学生管理系统项目说明

1. 项目来源和背景

随着时代发展，学生成为国家关注培养的重点，随着学生人数的增多，手工记录学生信息的方式已经不再是适用于当下，；利用信息化的学生信息管理系统更方便、快捷地统计学生的信息，记录学生的信息，对学生信息的变化及时更新，同样也可以使人们实时地了解学生成绩的动态，更好地管理学生，更准确地指引学生方向。

1. 项目设计即需求分析

(1)实现学生成绩信息的录人。

(2)可以按姓名、学号删除成绩信息。

(3)可以按姓名、学号进行查找。

(4)可以修改某个特定学号的学生信息

(5)可以增加一个学生的信息。

(6)对所有学生按总分从高到低进行排序。

(7)具有一定的统计功能，如输出学生总数、输出每门课程小于 60 分的人数、输出每门课程的最高分等。

(8)将所有操作之后的数据保存为磁盘文件，下次打开软件的时候能自动调人数据。

(9)对以上功能进行菜单式选择的方式进行操作。

框架图如下

学生信息管理系统

添加记录

显示记录

修改记录

删除记录

查找记录

统

计

排

序

退出系统

总结：先对预模块进行设计，接着分步编写每个函数，设计主函数实现函数功能，最终实现学生管理系统功能

代码说明

1. 预模块处理

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <dos.h>

#include <string.h>

// 宏定义部分

#define LEN sizeof(struct student) // 学生结构体长度

#define FORMAT "%-8d%-15s%-12.11f%-12.11f%-12.11f%-12.1lf" // 输出格式

#define DATA stu[i].num, stu[i].name, stu[i].elec, stu[i].expe, stu[i].requ, stu[i].sum

// 定义学生信息结构体

typedef struct student {

int num; // 学号

char name[15]; // 姓名

double elec; // 选修课成绩

double expe; // 实验课成绩

double requ; // 必修课成绩

double sum; // 总分

} stu;

stu students[50]; // 定义结构体数组

int count = 0; // 用于记录当前学生人数

stu imformation[100];//用于查找学生信息

说明：首先结合后续代码编写需求引用函数库，之后通过宏定义实现了自定义结构体类型的长度、输出的格式控制部分和结构体类型的数组引用成员的输出列表，接着定义学生结构体，用于存储学生的信息，包括学生的学号，姓名，选修课成绩，实验课成绩，必修课成绩，总分；之后预先定义学生结构体数组用于存储学生信息，并定义一个变量来记录学生的人数

2. 函数编写分步实现功能

//菜单显示函数

void menu() {

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

printf("| 学生信息管理系统 |\n");

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

printf("| 1. 录入学生信息 |\n");

printf("| 2. 查询学生信息 |\n");

printf("| 3. 删除学生信息 |\n");

printf("| 4. 修改学生信息 |\n");

printf("| 5. 插入学生信息 |\n");

printf("| 6. 排序 |\n");

printf("| 7. 统计学生总数 |\n");

printf("| 8. 显示所有学生信息 |\n");

printf("| 0. 退出系统 |\n");

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

}

说明：通过在系统界面显示菜单，让用户选择对应的功能

// 录入学生成绩信息函数

void in(stu students[], int\* count) {

int n;

printf("请输入要录入的学生人数: ");

scanf("%d", &n);

//录入学生信息

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

(\*count)++;

printf("录入第 %d 个学生的信息:\n", (\*count));

printf("学号: ");

scanf("%d", &students[(\*count) - 1].num);

printf("姓名: ");

scanf("%s", students[(\*count) - 1].name);

printf("选修课成绩: ");

scanf("%lf", &students[(\*count) - 1].elec);

printf("实验课成绩: ");

scanf("%lf", &students[(\*count) - 1].expe);

printf("必修课成绩: ");

scanf("%lf", &students[(\*count) - 1].requ);

// 计算总分

students[(\*count) - 1].sum = students[(\*count) - 1].elec + students[(\*count) - 1].expe + students[(\*count) - 1].requ;

//printf("总分: %.2f\n", students[\*count].sum);

printf("%d,%s,%lf,%lf,%lf,%lf\n", students[(\*count) - 1].num,

students[(\*count) - 1].name,

students[(\*count) - 1].elec,

students[(\*count) - 1].expe,

students[(\*count) - 1].requ,

students[(\*count) - 1].sum);

//第一个元素的索引是0，所以(\*count) - 1（例子：第一次循环count变成1）

FILE\* file = fopen("school.csv", "a");

if (file == NULL) {

perror("no file");

exit(EXIT\_FAILURE);

