause");

break;

case 8:

grades.write\_in();

system("pause");

break;

default:

cout <<"没有该选项，请重新选择！"<< endl;

break;

}

}

return 0;

}

5、manage.cpp

#include"manage.h"

#include<iostream>

#include<vector>

#include<cstdlib>

#include<iomanip>

#include<fstream>

#include<sstream>

void manage::input() {

ifstream ifs1;

ifs1.open("student.txt", ios::in);

if (!ifs1.is\_open()) {

cout << "student.txt文件未打开！" << endl;

}

else {

cout << "student.txt文件已打开！" << endl;

}

student temp1;

string buf1;

while (getline(ifs1, buf1))

{

if (buf1 == "#END")break;

if (buf1[0] < '0' || buf1[0]>'9')continue;

istringstream is(buf1);

is >> temp1.m\_ID >> temp1.m\_name >> temp1.m\_class >> temp1.m\_major;

m\_a.push\_back(temp1);

}

for (auto it = m\_a.begin(); it != m\_a.end(); it++) {

cout << (\*it).m\_ID << '\t' << (\*it).m\_name << '\t' << (\*it).m\_class << '\t' << (\*it).m\_major << endl;

}

ifs1.close();

ifstream ifs2;

ifs2.open("score.txt", ios::in);

if (!ifs2.is\_open()) {

cout << "score.txt文件未打开！" << endl;

}

else {

cout << "score.txt文件已打开！" << endl;

}

course temp2;

string buf2;

while (getline(ifs2, buf2))

{

if (buf2 == "#END")break;

if (buf2[0] < '0' || buf2[0]>'9') continue;

istringstream is(buf2);

is >> temp2.m\_ID >> temp2.m\_name >> temp2.m\_course >> temp2.m\_score;

for (int i = 0; i < m\_a.size(); ++i) {

if(temp2.m\_ID==m\_a[i].m\_ID)

m\_a[i].m\_cs.push\_back(temp2);

}

}

/\*for (int i = 0; i < m\_a.size(); ++i) {

for (auto it = m\_a[i].m\_cs.begin(); it != m\_a[i].m\_cs.end(); it++) {

cout << (\*it).m\_ID << '\t' << (\*it).m\_name << '\t' << (\*it).m\_course << '\t' << (\*it).m\_score << endl;

}

}\*/

ifs2.close();

ifstream ifs3;

ifs3.open("module.txt", ios::in);

if (!ifs3.is\_open()) {

cout << "module.txt文件未打开！" << endl;

}

else {

cout << "module.txt文件已打开！" << endl;

}

string m\_number;

string m\_course;

double m\_credit;

string buf3;

while (getline(ifs3, buf3))

{

if (buf3 == "#END")break;

if (buf3[0] < '0' || buf3[0]>'9')continue;

istringstream is(buf3);

is >> m\_number >> m\_course >> m\_credit;

for (int i = 0; i < m\_a.size(); ++i) {

for (int j = 0; j < m\_a[i].m\_cs.size(); ++j) {

if (m\_a[i].m\_cs[j].m\_course == m\_course) {

m\_a[i].m\_cs[j].m\_credit = m\_credit;

}

}

}

}

ifs3.close();

double sum(0),failed(0);

for (int i = 0; i < m\_a.size(); ++i) {

for(int j=0;j<m\_a[i].m\_cs.size();++j){

if (m\_a[i].m\_cs[j].m\_score >= 60) {

sum += m\_a[i].m\_cs[j].m\_credit;

}

else {

++failed;

}

}

m\_a[i].m\_completed = sum;

if (sum >= 50) m\_a[i].m\_uncompleted = 0;

else {

m\_a[i].m\_uncompleted = 50 - sum;

}

m\_a[i].m\_failed = failed;

sum = 0;

failed = 0;

}

}

void manage::remove() {

string s;

char c;

cout << "请输入你想删除的学生名字:" << endl;

cin >> s;

cout << "你确认要删除这个学生的信息吗?Y/N" << endl;

cin >> c;

if(c == 'Y' || c == 'y') {

for (int i = 0; i < m\_a.size(); ++i) {

if (m\_a[i].m\_name == s) {

for (int j = i + 1; j <m\_a.size(); ++j) {

swap(m\_a[j - 1], m\_a[j]);

