实验题目：学生综合测评系统

实验目的：设计和实现一个学生综合测评系统，能够录入学生信息，计算综合测评总分，并能根据不同的功能要求进行学生信息的管理和查询。

实验环境：使用C语言进行编程

实验方法：

1. 录入学生信息：

实验目的：能够录入学生的基本信息，并将其存储到文件中。

实验步骤：

- 定义一个结构体`Student`来表示学生信息，包括学号、姓名、性别、联系电话等。

- 声明一个数组`students`来保存学生信息，设置一个变量`count`来记录学生数量。

- 编写函数`add\_or\_modify\_student`，用于录入或修改学生信息。在函数中，要求用户输入学生信息，并将其存储到数组`students`中。

- 将录入的学生信息存储到文件中，以便下次程序运行时可以加载。

// 录入学生信息函数

void add\_or\_modify\_student(struct Student students[], int \*count) {

printf("请输入学号：");

char student\_id[20];

scanf("%s", student\_id);

for (int i = 0; i < \*count; i++) {

if (strcmp(students[i].student\_id, student\_id) == 0) {

printf("已找到学号为 %s 的学生信息，进行修改。\n", student\_id);

printf("请输入姓名：");

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

printf("请输入性别：");

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

printf("请输入联系电话：");

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

printf("请输入C语言成绩：");

scanf("%f", &students[i].c\_language);

printf("请输入高等数学成绩：");

scanf("%f", &students[i].higher\_math);

printf("请输入外语成绩：");

scanf("%f", &students[i].foreign\_language);

save\_student\_data(students, \*count);

printf("学生信息修改成功。\n");

return;

}

}

// 如果没有找到该学号的学生，则添加新的学生信息

if (\*count >= MAX\_STUDENTS) {

printf("学生人数已达上限，无法添加新的学生。\n");

return;

}

strcpy(students[\*count].student\_id, student\_id);

printf("请输入姓名：");

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

printf("请输入性别：");

scanf("%s", students[\*count].gender);

printf("请输入联系电话：");

scanf("%s", students[\*count].phone);

printf("请输入C语言成绩：");

scanf("%f", &students[\*count].c\_language);

printf("请输入高等数学成绩：");

scanf("%f", &students[\*count].higher\_math);

printf("请输入外语成绩：");

scanf("%f", &students[\*count].foreign\_language);

(\*count)++;

save\_student\_data(students, \*count);

printf("学生信息添加成功。\n");

} ```

2. 修改学生信息：

实验目的：能够根据学号修改学生的信息。

实验步骤：

- 编写函数`add\_or\_modify\_student`，在录入学生信息时，先要求用户输入要修改的学号。

- 判断该学号是否存在于数组`students`中，如果存在，则要求用户输入修改后的信息，并更新数组中对应学生的信息。

- 如果学号不存在，则将新的学生信息添加到数组中。

代码同上

3. 删除学生信息：

实验目的：能够根据学号删除学生的信息。

实验步骤：

- 编写函数`delete\_student`，要求用户输入要删除的学号。

- 在数组`students`中查找该学号，如果存在，则输出学生信息并确认是否删除。

- 如果确认删除，则从数组中删除该学生信息，并更新学生数量。

// 删除学生信息函数

void delete\_student(struct Student students[], int \*count) {

printf("请输入要删除的学生学号：");

char student\_id[20];

scanf("%s", student\_id);

int found = 0;

for (int i = 0; i < \*count; i++) {

if (strcmp(students[i].student\_id, student\_id) == 0) {

printf("找到学号为 %s 的学生信息：\n", student\_id);

printf("姓名：%s\n", students[i].name);

printf("性别：%s\n", students[i].gender);

printf("联系电话：%s\n", students[i].phone);

printf("C语言成绩：%.2f\n", students[i].c\_language);

printf("高等数学成绩：%.2f\n", students[i].higher\_math);

printf("外语成绩：%.2f\n", students[i].foreign\_language);

printf("同学互评分：%.2f\n", students[i].peer\_evaluation);

printf("品德成绩：%.2f\n", students[i].character\_score);

printf("任课教师评分：%.2f\n", students[i].teacher\_evaluation);

printf("综合测评总分：%.2f\n", students[i].total\_score);

printf("综合测评名次：%d\n", students[i].total\_rank);

printf("确认是否删除该学生信息（Y/N）：");

char confirm[2];

scanf("%s", confirm);

if (strcmp(confirm, "Y") == 0 || strcmp(confirm, "y") == 0) {

for (int j = i; j < \*count - 1; j++) {

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

}

(\*count)--;

save\_student\_data(students, \*count);

printf("学生信息删除成功。\n");