}

fprintf(file, "%d,%s,%lf,%lf,%lf,%lf\n",

students[(\*count) - 1].num,

students[(\*count) - 1].name,

students[(\*count) - 1].elec,

students[(\*count) - 1].expe,

students[(\*count) - 1].requ,

students[(\*count) - 1].sum);

fclose(file);

printf("录入成功。\n");

}

}

说明：首先提示用户需要录入的学生人数，之后结合结构体数组分别对学生的每个信息进行录入，需要注意的是，这里用了一个（\*count）对学生人数进行统计，这样可以确保学生的人数是动态变化的，而不是每次打开系统学生的人数会清零需要重新新录入，同时每次循环（\*count）的值自增一次，在打印对应信息的时候下标索引值也要发生变化，最后通过文件操作录入并保存学生信息

//查找学生信息函数

void search()

{

FILE\* file = fopen("school.csv", "r");

if (file == NULL) {

perror("no file");

exit(EXIT\_FAILURE);

}

printf("文件打开成功\n");

int numStu = 0;

while (fscanf(file, "%d,%[^,],%lf,%lf,%lf,%lf\n",

&imformation[numStu].num,

imformation[numStu].name,

&imformation[numStu].elec,

&imformation[numStu].expe,

&imformation[numStu].requ,

&imformation[numStu].sum) == 6)

{

numStu++;

}

fclose(file);

int a;

printf("输入需要查询的学号\n");

scanf("%d", &a);

int i;

for (i = 0; i <= 100; i++)

{

if (imformation[i].num != a)

{

continue;

}

if (imformation[i].num == a)

{

printf("学号：%d，姓名：%s，选修课成绩：%.2lf，实验课成绩：%.2lf，必修课成绩：%.2lf，总分：%.2lf\n",

imformation[i].num, imformation[i].name, imformation[i].elec, imformation[i].expe, imformation[i].requ, imformation[i].sum);

return;

}

}

printf("未查询到该生信息\n");//如果到100都没有找到，循环结束，显示没有该生信息

说明：通过文件操作打开已保存的学生信息，录入到本地之后存储在一个新的数组里面，之后哦通过循环遍历来实现学生信息查找功能，需要注意的是这样在查找之前需要做判断学生的信息是否存在（即有在文件中无和需要查询学生的信息匹配）

//删除学生信息函数

void del()

{

// 首先以只读模式打开文件，读取学生信息

FILE\* file = fopen("school.csv", "r");

if (file == NULL) {

perror("无法打开文件");

return;

}

int numStu = 0;

// 从文件中读取学生信息并存储到 imformation 数组中

while (fscanf(file, "%d,%[^,],%lf,%lf,%lf,%lf\n",

&imformation[numStu].num,

imformation[numStu].name,

&imformation[numStu].elec,

&imformation[numStu].expe,

&imformation[numStu].requ,

&imformation[numStu].sum) == 6) {

numStu++;

}

fclose(file);

int del\_num;

// 提示用户输入要删除的学生学号

printf("请输入要删除学生的学号：");

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

// 标记是否找到要删除的学生

int found = 0;

// 以读写模式打开文件

file = fopen("school.csv", "w");//"w"只写模式。如果文件不存在，创建文件；如果文件存在，清空文件

if (file == NULL) {

perror("无法打开文件进行写入");

return;

}

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

// 如果不是要删除的学生，将其信息重新写入文件

if (imformation[i].num != del\_num) {

fprintf(file, "%d,%s,%.2lf,%.2lf,%.2lf,%.2lf\n",

imformation[i].num,

imformation[i].name,

imformation[i].elec,

imformation[i].expe,

imformation[i].requ,

imformation[i].sum);

}

else {

found = 1;

}

}

fclose(file);

if (found) {

printf("学号为 %d 的学生信息已删除。\n", del\_num);

}

else {

printf("未找到学号为 %d 的学生信息。\n", del\_num);

}

}

说明：

**文件打开与读取**：

首先以只读模式打开文件 D:\\ye\\桌面\\cpp\\school.csv，使用 fopen 函数。

若文件打开失败，通过 perror 输出错误信息并返回。

定义 int numStu = 0 用于统计学生数量，使用 fscanf 从文件中读取学生信息（学号、姓名、选修课成绩、实验课成绩、必修课成绩、总分）存储到 imformation 数组中，每成功读取一条记录，numStu 加 1。

