# 学生成绩管理系统

-------汇报稿

学校：嘉应学院（江北校区）

学院：计算机学院

制作人：谭绍洋

班级：2202班

完成时间：2023年2月3日

## 目录

1. **需求分析**-------------------------------------------------------------------------------------P3
2. **总体设计**-------------------------------------------------------------------------------------P4
3. 主体设计思路
4. 定义学生数据结构模块
5. 定义学生数据数组模块
6. 函数声明模块
7. int main(void)模块
8. void login()模块
9. void addStudent()模块
10. void delStudent()模块
11. void updateStudent()模块
12. void queryStudent()模块
13. void analyzeStudent()模块
14. void outputStudent()模块
15. **C语言源代码（系统实现）**--------------------------------------------------------P9
16. **操作流程（测试用例）**---------------------------------------------------------------P17

# 需求分析

**学生成绩管理系统应该包含的功能**

1. **登录功能**：允许管理员登录，登录成功后才能访问其他功能。
2. **录入学生信息**：可录入学生信息，例如：学号、姓名、班级、成绩。
3. **删除学生信息**：输入学号可删除该学生录入的全部信息。
4. **更新学生信息**：输入学号可修改该学生的姓名、班级、成绩。
5. **查询学生成绩**：输入学号可查询该学生该科成绩。
6. **分析学生成绩**：包括平均分、最高分、最低分、按分数高低打印及格与不及格名单及人数等。
7. **导出报表**：可将录入的学生信息导出到指定的文档文本中。
8. **退出系统**：输入指令可退出该系统。

# 总体设计

1. 主体设计思路

(1)首先，在main()函数中调用login()函数，用于管理者登录；

(2)然后，获取总分和及格线；

(3)接下来，在while循环中打印主菜单，根据用户需求在switch中调用相应的函数；

(4)比如，若输入为1，则调用addStudent()函数，用于录入学生信息；

(5)继续，若输入为2，则调用delStudent()函数，用于删除学生信息；

(6)接着，若输入为3，则调用updateStudent()函数，用于更新学生信息；

(7)然后，若输入为4，则调用queryStudent()函数，用于查询成绩；

(8)接下来，若输入为5，则调用analyzeStudent()函数，用于分析成绩；

(9)最后，若继续输入为6，则调用outputStudent()函数，用于导出报表；

(10)若输入为0，则退出程序。

1. 定义学生数据结构模块

首先定义一个Student结构体，用于存储学生信息，包括学生的学号、名字、班级和成绩。这个结构体可以用来存储和管理学生信息，方便程序操作和查询。

1. 定义学生数据数组模块

该模块定义一个学生数组，用于存储学生信息，以及宏定义，用来保存学生总数、总分、及格线、及格人数和不及格人数。它们可以用来记录和统计学生成绩，实现及格率等数据的计算。

1. 函数声明模块

这个模块声明了一组函数，用于管理学生信息，包括管理者登录，录入、删除、更新、查询、分析和导出学生信息。这些函数可以用来实现对学生信息的完整管理，并方便之后的调用。

1. int main(void)模块

(1)登录系统：首先调用login()函数进行登录；

(2)输入总分和及格线：输入该科总分和及格分数；

(3)打印主菜单：根据用户输入的不同数字，调用不同的功能函数，实现添加、删除、更新、查询、分析和导出报表等功能；

(4)退出系统：输入0，退出系统。

1. void login()模块

(1)输入管理员密码：输入管理员密码；

(2)比对密码：用strcmp函数比对用户输入的密码和系统设定的密码；

(3)判断登录结果：如果密码相同，则登录成功；如果密码不同， 则登录失败，终止程序运行。

1. void addStudent()模块

该模块的作用是用来添加学生的信息，具体步骤如下：

(1)使用malloc函数在堆内存中为指针stu分配一块存储Student类型数据的内存空间；

(2)读取学生的学号和姓名，并判断是否重复输入；

(3)如果输入学号重复，则提示添加失败，并释放stu指向的动态内存；

(4)读取学生的班级和成绩；

(5)如果学生成绩大于总分，则提示添加失败，并释放stu指向的动态内存；

(6)如果学生成绩小于等于总分，则将stu指向的结构体内容赋值给students[stuNum]，学生数量加一，提示添加成功；

(7)最后释放stu指向的动态内存，避免内存泄漏。

1. void delStudent()模块

该模块的作用是删除学生信息。

(1)首先请用户输入要删除的学生学号。

(2)然后使用循环遍历students数组，在数组中找到学号与输入的相同的学生，将最后一个学生数据覆盖到当前位置，并将学生总数stuNum减一。

(3)最后输出删除学生成功的信息，并退出函数delStudent()。

1. void updateStudent()模块

该模块的作用是更新学生信息。

(1)首先请用户输入要更新的学生学号。

(2)然后使用循环遍历students数组，在数组中找到学号与输入的相同的学生，重新输入学生的姓名、班级、成绩等信息。

(3)最后输出更新学生成功的信息，并退出函数updateStudent()。

1. void queryStudent()模块

该模块的作用是查询学生信息。

(1)首先，在函数queryStudent()中，通过printf()函数打印提示，要求用户输入要查询的学生学号；

(2)用scanf()函数获取用户输入的学号，将其保存到变量query\_id中；

(3)然后，用一个for循环，依次检查数组students[]中的每一个元素，如果students[i].id == query\_id，即学号相同，则说明找到了要查询的学生；

