《C/C++程序设计实践》报告

设计题目：	基于结构体的学生信息管理系统
学院名称：	船舶电气工程学院
专业班级：	物联网工程 2023-1
姓　　名：	温蕊娇
学　　号：	2220232283
指导教师：	柳丽川、胡英
提交日期：	2024.07.26

一、实验内容

编写并调试程序，实现学校各专业班级学生信息的管理。10 个学生的信息存储在文件 studentInit.dat 中。定义学生信息的结构体类型，包括：学号、姓名、专业、班级、3 门成绩；和符号常量 N（学生数）。（同一班级的学生属于同一专业，同一专业的学生可以属于不同的班级）

```
#define N 10
struct Student{
char num[15];//学号
char name[15];//姓名
char major[10];//专业（IoT,automatic,electrical）
int classNo;//班级(1-2)
int score[3];//3 门课的成绩(0-2)
};
typedef struct Student STU;
```

二、实验要求

(1) main 函数：以菜单形式将各项功能提供给用户，根据用户的选择，调用相应的函数。

STU student[N]; //保存输入的 N 名学生信息

(2) 定义函数 void Input(STU *p, int n)：从文件 sutdentInit 中输入 n 个学生的信息。

(3) 定义函数 void Output(STU *p)：将 p 所指的某个学生信息表格化屏幕输出。

(4) 定义函数 STU Fetch(int studentIndex)：从文件中随机读取第 studentIndex 个(0<= studentIndex <=N-1)学生的信息。

(5) 定义函数 void Search(STU *p, int classNo, char s, int scoreSum)：实现班级和成绩的综合查找（如 1 班，总分>240 的同学）。

(6) 定义函数 int Max(STU *p, int scoreIndex)：求所有学生、下标为 scoreIndex 的课程分数最高的学生序号（在数组中的下标），学生序号作为返回值。

(7) 定义函数 void Sort_select(STU *p)：对所有学生，按平均成绩由低到高进行简单选择排序。

(8) 定义函数 void Sort_buble(STU *p, int n)：对某个班级的学生，按平均成绩由高到低进行起泡排序。并调用 Output 输出。

STU stu_class_ave[N]; //按平均成绩排序后的某个班级的学生信息；

int count; //实际元素个数

(9) 定义函数 void Sort_insert(STU *p, int n, char *major)：对某个专业的学生，按某门课程成绩由低到高进行直接插入排序。并调用 Output 输出。

定义局部变量：

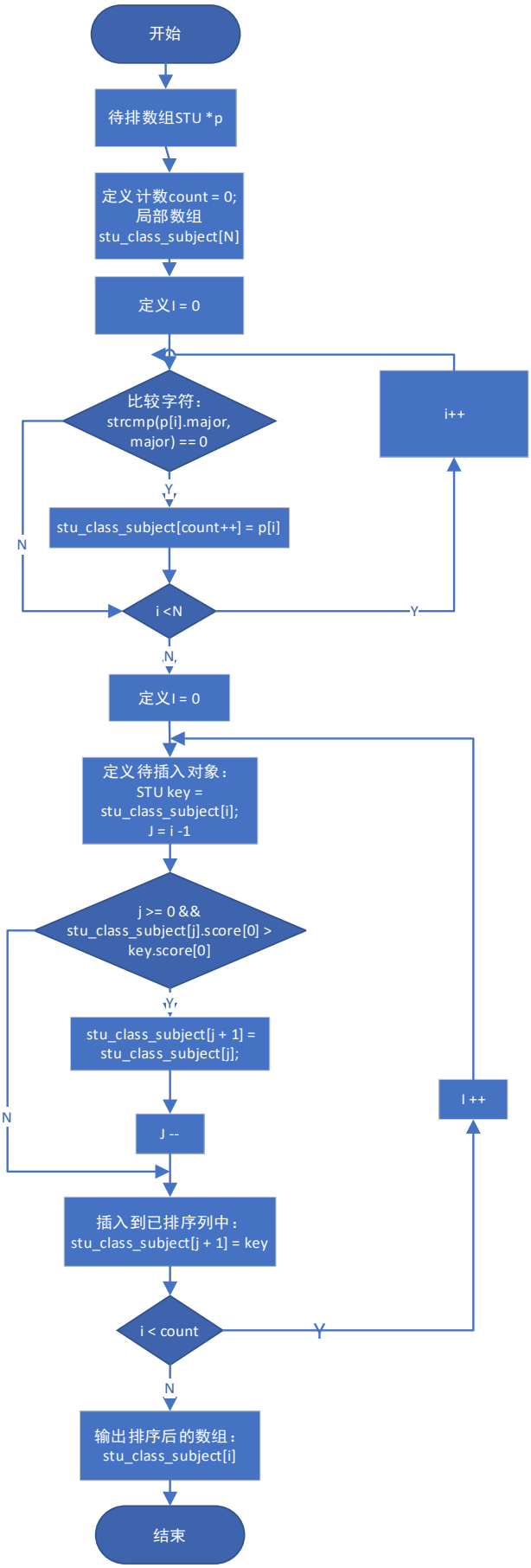
STU stu_class_ subject [N]; //按某门课程成绩排序后的某个专业的学生信息；