读取结束后关闭文件。

**用户输入：**

提示用户输入要修改的学生学号，并存储在 modify\_num 中。

定义 int found = 0 用于标记是否找到要修改的学生。

**文件修改操作：**

再次以读写模式打开文件 D:\\ye\\桌面\\cpp\\school.csv，若打开失败，输出错误信息并返回。

遍历 imformation 数组，找到与输入学号匹配的学生信息。

若找到，输出当前学生信息，提示用户输入新的信息（学号、姓名、各科成绩），重新计算总分。

输出修改后的学生信息。

将文件指针定位到修改位置，遍历 imformation 数组，将修改后的信息重新写入文件，对于匹配的学生记录，将其新信息写入，对于其他记录，将原信息重新写入。

若找到修改学生，将 found 标记为 1 并退出查找循环。

**结果输出：**

若未找到要修改的学生学号，输出相应信息。

最后关闭文件。

//修改学生信息函数

void modify()

{

// 首先以只读模式打开文件，读取学生信息

FILE\* file = fopen("school.csv", "r");

if (file == NULL) {

perror("无法打开文件");

return;

}

int numStu = 0;

// 从文件中读取学生信息并存储到 imformation 数组中

while (fscanf(file, "%d,%[^,],%lf,%lf,%lf,%lf\n",

&imformation[numStu].num,

imformation[numStu].name,

&imformation[numStu].elec,

&imformation[numStu].expe,

&imformation[numStu].requ,

&imformation[numStu].sum) == 6) {

numStu++;

}

fclose(file);

int modify\_num;

// 提示用户输入要修改的学生学号

printf("请输入要修改的学生的学号：");

scanf("%d", &modify\_num);

// 标记是否找到要修改的学生

int found = 0;

// 以读写模式打开文件

file = fopen("school.csv", "r+");

if (file == NULL) {

perror("无法打开文件进行写入");

return;

}

for (int j = 0; j < numStu; j++) {

if (imformation[j].num == modify\_num) {

// 输出当前学生信息

printf("目前该学生信息：\n");

printf("学号：%d，姓名：%s，选修课成绩：%.2lf，实验课成绩：%.2lf，必修课成绩：%.2lf，总分：%.2lf\n",

imformation[j].num, imformation[j].name, imformation[j].elec, imformation[j].expe, imformation[j].requ, imformation[j].sum);

// 提示用户输入新的信息

printf("修改学号为：");

scanf("%d", &imformation[j].num);

printf("修改姓名为：");

scanf("%s", imformation[j].name);

printf("修改选修课成绩为：");

scanf("%lf", &imformation[j].elec);

printf("修改实验课成绩为：");

scanf("%lf", &imformation[j].expe);

printf("修改必修课成绩为：");

scanf("%lf", &imformation[j].requ);

// 重新计算总分

imformation[j].sum = imformation[j].elec + imformation[j].expe + imformation[j].requ;

printf("修改后该学生信息：\n");

printf("学号：%d，姓名：%s，选修课成绩：%.2lf，实验课成绩：%.2lf，必修课成绩：%.2lf，总分：%.2lf\n",

imformation[j].num, imformation[j].name, imformation[j].elec, imformation[j].expe, imformation[j].requ, imformation[j].sum);

// 将文件指针定位到修改的位置

//fseek(file, 0, SEEK\_SET);

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

if (i == j) {

fprintf(file, "%d,%s,%.2lf,%.2lf,%.2lf,%.2lf\n",

imformation[i].num,

imformation[i].name,

imformation[i].elec,

imformation[i].expe,

imformation[i].requ,

imformation[i].sum);

}

else {

fprintf(file, "%d,%s,%.2lf,%.2lf,%.2lf,%.2lf\n",

imformation[i].num,

imformation[i].name,

imformation[i].elec,

imformation[i].expe,

imformation[i].requ,

imformation[i].sum);

}

}

found = 1;

break;

}

}

if (!found) {

printf("未找到要修改的学生学号。\n");

}

fclose(file);

}

说明：

**文件打开与读取**：

首先以只读模式打开文件 D:\\ye\\桌面\\cpp\\school.csv，使用 fopen 函数。

若文件打开失败，通过 perror 输出错误信息并返回。

定义 int numStu = 0 用于统计学生数量，使用 fscanf 从文件中读取学生信息（学号、姓名、选修课成绩、实验课成绩、必修课成绩、总分）存储到 imformation 数组中，每成功读取一条记录，numStu 加 1。