(4)最后，调用printf()函数，打印学生的学号、姓名、班级和成绩，之后跳出queryStudent()函数，结束查询。

1. void analyzeStudent()模块

该模块的作用是分析学生的成绩。

(1)首先判断学生信息是否已经录入，如果已经录入，则进行下一步，否则提示还未录入学生信息。

(2)统计总分，并找出最高分和最低分。

(3)对学生的成绩进行冒泡排序，从高到低排序。

(4)打印及格名单，以及不及格名单，并统计及格人数和不及格人数，最后计算及格率。

1. void outputStudent()模块

该模块的作用是导出报表。

(1)定义函数outputStudent，用于把学生信息导出到student.txt中。

(2)判断stuNum是否为0，如果为0，则表明尚未录入学生信息，输出提示并中断函数执行，跳出outputStudent()函数。

(3)创建student.txt文本文档，以只写的形式打开，如果文件创建失败，则输出提示并中断函数执行，跳出outputStudent()函数。

(4)在student.txt中输出学号、姓名、班级和成绩的标题，以便用户查看。

(5)使用循环语句，将学生信息写入student.txt文件，以便用户查看。

(6)最后关闭文件，提示报表已导出，结束函数outputStudent() 的执行。

# C语言源代码（系统实现）

/\*

学生成绩管理系统应该包含的功能

1.登录功能：允许管理员登录，登录成功后才能访问其他功能。

2.录入学生信息：可录入学生信息，例如：学号、姓名、班级、成绩。

3.删除学生信息：输入学号可删除该学生录入的全部信息。

4.更新学生信息：输入学号可修改该学生的姓名、班级、成绩。

5.查询学生成绩：输入学号可查询该学生该科成绩。

6.分析学生成绩：包括平均分、最高分、最低分、按分数高低打印及格与不及格名单及人数等。

7.导出报表：可将录入的学生信息导出到指定的文档文本中。

8.退出系统：输入指令可退出该系统。

能实现以上功能的C语言完整代码：

\*/

#include <stdio.h>

#include <stdlib.h>

//定义学生数据结构

typedef struct Student

{

int id;

char name[20];

char class[20];

float score;

} Student;

//定义学生数据数组

Student students[100];

int stuNum;//宏定义学生总数

float all;//宏定义总分

int pass;//宏定义及格线

int sumpass;//宏定义及格人数

int sumunpass;//宏定义不及格人数

//函数声明

void login();//管理者登录

void addStudent();//录入学生信息

void delStudent();//删除学生信息

void updateStudent();//更新学生信息

void queryStudent();//查询成绩

void analyzeStudent();//分析成绩

void outputStudent();//导出报表

int main(void)

{

login();//引用登录函数

printf("---------欢迎使用本系统！---------\n");//修饰

printf("请输入该科总分：");

scanf("%f",&all);//录入总分

printf("及格线：");

scanf("%d",&pass);//录入及格分数

while (1)//打印主菜单

{

int sel;

printf("\t\n");

printf("请选择操作：\n");

printf(" ---------------\n");

printf("| 1. 添加学生 |\n");

printf("| 2. 删除学生 |\n");

printf("| 3. 更新学生 |\n");

printf("| 4. 查询学生 |\n");

printf("| 5. 分析学生 |\n");

printf("| 6. 导出报表 |\n");

printf("| 0. 退出 |\n");

printf(" ---------------\n");

scanf("%d", &sel);

switch (sel) //根据输入引用对应功能

{

case 1: addStudent(); break;

case 2: delStudent(); break;

case 3: updateStudent(); break;

case 4: queryStudent(); break;

case 5: analyzeStudent(); break;

case 6: outputStudent(); break;

case 0: return 0;

default: printf("输入有误，请重新输入！\n"); break;

}

}

return 0;

}

//登录功能

void login()

{

char password[20];

printf("请输入管理员密码：\n");

scanf("%s", password);

if (strcmp(password, "123456") == 0)//用stringcompare比对密钥

{

printf("登录成功！\n");

printf("\t\n");

}

else

{

printf("密码错误，登录失败！\n");

exit(0);//终止程序运行，正常退出程序

}

}

//添加学生

void addStudent()