}

}

}

if (m\_a[m\_a.size() - 1].m\_name != s) {

cout << "该学生不存在！" << endl;

}

else {

m\_a.pop\_back();

cout << "该学生信息已删除！" << endl;

}

}

}

void manage::change() {

string s, t;

char c;

cout << "请输入你想修改信息的学生姓名：" << endl;

cin >> s;

double score;

do {

cout << "请输入你想修改的课程(如果未显示修改成功，则学生或课程不存在)：" << endl;

cin >> t;

for (int i = 0; i <m\_a.size(); ++i) {

if (m\_a[i].m\_name == s) {

for (int j = 0; j<m\_a[i].m\_cs.size(); ++j) {

if (m\_a[i].m\_cs[j].m\_course == t) {

cout << "请输入你想修改成绩：" << endl;

cin >> score;

m\_a[i].m\_cs[j].m\_score = score;

cout << "修改成功！" << endl;

}

}

}

}

system("pause");

cout << "你还想继续修改吗？Y/N" << endl;

cin >> c;

} while (c == 'Y' || c == 'y');

}

void manage::add() {

string s;

course a;

char c, h;

do {

cout << "请输入你想新增科目和成绩的学生名字：（如果未显示，则学生不存在）" << endl;

cin >> s;

do {

for (int i = 0; i <m\_a.size(); ++i) {

if (m\_a[i].m\_name == s) {

cout << "请输入你想增加的科目名称：" << endl;

cin >> a.m\_course;

cout << "请输入这个科目的分数：" << endl;

cin >> a.m\_score;

a.m\_ID = m\_a[i].m\_ID;

a.m\_name = m\_a[i].m\_name;

ifstream ifs3;

ifs3.open("module.txt", ios::in);

if (!ifs3.is\_open()) {

cout << "module.txt文件未打开！" << endl;

}

string m\_number;

string m\_course;

double m\_credit;

string buf3;

while (getline(ifs3, buf3))

{

if (buf3 == "#END")break;

if (buf3[0] < '0' || buf3[0]>'9')continue;

istringstream is(buf3);

is >> m\_number >> m\_course >> m\_credit;

for (int j = 0; j < m\_a[i].m\_cs.size(); ++j) {

if (a.m\_course== m\_course) {

a.m\_credit= m\_credit;

if (a.m\_score >= 60) {

m\_a[i].m\_completed += m\_credit;

}

else {

++m\_a[i].m\_failed;

}

if (m\_a[i].m\_completed >= 50) {

m\_a[i].m\_uncompleted = 0; }

else {

m\_a[i].m\_uncompleted = 50 - m\_a[i].m\_completed;

}

}

}

}

ifs3.close();

m\_a[i].m\_cs.push\_back(a);

break;

}

}

system("pause");

cout << "你还想继续增加该学生的科目和分数吗？Y/N" << endl;

cin >> c;

} while (c == 'Y' || c == 'y');

cout << "你还想增加另一个学生的科目和分数吗？Y/N" << endl;

cin >> h;

} while (h == 'Y' || h == 'y');

}

void manage::rank() {

for (int i = 0; i < m\_a.size()-1; ++i) {

for (int j = m\_a.size()-1; j > i; --j) {

if (m\_a[j].m\_ID < m\_a[j-1].m\_ID) {

swap(m\_a[j].m\_ID, m\_a[j - 1].m\_ID);

}

}

}

}

void manage::search1() {

string s;

cout << "请输入你想查找的学生姓名：" << endl;

cin >> s;

cout << "查找到的信息如下所示(如果学生不存在：输出为空白)：" << endl;

for (int i = 0; i <m\_a.size(); ++i) {

if (m\_a[i].m\_name == s) {

for (auto &j : m\_a[i].m\_cs) {

cout <<'\t'<<j.m\_course <<'\t'<< j.m\_score << endl;

}

}

}

system("pause");

}

void manage::search2() {

string s;

cout << "请输入你想查找的班号：" << endl;

cin >> s;

cout << "查找到的信息如下所示（如果班级不存在：输出为空白）：" << endl;

vector<student>vi;

for (int i = 0; i < m\_a.size(); ++i) {

vi.push\_back(m\_a[i]);