} else {

printf("取消删除学生信息。\n");

}

found = 1;

break;

}

}

if (!found) {

printf("未找到学号为 %s 的学生信息。\n", student\_id);

}

}

4. 浏览学生信息：

实验目的：能够显示所有学生的信息。

实验步骤：

- 编写函数`browse\_student\_info`，遍历数组`students`，输出每个学生的信息。

// 浏览学生信息函数

void browse\_student\_info(struct Student students[], int count) {

printf("浏览学生信息：\n");

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

printf("学号：%s，姓名：%s，性别：%s，联系电话：%s，C语言成绩：%.2f，高等数学成绩：%.2f，外语成绩：%.2f\n",students[i].student\_id,students[i].name,students[i].gender,students[i].phone,students[i].c\_language, students[i].higher\_math, students[i].foreign\_language);

}

}

5. 录入同学互评分：

实验目的：能够录入学生的同学互评分。

实验步骤：

- 在结构体`Student`中添加同学互评分字段。

- 编写函数`enter\_peer\_evaluation`，要求用户输入每个学生的同学互评分。

// 录入同学互评分函数

oid enter\_peer\_evaluation(struct Student students[], int count) {

printf("请输入同学互评分：\n");

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

printf("请输入学号为 %s 的学生同学互评分：", students[i].student\_id);

scanf("%f", &students[i].peer\_evaluation);

}

save\_student\_data(students, count);

printf("同学互评分录入成功。\n");

}

6. 录入品德成绩：

实验目的：能够录入学生的品德成绩。

实验步骤：

- 在结构体`Student`中添加品德成绩字段。

- 编写函数`enter\_character\_scores`，要求用户输入每个学生的品德成绩。

// 录入品德成绩函数

void enter\_character\_scores(struct Student students[], int count) {

printf("请输入品德成绩：\n");

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

printf("请输入学号为 %s 的学生品德成绩：", students[i].student\_id);

scanf("%f", &students[i].character\_score);

}

save\_student\_data(students, count);

printf("品德成绩录入成功。\n");

}

7. 录入任课教师评分：

实验目的：能够录入学生的任课教师评分。

实验步骤：

- 在结构体`Student`中添加任课教师评分字段。

- 编写函数`enter\_teacher\_evaluation`，要求用户输入每个学生的任课教师评分。

// 录入任课教师评分函数

void enter\_teacher\_evaluation(struct Student students[], int count) {

printf("请输入任课教师评分：\n");

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

printf("请输入学号为 %s 的学生任课教师评分：", students[i].student\_id);

scanf("%f", &students[i].teacher\_evaluation);

}

save\_student\_data(students, count);

printf("任课教师评分录入成功。\n");

}

8. 计算综合测评总分：

实验目的：能够根据各项成绩计算学生的综合测评总分。

实验步骤：

- 在结构体`Student`中添加综合测评总分字段。

- 编写函数`calculate\_total\_scores`，根据规定的权重系数计算学生的综合测评总分。

// 计算综合测评总分函数

void calculate\_total\_scores(struct Student students[], int count) {

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

students[i].total\_score =

(students[i].c\_language + students[i].higher\_math + students[i].foreign\_language) / 3 \* 0.6 +

students[i].peer\_evaluation \* 0.1 +

students[i].character\_score \* 0.1 +

students[i].teacher\_evaluation \* 0.2;

}

save\_student\_data(students, count);

printf("综合测评总分计算完成。\n");

}

12. 按照综合测评成绩名次输出学生信息：

实验目的：能够按照综合测评成绩的名次对学生信息进行排序并输出。

实验步骤：

- 编写函数`sort\_by\_total\_scores`，根据学生的综合测评总分进行排序。

- 遍历排序后的数组`students`，输出学生的信息。

// 按照综合测评成绩名次输出学生信息函数

void print\_student\_info\_by\_total\_rank(struct Student students[], int count) {