{

Student \*stu = (Student \*)malloc(sizeof(Student));

//动态内存分配语句，为指针stu分配一块存储Student类型数据的内存空间。

printf("请输入学生学号：");

scanf("%d", &stu->id);

///学号具有唯一性，判断是否重复输入。

int flag = 0;

int i;

for(i = 0; i < stuNum; i++)

{

if(students[i].id == stu->id)

{

flag = 1;

break;

}

}

if(flag == 1)

{

printf("学号重复，添加失败！\n");

free(stu);

return;

}

///

printf("请输入学生姓名：");

scanf("%s", stu->name);

printf("请输入学生班级：");

scanf("%s", stu->class);

printf("请输入学生成绩(分)：");

scanf("%f", &stu->score);

if(stu->score > all)

{

printf("学生成绩大于总分！添加失败！\n");

free(stu);//释放stu所指向的动态内存，即释放掉动态开辟的内存空间，避免内存泄漏。

return;//中断函数执行，跳出addStudent()函数

}

else

{

students[stuNum] = \*stu;//将stu所指向的结构体内容赋值给students[stuNum]，以添加新的学生信息

stuNum++;//学生数量加一，记录学生总数量

printf("\t\n");

printf("添加成功！\n");

}

free(stu);//释放stu所指向的动态内存，即释放掉动态开辟的内存空间，避免内存被无限占用。

}

//删除学生

void delStudent()

{

int i, del\_id;

printf("请输入要删除的学生学号：");

scanf("%d", &del\_id);

for (i = 0; i < stuNum; i++)

{

// 找到学号相同的学生

if (students[i].id == del\_id)

{

// 将最后一个学生数据覆盖到当前位置

students[i] = students[stuNum - 1];

//学生的索引号从0开始，所以最后一个学生的索引号应该是stuNum-1，而不是stuNum。

stuNum--;// 学生数量减一

printf("\t\n");

printf("删除学生成功！\n");

return;//中断函数执行，跳出delStudent()函数

}

}

printf("未找到该学号！\n");

}

//更新学生

void updateStudent()

{

int i, update\_id;

printf("请输入要更新的学生学号：");

scanf("%d", &update\_id);

for (i = 0; i < stuNum; i++)

{

// 找到学号相同的学生,重新输入即可

if (students[i].id == update\_id)

{

printf("请输入更新后的学生姓名：");

scanf("%s", students[i].name);

printf("请输入更新后的学生班级：");

scanf("%s", students[i].class);

printf("请输入更新后的学生成绩：");

scanf("%f", &(students[i].score));

printf("\t\n");

printf("更新学生成功！\n");

return;//中断函数执行，跳出updateStudent()函数

}

}

printf("未找到该学号！\n");

}

//查询学生

void queryStudent()

{

int i, query\_id;

printf("请输入要查询的学生学号：");

scanf("%d", &query\_id);

for (i = 0; i < stuNum; i++)

{

// 找到学号相同的学生

if (students[i].id == query\_id)

{

printf("\t\n");

printf("学号：%d\n", students[i].id);

printf("姓名：%s\n", students[i].name);

printf("班级：%s\n", students[i].class);

printf("成绩：%.2f分\n", students[i].score);

return;//中断函数执行，跳出queryStudent()函数

}

}

printf("未找到该学号！\n");

}

//分析学生

void analyzeStudent()

{

if(stuNum!=0)//判断是否已经录入学生信息

{

int i;

float sum = 0;

float max = 0;

for (i = 0; i < stuNum; i++)

{

sum += students[i].score;//统计总分

if (students[i].score > max)

{

max = students[i].score;//找出最高分

}

if (students[i].score < all)

{

all = students[i].score;//找出最低分

}

}

printf("\t\n");

printf("平均成绩：%.2f分\n", sum / stuNum);

printf("最高成绩：%.2f分\n", max);

printf("最低成绩：%.2f分\n", all);

//冒泡排序成绩

int j;

for (i = 0; i < stuNum - 1; i++)

{

for (j = 0; j < stuNum - i - 1; j++)

{

if (students[j].score < students[j + 1].score)

{

Student temp = students[j];

students[j] = students[j + 1];

students[j + 1] = temp;

}

}

}

printf("从高到低排序后的学生成绩：\n");

printf("及格名单\n");

printf("学号\t姓名\t班级\t成绩\n");

sumpass=0;

sumunpass=0;

for (i = 0; i < stuNum; i++)

{

if(students[i].score >= pass)//打印合格名单

{

sumpass = sumpass + 1;

printf("%d\t%s\t%s\t%.2f\n",students[i].id,students[i].name,students[i].class,students[i].score);

}

}

printf("不及格名单\n");

printf("学号\t姓名\t班级\t成绩\n");

for (i = 0; i < stuNum; i++)