}

for (int i = 0; i <vi.size()-1; ++i) {

for (int j =vi.size()-1; j > i; --j) {

if (vi[j].m\_completed>vi[j-1].m\_completed) {

swap(vi[j], vi[j-1]);

}

}

}

system("pause");

ofstream outfile;

outfile.open("student2.txt", ios::app);

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

cout << "student2.txt文件未打开！" << endl;

}

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

if (vi[i].m\_class == s) {

cout << '\t' << "姓名" << '\t' << "已修学分" << '\n';

cout << '\t' << vi[i].m\_name << '\t' << vi[i].m\_completed << '\n';

cout << '\t' << "课程" << '\t' << "分数" << '\n';

outfile << '\t' << "姓名" << '\t' << "已修学分" << '\n';

outfile << '\t' << vi[i].m\_name << '\t' << vi[i].m\_completed << '\n';

outfile << '\t' << "课程" << '\t' << "分数" << '\n';

for (auto &j:vi[i].m\_cs) {

cout<<'\t'<<j.m\_course<<'\t' << j.m\_score<<'\n';

outfile<< '\t' << j.m\_course << '\t' << j.m\_score << '\n';

}

cout << "---------------------------------------------------------\n";

outfile << "---------------------------------------------------------\n";

}

}

outfile.close();

}

void manage::search3() {

string s;

cout << "请输入你要查找的课程名称：" << endl;

cin >> s;

cout << "查找到的信息如下所示（如果课程不存在：输出为空白）：" << endl;

vector<course>a;

course c;

int k(0);

for (int i = 0; i <m\_a.size(); ++i) {

for (int j = 0; j <m\_a[i].m\_cs.size(); ++j) {

if (m\_a[i].m\_cs[j].m\_course == s) {

c.m\_ID = m\_a[i].m\_ID;

c.m\_name=m\_a[i].m\_name;

c.m\_number = m\_a[i].m\_class;

c.m\_course = s;

c.m\_score = m\_a[i].m\_cs[j].m\_score;

a.push\_back(c);

++k;

}

}

}

system("pause");

for (int i = 0; i < a.size()-1; ++i) {

for (int j = a.size() - 1; j > i; --j) {

if (a[j].m\_score > a[j - 1].m\_score)

swap(a[j], a[j - 1]);

}

}

ofstream outfile;

outfile.open("student2.txt", ios::app);

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

cout << "student2.txt文件未打开！" << endl;

}

else {

cout << "student2.txt文件已打开！" << endl;

}

cout << '\t' << "#学号(ID)" << '\t' << "姓名" << '\t' << "班级" << '\t' << "课程名称" << '\t' << "成绩" << '\n';

outfile<< '\t' << "#学号(ID)" << '\t' << "姓名" << '\t' << "班级" << '\t' << "课程名称" << '\t' << "成绩" << '\n';

for (auto& i : a) {

cout << '\t' << i.m\_ID << '\t' << i.m\_name << '\t' << i.m\_number << '\t' << i.m\_course << '\t' << i.m\_score << '\n';

outfile << '\t' << i.m\_ID << '\t' << i.m\_name << '\t' << i.m\_number << '\t' << i.m\_course << '\t' << i.m\_score << '\n';

}

outfile.close();

}

void manage::search4() {

ofstream outfile;

outfile.open("student2.txt", ios::app);

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

cout << "student2.txt文件未打开！" << endl;

}

else {

cout << "student2.txt文件已打开！" << endl;

}

cout << "查找到的信息如下所示：" << endl;

vector<student>vi;

for (int i = 0; i < m\_a.size(); ++i) {

vi.push\_back(m\_a[i]);

}

for (int i = 0; i < vi.size() - 1; ++i) {

for (int j = vi.size() - 1; j > i; --j) {

if (vi[j].m\_completed > vi[j - 1].m\_completed) {

swap(vi[j], vi[j - 1]);