// 按照综合测评总分排序

for (int i = 0; i < count - 1; i++) {

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

if (students[j].total\_score < students[j + 1].total\_score) {

struct Student temp = students[j];

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

students[j + 1] = temp;

}

}

}

printf("按照综合测评名次输出学生信息：\n");

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

printf("学号：%s，姓名：%s，综合测评总分：%.2f，综合测评名次：%d\n", students[i].student\_id,

students[i].name, students[i].total\_score, i + 1);

}

}

13. 按照单科成绩排名输出学生信息：

实验目的：能够按照单科成绩的排名对学生信息进行排序并输出。

实验步骤：

- 编写函数`sort\_by\_subject\_scores`，根据学生的单科成绩进行排序。

- 遍历排序后的数组`students`，输出学生的信息。

// 按照单科成绩排名输出学生信息函数

void print\_student\_info\_by\_subject\_score(struct Student students[], int count) {

// 按照C语言成绩排序

for (int i = 0; i < count - 1; i++) {

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

if (students[j].c\_language < students[j + 1].c\_language) {

struct Student temp = students[j];

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

students[j + 1] = temp;

}

}

}

printf("按照C语言成绩排名输出学生信息：\n");

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

printf("学号：%s，姓名：%s，C语言成绩：%.2f\n", students[i].student\_id, students[i].name,

students[i].c\_language);

}

}

总代码：

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_STUDENTS 100

struct Student {

char student\_id[20];

char name[20];

char gender[10];

char phone[20];

float c\_language;

float higher\_math;

float foreign\_language;

float peer\_evaluation;

float character\_score;

float teacher\_evaluation;

float total\_score;

int total\_rank;

};

void save\_student\_data(struct Student students[], int count) {

FILE \*file = fopen("student\_data.txt", "w");

if (file == NULL) {

printf("无法打开文件。\n");

exit(1);

}

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

fprintf(file, "%s %s %s %s %.2f %.2f %.2f %.2f %.2f %.2f %.2f %d\n", students[i].student\_id,

students[i].name, students[i].gender, students[i].phone, students[i].c\_language,

students[i].higher\_math, students[i].foreign\_language, students[i].peer\_evaluation,

students[i].character\_score, students[i].teacher\_evaluation, students[i].total\_score,

students[i].total\_rank);

}

fclose(file);

}

void load\_student\_data(struct Student students[], int \*count) {

FILE \*file = fopen("student\_data.txt", "r");

if (file == NULL) {

printf("无法打开文件。\n");

exit(1);

}

\*count = 0;

while (fscanf(file, "%s %s %s %s %f %f %f %f %f %f %f %d\n", students[\*count].student\_id,

students[\*count].name, students[\*count].gender, students[\*count].phone,

&students[\*count].c\_language, &students[\*count].higher\_math,

&students[\*count].foreign\_language, &students[\*count].peer\_evaluation,

&students[\*count].character\_score, &students[\*count].teacher\_evaluation,

&students[\*count].total\_score, &students[\*count].total\_rank) != EOF) {

(\*count)++;

}

fclose(file);

}

void add\_or\_modify\_student(struct Student students[], int \*count) {

printf("请输入学号：");

char student\_id[20];

scanf("%s", student\_id);

for (int i = 0; i < \*count; i++) {

if (strcmp(students[i].student\_id, student\_id) == 0) {

printf("已找到学号为 %s 的学生信息，进行修改。\n", student\_id);

printf("请输入姓名：");

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

printf("请输入性别：");

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

printf("请输入联系电话：");

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

printf("请输入C语言成绩：");

scanf("%f", &students[i].c\_language);

printf("请输入高等数学成绩：");

scanf("%f", &students[i].higher\_math);

printf("请输入外语成绩：");

scanf("%f", &students[i].foreign\_language);

save\_student\_data(students, \*count);

printf("学生信息修改成功。\n");

return;

}

}

// 如果没有找到该学号的学生，则添加新的学生信息

if (\*count >= MAX\_STUDENTS) {

printf("学生人数已达上限，无法添加新的学生。\n");

return;

}

strcpy(students[\*count].student\_id, student\_id);

printf("请输入姓名：");