{

if(students[i].score < pass)//打印不合格名单

{

sumunpass = sumunpass + 1;

printf("%d\t%s\t%s\t%.2f\n",students[i].id,students[i].name,students[i].class,students[i].score);

}

}

printf("及格人数：%d 人\n",sumpass);

printf("不及格人数：%d 人\n",sumunpass);

printf("及格率为：%.2f%%\n",(float)sumpass/stuNum \* 100);

}

else printf("您还未录入学生成绩！请录入学生后再分析！\n");

}

//导出报表

void outputStudent()

{

if(stuNum!=0)//判断是否已经录入学生信息

{

FILE \*fp = fopen("student.txt", "w");//只写形式创建student.txt文本文档

if (fp == NULL)

{

printf("创建文件失败！\n");

return;//中断函数执行，跳出outputStudent()函数

}

int i;

fprintf(fp, "学号\t姓名\t班级\t成绩\n");

for (i = 0; i < stuNum; i++)

{

fprintf(fp, "%d\t%s\t%s\t%.2f\n", students[i].id, students[i].name, students[i].class, students[i].score);

}

printf("报表已导出！\n");

fclose(fp);

}

else printf("请录入学生信息再导出报表！\n");

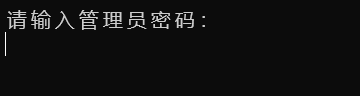
}

# 操作流程（测试用例）

**使用流程建议：**

**登录 --> 录入总分、及格线 --> 录入学生信息 --> 查询/分析/更新/删除/导出**

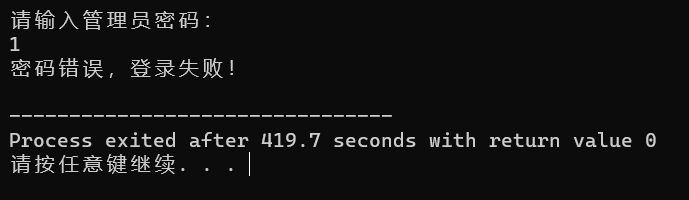
运行该系统，您将看到以下提示：



默认管理员密码为：123456

每次输入完成，请摁键盘上的“Enter”键完成输入。

如果密码错误，则会提示：“密码错误，登录失败！”，并退出本系统，如下图：



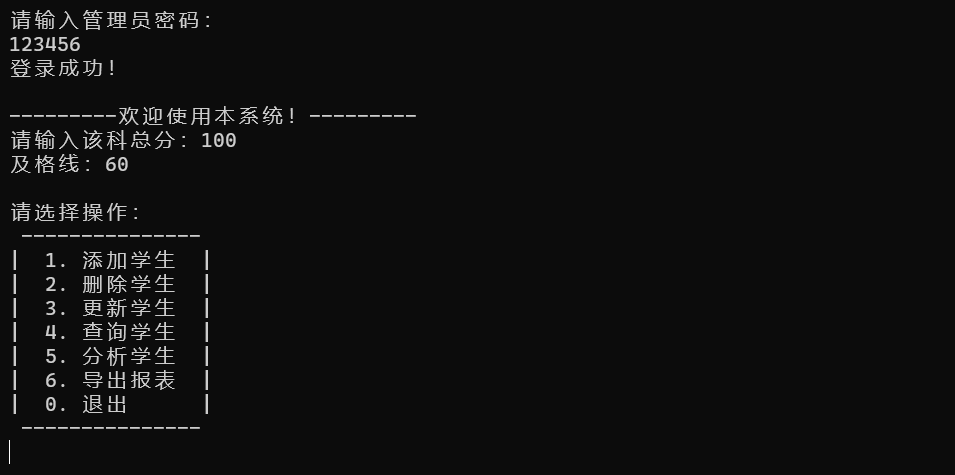
如果密码正确，则会提示：“登录成功！”，并提示输入该科总分，如下图：



输入该科总分：100（测试用例），系统会提示您再输入及格线分数：60（测试用例），如下图：

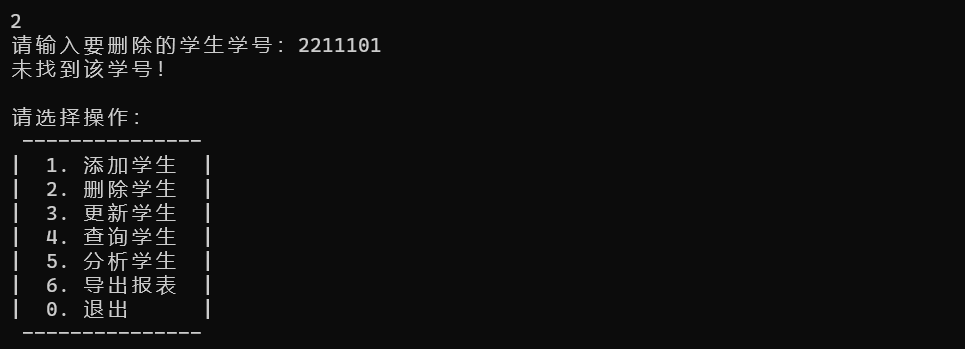


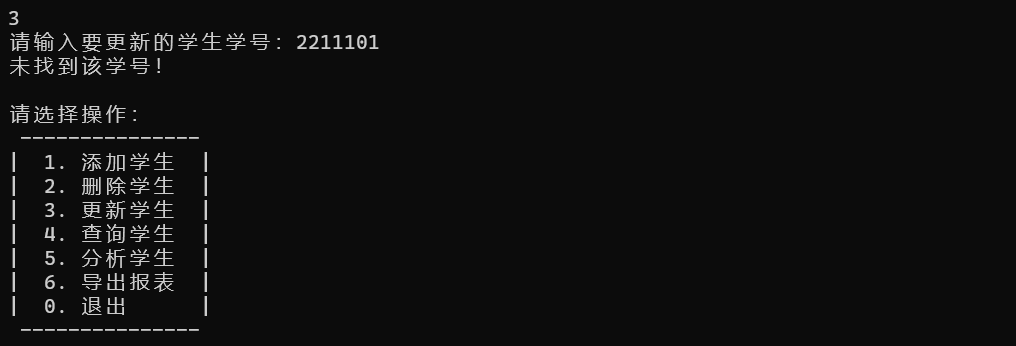
输入及格线分数（学生分数=及格线分数视为及格）后，您会看到系统主菜单页，并提示您选择操作，如下图：

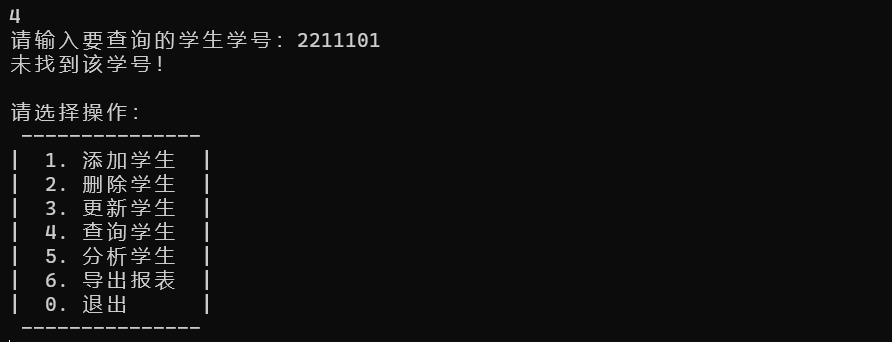


选择操作时，只需要输入功能前的阿拉伯数字“1”“2”“3”······就行。

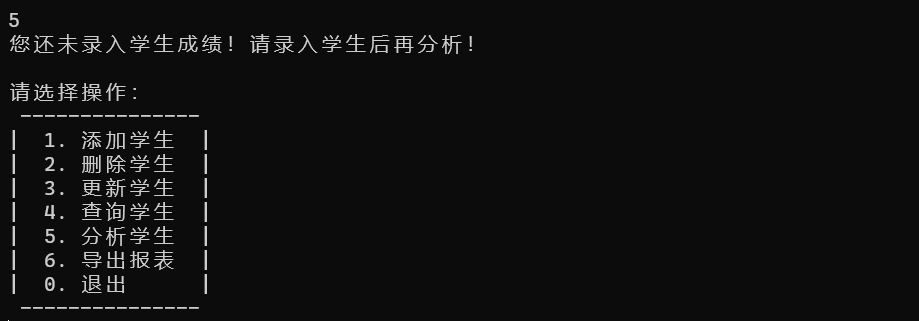
如果没有添加学生信息，而选择“删除学生”“更新学生”“查询学生”功能，当您输入学号2211101（测试用例）后，您会被提示“未找到该学号！”，如下图：