int count; //实际元素个数

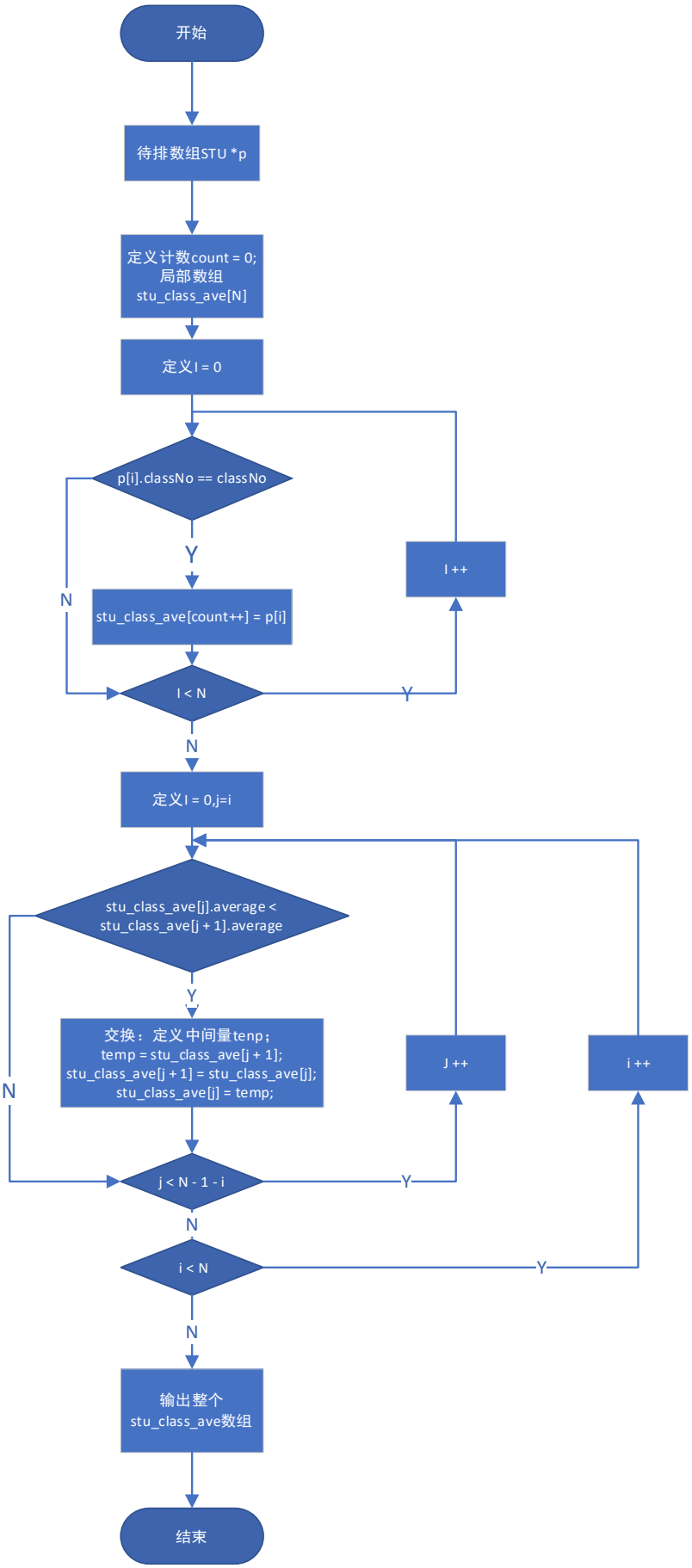
(10) 定义函数 void Save(STU *p,int n)：将学生信息存入文件。