读取结束后关闭文件。

**用户输入：**

提示用户输入要修改的学生学号，并存储在 modify\_num 中。

定义 int found = 0 用于标记是否找到要修改的学生。

**文件修改操作：**

再次以读写模式打开文件 D:\\ye\\桌面\\cpp\\school.csv，若打开失败，输出错误信息并返回。

遍历 imformation 数组，找到与输入学号匹配的学生信息。

若找到，输出当前学生信息，提示用户输入新的信息（学号、姓名、各科成绩），重新计算总分。

输出修改后的学生信息。

将文件指针定位到修改位置，遍历 imformation 数组，将修改后的信息重新写入文件，对于匹配的学生记录，将其新信息写入，对于其他记录，将原信息重新写入。

若找到修改学生，将 found 标记为 1 并退出查找循环。

**结果输出：**

若未找到要修改的学生学号，输出相应信息。

最后关闭文件。

//插入学生信息函数

void insert()

{

// 首先以只读模式打开文件，读取学生信息

FILE\* file = fopen("school.csv", "r");

if (file == NULL) {

perror("无法打开文件");

return;

}

int numStu = 0;

// 从文件中读取学生信息并存储到 imformation 数组中

while (fscanf(file, "%d,%[^,],%lf,%lf,%lf,%lf\n",

&imformation[numStu].num,

imformation[numStu].name,

&imformation[numStu].elec,

&imformation[numStu].expe,

&imformation[numStu].requ,

&imformation[numStu].sum) == 6) {

numStu++;

}

fclose(file);

int insert\_num;

printf("请输入要插入的位置（从 0 开始）：");

scanf("%d", &insert\_num);

// 范围检查

if (insert\_num < 0 || insert\_num > numStu) {

printf("插入位置无效。\n");

return;

}

// 将插入位置之后的元素向后移动

for (int i = numStu; i >= insert\_num; i--) {

imformation[i + 1] = imformation[i];

}

// 提示用户输入新的信息

printf("请输入新的信息：\n");

printf("学号为：");

scanf("%d", &imformation[insert\_num].num);

printf("姓名为：");

scanf("%s", imformation[insert\_num].name);

printf("选修课成绩为：");

scanf("%lf", &imformation[insert\_num].elec);

printf("实验课成绩为：");

scanf("%lf", &imformation[insert\_num].expe);

printf("必修课成绩为：");

scanf("%lf", &imformation[insert\_num].requ);

// 计算总分

imformation[insert\_num].sum = imformation[insert\_num].elec + imformation[insert\_num].expe + imformation[insert\_num].requ;

// 以读写模式打开文件

file = fopen("school.csv", "w+");

if (file == NULL) {

perror("无法打开文件进行写入");

return;

}

// 将所有学生信息写入文件

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

fprintf(file, "%d,%s,%.2lf,%.2lf,%.2lf,%.2lf\n",

imformation[i].num,

imformation[i].name,

imformation[i].elec,

imformation[i].expe,

imformation[i].requ,

imformation[i].sum);

}

fclose(file);

printf("插入成功。\n");

}

说明：

**文件打开与读取**：

首先以只读模式打开文件 D:\\ye\\桌面\\cpp\\school.csv，使用 fopen 函数。

若文件打开失败，通过 perror 输出错误信息并返回。

定义 int numStu = 0 用于统计学生数量，使用 fscanf 从文件中读取学生信息（学号、姓名、选修课成绩、实验课成绩、必修课成绩、总分）存储到 imformation 数组中，每成功读取一条记录，numStu 加 1。

读取结束后关闭文件。

**用户输入**：

提示用户输入要插入信息的位置（从 0 开始），存储在 insert\_num 中。

进行范围检查，若插入位置无效（小于 0 或大于 numStu），输出错误信息并返回。

**元素移动与信息插入**：

将插入位置之后的元素依次向后移动一位，为新元素腾出空间。

提示用户输入新的学生信息（学号、姓名、各科成绩），计算新学生的总分。

**文件更新**：

以读写模式打开文件，若打开失败，输出错误信息并返回。

将更新后的 imformation 数组中的学生信息全部写入文件，使用 fprintf 函数。

关闭文件。

**结果输出**：

输出插入成功信息。

//交换函数

void swap(stu\* a, stu\* b)

{

stu temp = \*a;

\*a = \*b;

\*b = temp;

}

//学生成绩排序函数

void order()

{

// 首先以只读模式打开文件，读取学生信息

FILE\* file = fopen("school.csv", "r");

if (file == NULL) {

perror("无法打开文件");

return;

}

int numStu = 0;

// 从文件中读取学生信息并存储到 imformation 数组中

while (fscanf(file, "%d,%[^,],%lf,%lf,%lf,%lf\n",

&imformation[numStu].num,

imformation[numStu].name,

&imformation[numStu].elec,

&imformation[numStu].expe,

&imformation[numStu].requ,

&imformation[numStu].sum) == 6) {

numStu++;

}

fclose(file);

for (int i = 0; i < numStu; i++)