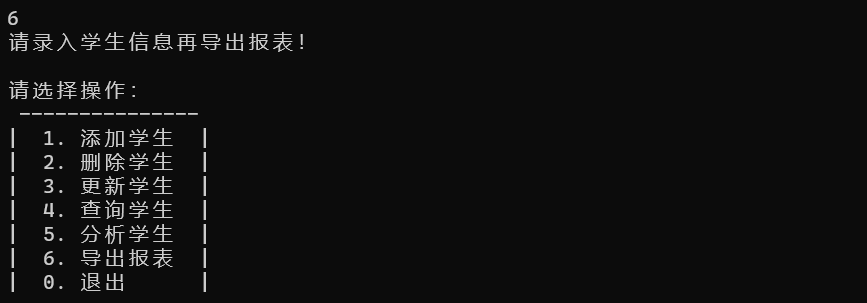




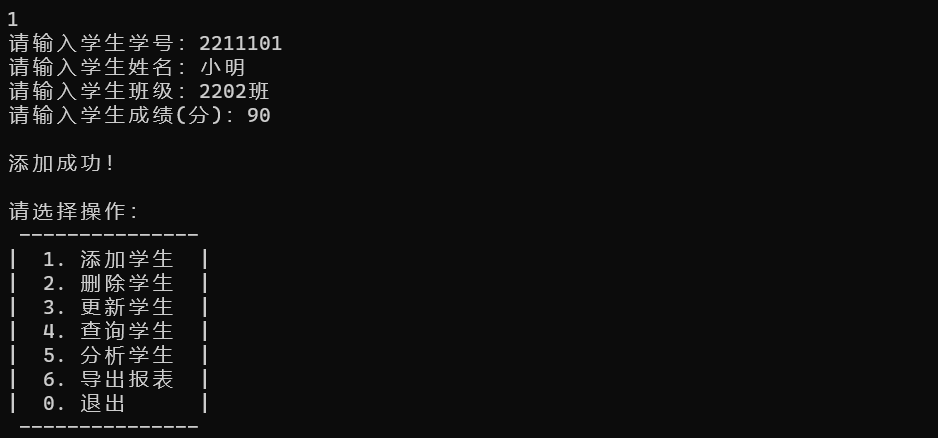
如果没有添加学生信息，而选择“分析学生”功能，您会被提示“您还未录入学生成绩！请录入学生后再分析！”，如下图：



如果没有添加学生信息，而选择“导出报表”功能，您会被提示“请录入学生信息再导出报表！”，如下图：



您要先添加学生信息才能使用其它功能，输入“1”选择“添加学生”功能，按照系统提示逐个输入学生信息，包括学号：2211101，姓名：小明，班级：2202班，分数：90（以上均为测试用例）。当您输入完毕，系统会提示“添加成功！”，如下图：

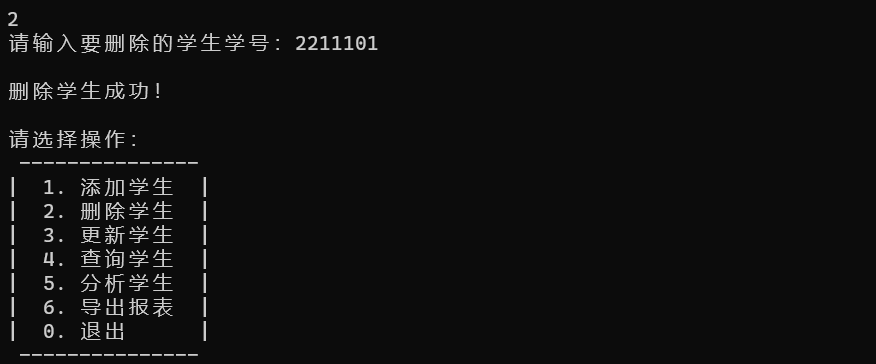


如果在已经添加了至少1个学生信息的情况下，输入与已录入学生信息相同的学号2211101（测试用例）时，系统会提示“学号重复，添加失败！”，因为在该系统中学号具有唯一性，此时，您应该重新添加不同学号的学生信息，如下图：

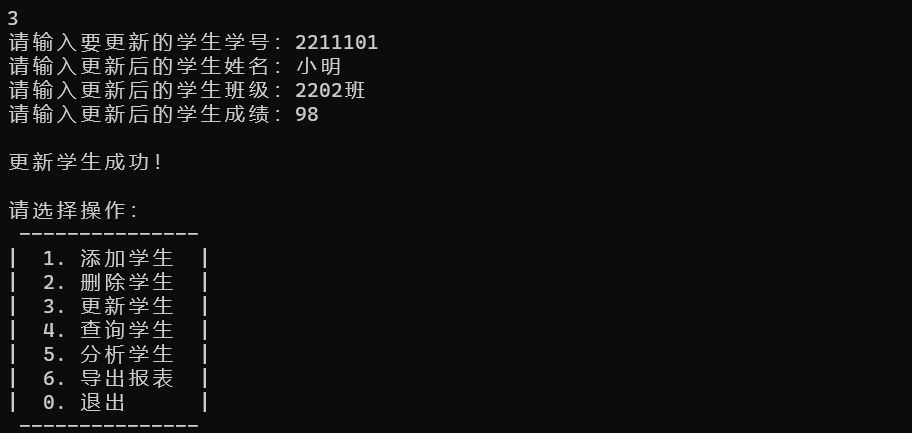


注意，添加学生信息时，本系统只能一个一个学生添加，不能批量录入。如果您的电脑磁盘内存足够大，理论上，可以录入无限个学生信息。当您录入完毕，此时您才可以正常使用本系统的其它功能。