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

printf("请输入性别：");

scanf("%s", students[\*count].gender);

printf("请输入联系电话：");

scanf("%s", students[\*count].phone);

printf("请输入C语言成绩：");

scanf("%f", &students[\*count].c\_language);

printf("请输入高等数学成绩：");

scanf("%f", &students[\*count].higher\_math);

printf("请输入外语成绩：");

scanf("%f", &students[\*count].foreign\_language);

(\*count)++;

save\_student\_data(students, \*count);

printf("学生信息添加成功。\n");

}

void delete\_student(struct Student students[], int \*count) {

printf("请输入要删除的学生学号：");

char student\_id[20];

scanf("%s", student\_id);

int found = 0;

for (int i = 0; i < \*count; i++) {

if (strcmp(students[i].student\_id, student\_id) == 0) {

printf("找到学号为 %s 的学生信息：\n", student\_id);

printf("姓名：%s\n", students[i].name);

printf("性别：%s\n", students[i].gender);

printf("联系电话：%s\n", students[i].phone);

printf("C语言成绩：%.2f\n", students[i].c\_language);

printf("高等数学成绩：%.2f\n", students[i].higher\_math);

printf("外语成绩：%.2f\n", students[i].foreign\_language);

printf("同学互评分：%.2f\n", students[i].peer\_evaluation);

printf("品德成绩：%.2f\n", students[i].character\_score);

printf("任课教师评分：%.2f\n", students[i].teacher\_evaluation);

printf("综合测评总分：%.2f\n", students[i].total\_score);

printf("综合测评名次：%d\n", students[i].total\_rank);

printf("确认是否删除该学生信息（Y/N）：");

char confirm[2];

scanf("%s", confirm);

if (strcmp(confirm, "Y") == 0 || strcmp(confirm, "y") == 0) {

for (int j = i; j < \*count - 1; j++) {

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

}

(\*count)--;

save\_student\_data(students, \*count);

printf("学生信息删除成功。\n");

} else {

printf("取消删除学生信息。\n");

}

found = 1;

break;

}

}

if (!found) {

printf("未找到学号为 %s 的学生信息。\n", student\_id);

}

}

void browse\_student\_info(struct Student students[], int count) {

printf("浏览学生信息：\n");

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

printf("学号：%s，姓名：%s，性别：%s，联系电话：%s，C语言成绩：%.2f，高等数学成绩：%.2f，外语成绩：%.2f\n",

students[i].student\_id, students[i].name, students[i].gender, students[i].phone,

students[i].c\_language, students[i].higher\_math, students[i].foreign\_language);

}

}

void enter\_peer\_evaluation(struct Student students[], int count) {

printf("请输入同学互评分：\n");

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

printf("请输入学号为 %s 的学生同学互评分：", students[i].student\_id);

scanf("%f", &students[i].peer\_evaluation);

}

save\_student\_data(students, count);

printf("同学互评分录入成功。\n");

}

void enter\_character\_scores(struct Student students[], int count) {

printf("请输入品德成绩：\n");

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

printf("请输入学号为 %s 的学生品德成绩：", students[i].student\_id);

scanf("%f", &students[i].character\_score);

}

save\_student\_data(students, count);

printf("品德成绩录入成功。\n");

}

void enter\_teacher\_evaluation(struct Student students[], int count) {

printf("请输入任课教师评分：\n");

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

printf("请输入学号为 %s 的学生任课教师评分：", students[i].student\_id);

scanf("%f", &students[i].teacher\_evaluation);

}

save\_student\_data(students, count);

printf("任课教师评分录入成功。\n");

}

void calculate\_total\_scores(struct Student students[], int count) {

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

students[i].total\_score =

(students[i].c\_language + students[i].higher\_math + students[i].foreign\_language) / 3 \* 0.6 +

students[i].peer\_evaluation \* 0.1 +

students[i].character\_score \* 0.1 +

students[i].teacher\_evaluation \* 0.2;

}

save\_student\_data(students, count);

printf("综合测评总分计算完成。\n");

}

void print\_student\_info\_by\_total\_rank(struct Student students[], int count) {

// 按照综合测评总分排序

for (int i = 0; i < count - 1; i++) {

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

if (students[j].total\_score < students[j + 1].total\_score) {

struct Student temp = students[j];

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

students[j + 1] = temp;

}

}

}

printf("按照综合测评名次输出学生信息：\n");