{

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

{

if (imformation[j].sum < imformation[j + 1].sum)

{

swap(&imformation[j], &imformation[j + 1]);

}

}

}

for (int i = 0; i < numStu; i++) //打印数组内数据

{

printf("学号：%d，姓名：%s，选修课成绩：%.2lf，实验课成绩：%.2lf，必修课成绩：%.2lf，总分：%.2lf\n",

imformation[i].num, imformation[i].name, imformation[i].elec, imformation[i].expe, imformation[i].requ, imformation[i].sum);

}

}

说明：

**文件打开与读取**：

首先以只读模式打开文件 D:\\ye\\桌面\\cpp\\school.csv，使用 fopen 函数。

若文件打开失败，通过 perror 输出错误信息并返回。

定义 int numStu = 0 用于统计学生数量，使用 fscanf 从文件中读取学生信息（学号、姓名、选修课成绩、实验课成绩、必修课成绩、总分）存储到 imformation 数组中，每成功读取一条记录，numStu 加 1。

读取结束后关闭文件。

**学生成绩排序**：

调用 swap 函数对学生信息进行排序。

采用冒泡排序算法，通过两层 for 循环，比较相邻学生的总分（imformation[j].sum 和 imformation[j + 1].sum），如果前一个学生的总分小于后一个学生的总分，调用 swap 函数交换它们的位置。

**结果输出**：

最后使用 printf 打印排序后的学生信息，包括学号、姓名、各科目成绩和总分。

//统计学生总数函数

void total()

{

int n=0;//局部定义一个变量记录学生的总人数

FILE\* file = fopen("school.csv", "r");

if (file == NULL) {

perror("no file");

exit(EXIT\_FAILURE);

}

printf("文件打开成功\n");

int numStu = 0;

while (fscanf(file, "%d,%[^,],%lf,%lf,%lf,%lf\n",

&imformation[numStu].num,

imformation[numStu].name,

&imformation[numStu].elec,

&imformation[numStu].expe,

&imformation[numStu].requ,

&imformation[numStu].sum) == 6)

{

numStu++;

}

fclose(file);

for (int i = 0; i < 100; i++)

{

//如果数组没有存储信息，不应该算一个学生，累加之前先做判断

if (imformation[i].num != NULL)

{

n++;

}

}

printf("当前学生总数为：%d\n", n);

}

说明：利用文件操作，导入本地文件并读取，之后利用循环遍历，定义一个变量n来统计学生的总数

//显示所有学生信息函数

void show()

{

FILE\* file = fopen("school.csv", "r");

if (file == NULL) {

perror("no file");

exit(EXIT\_FAILURE);

}

printf("文件打开成功\n");

int numStu = 0;

while (fscanf(file, "%d,%[^,],%lf,%lf,%lf,%lf\n",

&imformation[numStu].num,

imformation[numStu].name,

&imformation[numStu].elec,

&imformation[numStu].expe,

&imformation[numStu].requ,

&imformation[numStu].sum) == 6)

{

numStu++;

}

fclose(file);

printf("从文件中读取到的学生信息如下：\n");

for (int i = 0; i < numStu; i++)

{

printf("学号：%d，姓名：%s，选修课成绩：%.2lf，实验课成绩：%.2lf，必修课成绩：%.2lf，总分：%.2lf\n",

imformation[i].num, imformation[i].name, imformation[i].elec, imformation[i].expe, imformation[i].requ, imformation[i].sum);

}

说明：通过文件操作导入本地存储学生信息的文件，之后利用循环遍历依次打印每个结构体数组的内容，实现显示所有学生信息

3.主函数部分

// 主函数实现

int main() {

int choice;

// 显示主菜单

menu();

// 选择功能并执行

while (1) {

printf("请选择<0-8>：");

// 规避输入字母进入死循环

if (scanf("%d", &choice) != 1) { // 检查输入是否为整数

printf("输入无效，请输入数字！\n");

while (getchar() != '\n'); // 清空输入缓冲区

continue; // 重新进入循环

}

switch (choice) {

case 1: in(students, &count); break;

case 2: search(); break;

case 3: del(); break;

case 4: modify(); break;

case 5: insert(); break;

case 6: order(); break;

case 7: total(); break;

case 8: show(); break;

case 0: return 0; break;

default: printf("无效选择，请重新输入\n"); break;

}

}

return 0;

}

说明，利用switch-case结构实现用户选择功能，并在对应的功能选择中调用对应功能的函数，需要注意的是用户功能选择部分的代码设计，如果输入了不相关字符，需要对不相关字符进行处理之后继续让用户输入，如果不处理将出现死循环