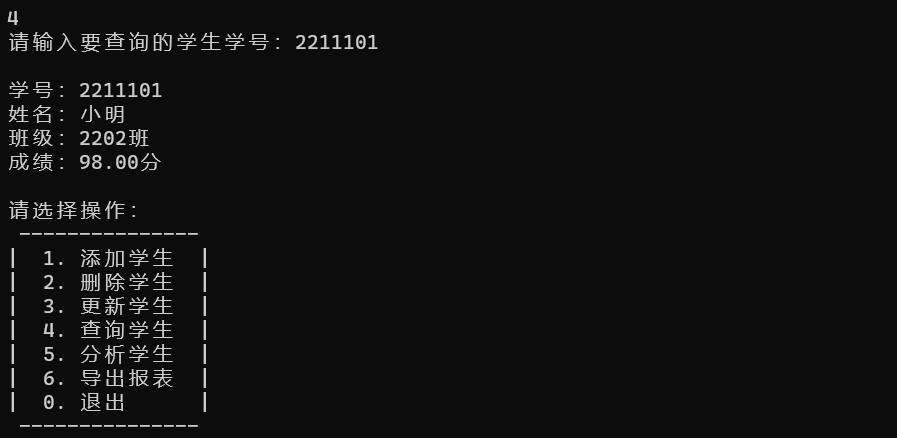
如“删除学生”，当您输入要删除的学生学号2211101（测试用例）后，系统会提示“删除学生成功！”，如下图：



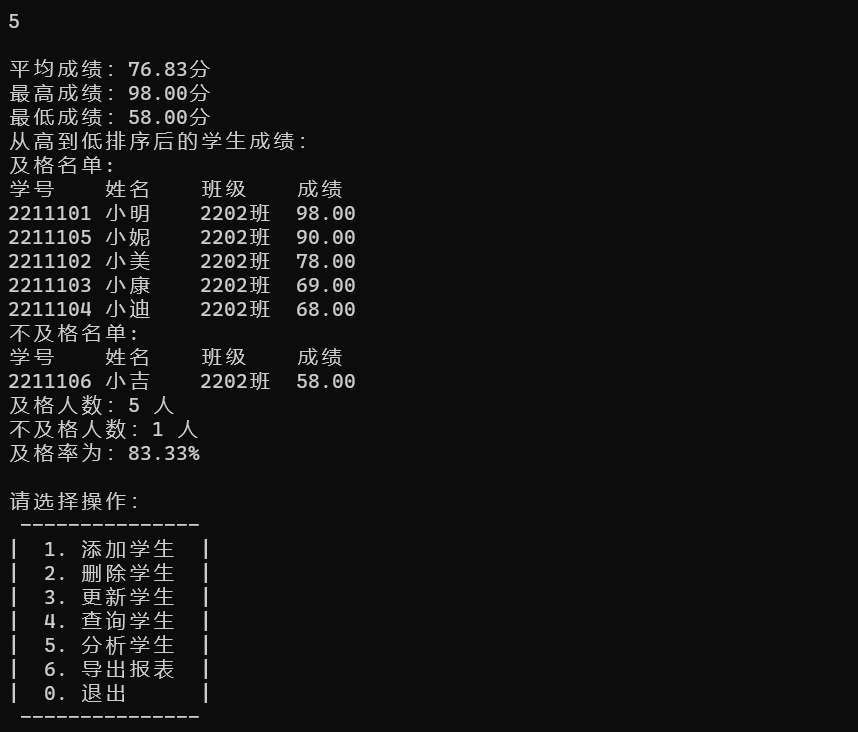
如“更新学生”，当您输入要更新的学生学号2211101（测试用例）后，请按系统提示修改学生姓名、班级、成绩信息（测试用例仅修改成绩），全部输入完毕后，系统会提示您“更新学生成功！”，如下图：



如“查询学生”，如果您想知道某一学生的信息时，您只需要输入该学生的学号2211101（测试用例），您就会的到该学生的姓名、班级、成绩信息。如下图：



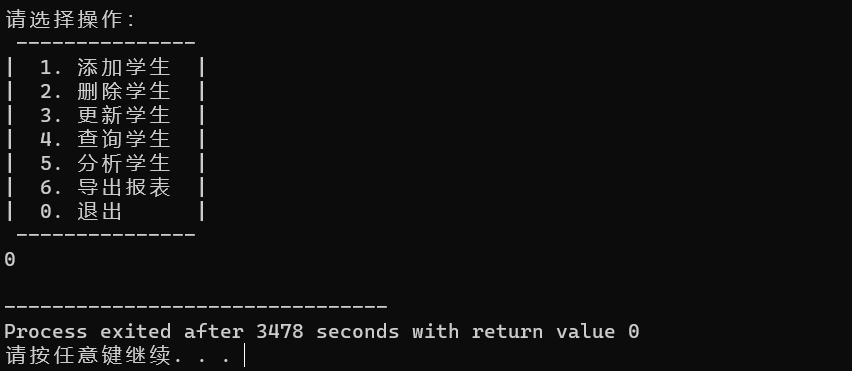
如“分析学生”，如下图：



如“导出报表”，如果您的电脑磁盘内存不足，则有可能无法正常导出。如果您的电脑磁盘内存足够，系统则会提示“报表已导出！”，本系统的报表是一个叫“student.txt”的文档文本，您可以在“我的电脑”中查找到。如下图：