三、算法流程图

1 插入排序流程图


开始
待排数组STU *p
定义计数count = 0;
局部数组
stu_class_subject[N]
定义I = 0
比较字符：
strcmp(p[i].major,
major) == 0
i++
Y
stu_class_subject[count++] = p[i]
N
i <N
Y
N
定义I = 0
定义待插入对象：
STU key =
stu_class_subject[i];
J = i -1
j >= 0 &&
stu_class_subject[j].score[0] >
key.score[0]
Y
stu_class_subject[j + 1] =
stu_class_subject[j];
I ++
N
J --
插入到已排序列中：
stu_class_subject[j + 1] = key
i < count
Y
N
输出排序后的数组：
stu_class_subject[i]
结束

2 冒泡排序流程图


开始
待排数组STU *p
定义计数count = 0;
局部数组
stu_class_ave[N]
定义I = 0
p[i].classNo == classNo
Y
N
stu_class_ave[count++] = p[i]
I ++
I < N
Y
N
定义I = 0,j=i
stu_class_ave[j].average <
stu_class_ave[j + 1].average
Y
N
交换：定义 中间量tenp；
temp = stu_class_ave[j + 1];
stu_class_ave[j + 1] = stu_class_ave[j];
stu_class_ave[j] = temp;
J ++
i ++
j < N - 1 - i
Y
N
i < N
Y
N
输出整个
stu_class_ave数组
结束

四、程序清单

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#define N 10       //不会占用内存

struct Student{
      char num[15];       //学号
      char name[15];      //姓名
      char major[10];     //专业（IoT,automatic,electrical）
      int classNo;        //班级(1-2)
      int score[3];       //3 门课的成绩(0-2)
      float average;      // 平均成绩
      };
typedef struct Student STU;//结构体变量

/*函数声明列表*/
void Input(STU *p, int n);
void Output(STU *p);
STU Fetch(int studentIndex);
int Max(STU *p, int scoreIndex);
void Search(STU *p, int classNo, char s, int scoreSum);
void Sort_buble(STU *p, int n);
void Sort_select(STU *p);
void Save(STU *p,int n);
void Sort_insert(STU *p, int n, char *major);
void calculate_average(STU *p, int n);

int main(){
      int n = 0;
      STU Student[N+n];      // 定义学生数组
      srand(time(0));        // 初始化随机数种子
      char choice;           // 用于选择是否结束菜单
      int num;
      do {
            printf("                        学生信息管理系统                        \n");
            printf("      ---------------------    ------------------------------ \n");
            printf("10.写入学生信息（文件中已有数据可按‘0’读取）//推荐先‘0’后‘10’\n");
            printf("      ---------------------    ------------------------------ \n");
            printf("       0.从文件中读取学生信息      1.输出全部学生信息          \n");
            printf("       2.随机读取学生信息          3.班级成绩综合查询          \n");
            printf("      ---------------------    ------------------------------ \n");
            printf("       4.查询课程最高分            5.倒序选择排序全员平均分    \n");
```

```
        printf("          6.正序冒泡排序班级平均分       7.倒序插入排序课程成绩          \n");
        printf("         ----------------------      ------------------------------- \n");
        printf("          8.保存学生信息到文件里       9.退出程序                      \n");
        printf("         ----------------------      ------------------------------- \n");          printf("请选择功能
编号:");
        scanf("%d", &num);
        switch (num){
            case 0:
                Input(Student,N+n);                  //从文件录入数据
                calculate_average(Student,N+n);
                break;
            case 1:
                printf("学生信息输出:\n");
                for(int i = 0;i < N+n;i ++){
                    Output(&Student[i]);       //输出数据
                }
                system("pause");
                break;
            case 2:{
                int index = rand() % N+n;      //生成随机索引
                STU random = Fetch(index);   //  根据索引获取随机学生信息
                Output(&random);
                break;
            }
            case 3:{
                int classNo;
                int scoreSum;
                char s;
                printf("输入班级(1 或 2): ");
                scanf("%d", &classNo);
                printf("输入比较符(>、<、=): ");
                scanf(" %c", &s);                    //  前面加空格，跳过前一个输入后的换行符
                printf("输入成绩总分: ");
                scanf(" %d", &scoreSum);
                Search(Student,classNo,s,scoreSum);
                break;
            }
            case 4:{
                int scoreIndex;
                printf("输入课程序号(0-2): ");
                scanf("%d", &scoreIndex);

                int idx = Max(Student,scoreIndex);
```

```
            if (idx != -1) {
                Output(&Student[idx]);
            } else {
                printf("未找到学生\n");
            }
            break;
        }
        case 5:{
            Sort_select(Student);
            break;
        }
        case 6:{
            STU stu_class_ave[N+n];
            int classNo;
            printf("输入班级: ");
            scanf("%d", &classNo);
            Sort_buble(Student,classNo);
            break;
        }
        case 7:{
            STU stu_class_subject[N+n];
            char major[10];
            printf("输入专业: ");
            scanf("%s", major);
            Sort_insert(Student,N+n,major);
            break;
        }
        case 8:{
            Save(Student,N+n);
            printf("保存完毕！\n");
            break;
        }
        case 9:
            printf("即将退出程序！\n");
            exit(0);
            break;
        case 10:{
            printf("请输入学生数量\n");
            scanf(" %d",&n);
            input(Student,n);                    //写入数据
            calculate_average(Student,n);
            break;
        }
        default:
```