}

}

}

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

cout << '\t' <<"#学号(ID)"<<'\t'<< "姓名"<<'\t'<<"班级" << '\t'<<"专业" <<'\t'<< "已修学分" << '\n';

outfile<< '\t' << "#学号(ID)" << '\t' << "姓名" << '\t' << "班级" << '\t' << "专业" << '\t' << "已修学分" << '\n';

cout <<'\t'<< vi[i].m\_ID<<'\t'<< vi[i].m\_name<<'\t'<< vi[i].m\_class <<'\t'<<vi[i].m\_major<<'\t'<< vi[i].m\_completed<<'\n';

outfile<<'\t' << vi[i].m\_ID << '\t' << vi[i].m\_name << '\t' << vi[i].m\_class << '\t' << vi[i].m\_major << '\t' << vi[i].m\_completed << '\n';

cout << '\t' << "课程" << '\t' << "分数" << '\n';

outfile << '\t' << "课程" << '\t' << "分数" << '\n';

for (int j = 0; j <vi[i].m\_cs.size(); ++j) {

cout << '\t' << vi[i].m\_cs[j].m\_course <<'\t'<< vi[i].m\_cs[j].m\_score << '\n';

outfile << '\t' << vi[i].m\_cs[j].m\_course << '\t' << vi[i].m\_cs[j].m\_score << '\n';

}

cout << "---------------------------------------------------------\n";

outfile << "---------------------------------------------------------\n";

}

outfile.close();

}

void manage::statistic1() {

double average = 0;//平均数

double dev = 0;//标准差

double pass = 0;//合格率

double sum = 0;

int k = 0;

int n = 0;

int q = 0;

vector<string>a;

ifstream ifs3;

ifs3.open("module.txt", ios::in);

if (!ifs3.is\_open()) {

cout << "module.txt文件未打开！" << endl;

}

string m\_number;

string m\_course;

double m\_credit;

string buf3;

while (getline(ifs3, buf3))

{

if (buf3 == "#END")break;

if (buf3[0] < '0' || buf3[0]>'9')continue;

istringstream is(buf3);

is >> m\_number >> m\_course >> m\_credit;

a.push\_back(m\_course);

}

ifs3.close();

ofstream outfile;

outfile.open("student2.txt", ios::app);

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

cout << "student2.txt文件未打开！" << endl;

}

string s;

cout << "输入你想查找的班级：（如果不显示，则不存在）" << endl;

cin >> s;

while (1) {

vector<int>b;

for (int i = 0; i < m\_a.size(); ++i) {

if (m\_a[i].m\_class == s) {

for (int j = 0; j < m\_a[i].m\_cs.size(); ++j) {

if (m\_a[i].m\_cs[j].m\_course == a[k]) {

sum += m\_a[i].m\_cs[j].m\_score;

b.push\_back(m\_a[i].m\_cs[j].m\_score);

++n;

}

}

}

}

average = sum / n;

for (int m = 0; m < b.size(); ++m) {

dev += pow(b[m] - average, 2);

}

dev = sqrt(dev / n);

cout << "课程名称：" << '\t' << a[k] << '\n';

outfile << "课程名称：" << '\t' << a[k] << '\n';

cout << "该课程的平均成绩：" << '\t' << average << '\n';

outfile << "该课程的平均成绩：" << '\t' << average << '\n';

cout << "该课程的标准差：" << '\t' << dev << '\n';

outfile << "该课程的标准差：" << '\t' << dev << '\n';

cout << "该门课不及格的学生信息如下所示：" << '\n';

outfile << "该门课不及格的学生信息如下所示：" << '\n';

for (int i = 0; i < m\_a.size(); ++i) {

if (m\_a[i].m\_class == s) {

for (int j = 0; j < m\_a[i].m\_cs.size(); ++j) {

if (m\_a[i].m\_cs[j].m\_course == a[k]) {

if (m\_a[i].m\_cs[j].m\_score < 60) {

cout << m\_a[i].m\_name << '\n';

++q;

}

}

}

}

}

cout << "该课程的合格率：" << (double)(n - q) / n << '\n';

outfile << "该课程的合格率：" << (double)(n - q) / n << '\n';

cout << "---------------------------------------------------------\n";

outfile << "---------------------------------------------------------\n";

++k;

sum = 0;

dev = 0;

n = 0;

q = 0;

if (k == 25) break;

}

}

void manage::statistic2(){

ofstream outfile;

outfile.open("student2.txt", ios::app);

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

cout << "student2.txt文件未打开！" << endl;

}

else {

cout << "student2.txt文件已打开！" << endl;