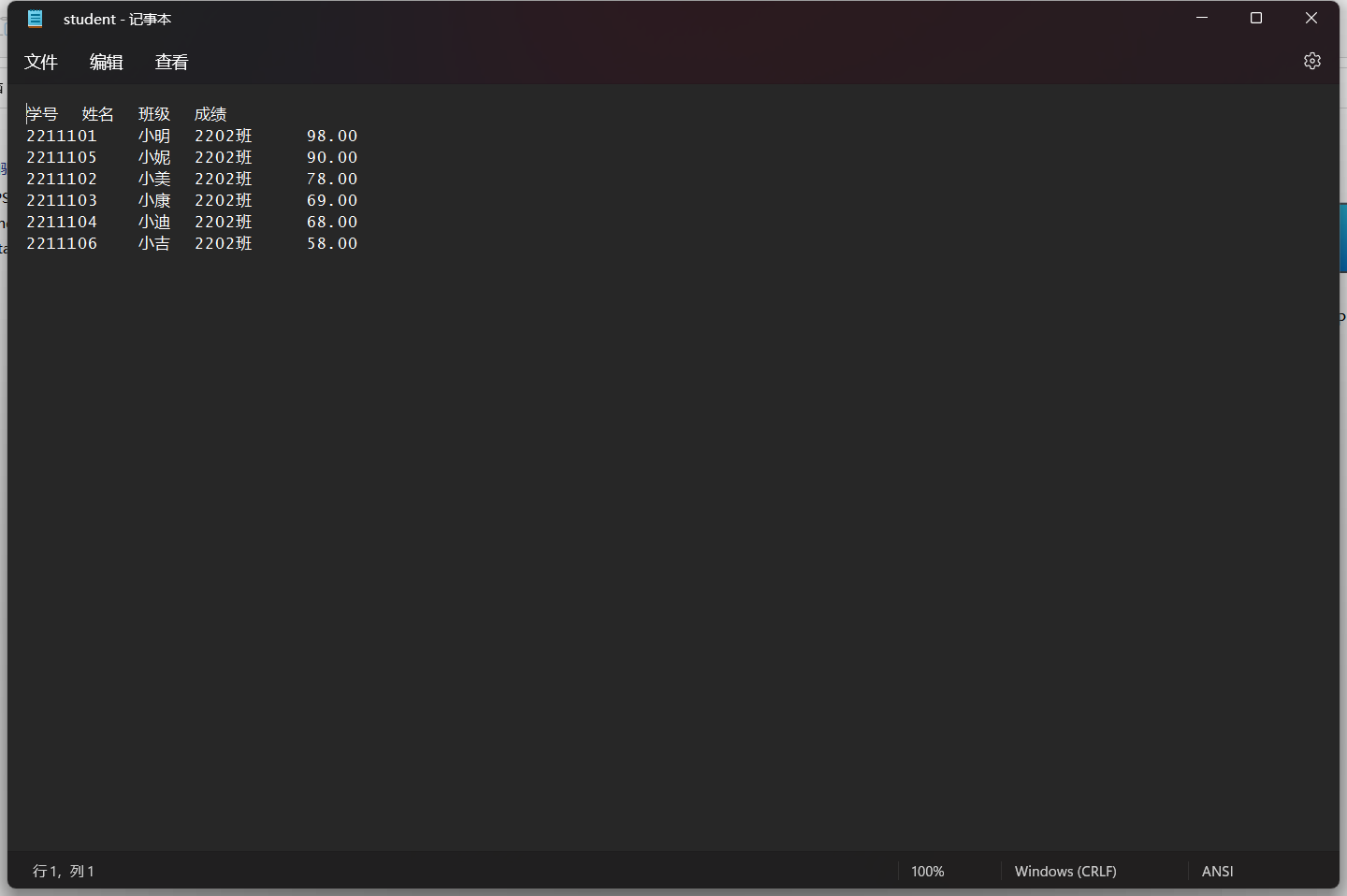


输入“0”退出系统，如下图：



打开“我的电脑”，并在搜索栏中输入“student.txt”，您就会看到您导出的报表了，如下图：





本系统的全部使用流程已展示完毕，感谢您的观看和使用！