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

printf("学号：%s，姓名：%s，综合测评总分：%.2f，综合测评名次：%d\n", students[i].student\_id,

students[i].name, students[i].total\_score, i + 1);

}

}

void print\_student\_info\_by\_subject\_score(struct Student students[], int count) {

// 按照C语言成绩排序

for (int i = 0; i < count - 1; i++) {

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

if (students[j].c\_language < students[j + 1].c\_language) {

struct Student temp = students[j];

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

students[j + 1] = temp;

}

}

}

printf("按照C语言成绩排名输出学生信息：\n");

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

printf("学号：%s，姓名：%s，C语言成绩：%.2f\n", students[i].student\_id, students[i].name,

students[i].c\_language);

}

}

void print\_student\_info(struct Student students[], int count) {

printf("学生综合信息输出：\n");

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

printf("学号：%s\n", students[i].student\_id);

printf("姓名：%s\n", students[i].name);

printf("性别：%s\n", students[i].gender);

printf("联系电话：%s\n", students[i].phone);

printf("C语言成绩：%.2f\n", students[i].c\_language);

printf("高等数学成绩：%.2f\n", students[i].higher\_math);

printf("外语成绩：%.2f\n", students[i].foreign\_language);

printf("同学互评分：%.2f\n", students[i].peer\_evaluation);

printf("品德成绩：%.2f\n", students[i].character\_score);

printf("任课教师评分：%.2f\n", students[i].teacher\_evaluation);

printf("综合测评总分：%.2f\n", students[i].total\_score);

printf("综合测评名次：%d\n", students[i].total\_rank);

printf("\n");

}

}

int main() {

struct Student students[MAX\_STUDENTS];

int count = 0;

load\_student\_data(students, &count);

int choice;

do {

printf("学生综合测评系统\n");

printf("1. 录入学生信息\n");

printf("2. 修改学生信息\n");

printf("3. 删除学生信息\n");

printf("4. 浏览学生信息\n");

printf("5. 录入同学互评分\n");

printf("6. 录入品德成绩\n");

printf("7. 录入任课教师评分\n");

printf("8. 计算综合测评总分\n");

printf("9. 按照综合测评名次输出学生信息\n");

printf("10. 按照单科成绩排名输出学生信息\n");

printf("11. 输出学生综合信息\n");

printf("0. 退出程序\n");

printf("请选择操作：");

scanf("%d", &choice);

switch (choice) {

case 1:

add\_or\_modify\_student(students, &count);

break;

case 2:

add\_or\_modify\_student(students, &count);

break;

case 3:

delete\_student(students, &count);

break;

case 4:

browse\_student\_info(students, count);

break;

case 5:

enter\_peer\_evaluation(students, count);

break;

case 6:

enter\_character\_scores(students, count);

break;

case 7:

enter\_teacher\_evaluation(students, count);

break;

case 8:

calculate\_total\_scores(students, count);

break;

case 9:

print\_student\_info\_by\_total\_rank(students, count);

break;

case 10:

print\_student\_info\_by\_subject\_score(students, count);

break;

case 11:

print\_student\_info(students, count);

break;

case 0:

printf("程序已退出。\n");

break;

default:

printf("无效的操作。\n");

break;

}

printf("\n");

} while (choice != 0);

return 0;

}

实验结果和讨论：

本实验设计并实现了一个学生综合测评系统，能够录入学生信息，计算综合测评总分，并提供了多种功能操作，如修改学生信息、删除学生信息、浏览学生信息、录入考试成绩、录入同学互评分、录入品德成绩、录入任课教师评分、学生数据管理、学生数据查询、按照综合测评成绩名次输出学生信息以及按照单科成绩排名输出学生信息。

本实验还存在一些改进的空间，如增加异常处理机制，确保输入的数据合法性；优化排序算法，提高排序效率；增加更多的功能和操作，如根据不同条件筛选学生信息等。

综合而言，本实验的设计和实现能够满足学生综合测评系统的基本需求，并提供了多种功能操作，为学生信息管理和查询提供了便利。在实验过程中，通过对代码的不断修改和调试，解决了一些问题，并对C语言编程有了更深入的理解。通过实验的结果和讨论，也发现了一些改进的方向，以进一步提升系统的功能和性能。