}

rank();

cout << '\t' << "学号" << '\t' << "姓名" << '\t' << "班级" << '\t'<<"需要完成的总学分"<<'\t' << "完成的总学分" << '\t' << "未修学分"

<< '\t' << "不及格的课程数" << '\n';

outfile<< '\t' << "#学号(ID)" << '\t' << "姓名" << '\t' << "班级" << '\t' << "需要完成的总学分" << '\t' << "完成的总学分" << '\t' << "未修学分"

<< '\t' << "不及格的课程数" << '\n';

for (int i = 0; i < m\_a.size(); ++i) {

cout << '\t' << m\_a[i].m\_ID << '\t' << m\_a[i].m\_name << '\t' << m\_a[i].m\_class << '\t' << m\_a[i].m\_total << '\t' << m\_a[i].m\_completed

<< '\t' << m\_a[i].m\_uncompleted<< '\t' << m\_a[i].m\_failed << '\n';

outfile<< '\t' << m\_a[i].m\_ID << '\t' << m\_a[i].m\_name << '\t' << m\_a[i].m\_class << '\t' << m\_a[i].m\_total << '\t' << m\_a[i].m\_completed

<< '\t' << m\_a[i].m\_uncompleted << '\t' << m\_a[i].m\_failed << '\n';

}

outfile<< "#END" << endl;

outfile.close();

}

void manage::fuzzy\_search() {

string c;

char r;

do {

cout << "请输入你想查询的学生名字的一部分:" << endl;

cin >> c;

vector<student>vi;

char a;

cout << '\t' << "学号" << '\t' << "姓名" << '\t' << "班级" << '\t' << "专业"

<< '\t' << "需完成的总学分" << '\t' << "已完成的学分" << '\t' << "不及格科目" << '\t' << "未修学分" << endl;

for (int i = 0; i < m\_a.size(); ++i) {

for (int j = 0; j < m\_a[i].m\_name.size(); ++j) {

if (m\_a[i].m\_name.find(c) != string::npos) {

cout << '\t' << m\_a[i].m\_ID << '\t' << m\_a[i].m\_name << '\t' << m\_a[i].m\_class << '\t' << m\_a[i].m\_major

<< '\t' << m\_a[i].m\_total << '\t' << m\_a[i].m\_completed << '\t' <<m\_a[i].m\_failed << '\t' << m\_a[i].m\_uncompleted << endl;

vi.push\_back(m\_a[i]);

break;

}

}

}

cout << "你想进行二次选择吗？Y/N" << endl;

cin >> a;

if (a == 'Y' || a == 'y') {

cout << "请选择你想查询的学生姓名:" << endl;

cin >> c;

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

if (vi[i].m\_name.find(c) != string::npos) {

cout << '\t' << vi[i].m\_name << '\t' << vi[i].m\_ID << '\t' << vi[i].m\_class << '\t' << vi[i].m\_major

<< '\t' << vi[i].m\_total << '\t' << vi[i].m\_completed << '\t' << vi[i].m\_failed << '\t' << vi[i].m\_uncompleted << endl;

break;

}

}

}

cout << "你想要继续模糊查询吗？Y/N" << endl;

cin >> r;

} while (r == 'Y' || r == 'y');

}

void manage::write\_in() {

ofstream outfile;

outfile.open("score.txt", ios::out);

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

cout << "score.txt文件未打开！" << endl;

}

outfile << '\t' << "#学号" << '\t' << "姓名" << '\t' << "课程名称" << '\t' << "成绩" << endl;

for (int i = 0; i < m\_a.size(); ++i) {

for (int j = 0; j < m\_a[i].m\_cs.size(); ++j) {

outfile << '\t' << m\_a[i].m\_cs[j].m\_ID << '\t' << m\_a[i].m\_cs[j].m\_name << '\t' << m\_a[i].m\_cs[j].m\_course <<

'\t' << m\_a[i].m\_cs[j].m\_score << '\n';

}

}

outfile << "#END";

cout << "已将信息写入！" << endl;

}

**原创性声明：**

本人声明报告中的内容和程序为本人独立完成，引用他人的文献、数据、图件、资料均已明确标出。除标注内容外，不包含任何形式的他人成果，无侵权和抄袭行为，并愿意承担由此产生的后果。

作者签字：

时间：2021.06.22

报告成绩

指导老师签字：