```
                printf("请在 0-12 之间选择\n\n\n");
                break;
            }
            printf("是否继续操作？(y/n): ");
            scanf(" %c",&choice);
        } while (choice == 'y' || choice == 'Y');  //  判断用户是否继续操作
}



//从文件载入信息 over
void Input(STU *p,int n){
    FILE *file = fopen("studentInit.dat", "rb");        //只读，二进制，
    if (file == NULL) {                                 //若文件不存在则打开失败
        printf("无法打开文件\n");
        return;
    }
    fread(p,sizeof(STU),n,file);
    fclose(file);
    printf("已从文件中录入学生信息！\n");
}


//根据指针显示某同学信息 over
void Output(STU *p) {
    printf("|-----------------------|\n");
    printf("|  学号: %-25s |\n", p->num);
    printf("|  姓名: %-25s |\n", p->name);
    printf("|  专业: %-25s |\n", p->major);
    printf("|  班级: %-25d |\n", p->classNo);
    printf("|  三门课成绩: %-5d    %-5d %-5d|\n",p->score[0],p->score[1],p->score[2]);
    printf("|-----------------------|\n");
}

//查找特定(随机)的同学 over
STU Fetch(int studentIndex) {
    FILE *file = fopen("studentInit.dat", "rb");                //只读，二进制，
    STU student;
    if (file == NULL) {                                         //若文件不存在则打开失败
        printf("无法打开文件\n");
        return student;
    }                                                           //文件指针位于文件开头
    fseek(file, studentIndex * sizeof(STU), SEEK_SET);          //将文件指针移动，sizeof(STU)表示
STU 大小，SEEK_SET 从文件头偏移
```

```
    fread(&student, sizeof(STU), 1, file);

    fclose(file);
    return student;
}

//根据班级和成绩查找符合条件的同学 0ver
void Search(STU *p, int classNo, char s, int scoreSum){
    int m=0;                                                    //是否找到学生
    for (int i = 0; i < 10; i++) {
        int sum = p[i].score[0] + p[i].score[1] + p[i].score[2];
        if (p[i].classNo == classNo) {
            if ((s == '>' && sum > scoreSum) ||(s == '<' && sum < scoreSum) ||(s == '=' && sum ==
scoreSum)) {
                Output(&p[i]);
                m = 1;
            }
        }
    }
    if (!m) {
        printf("未找到符合条件的学生\n");
    }
}

//找出当前班级下标最大的同学 over
int Max(STU *p, int scoreIndex) {
    FILE *file = fopen("studentInit.dat", "rb");
    if (file == NULL) {
        printf("无法打开文件\n");
        return -1;
    }

    fread(p, sizeof(STU), N, file);//从文件中取 N 个 sizeof(STU) 大小的数据块到数组 p 中。

    int maxIdx = -1;                    //最大分数对应的学生索引
    int maxScore = -1;                  //保证最大分数初始最小
    for (int i = 0; i < N; i++) {
        if (p[i].score[scoreIndex] > maxScore) {
            maxScore = p[i].score[scoreIndex];
            maxIdx = i;
        }
    }
    fclose(file);
    return maxIdx;                  // 返回找到的最大分数对应的学生在数组中的索引
```

```
}

//平均成绩由低到高进行简单选择排序。over
void Sort_select(STU *p){
	int j,i;
	STU temp;

	for(i = 0;i < N - 1;i ++){
		int min_i = i;
		//找最小
		for (j = i + 1; j < N; j++) {
			int sum_i = p[min_i].score[0] + p[min_i].score[1] + p[min_i].score[2];
			int sum_j = p[j].score[0] + p[j].score[1] + p[j].score[2];
			if (sum_j < sum_i) {
				min_i = j;
			}
		}
		// 交换找到的成绩最小学生到当前位置
		if (min_i != i) {
			temp = p[i];
			p[i] = p[min_i];
			p[min_i] = temp;
		}
	}
	printf("选择排序完成！\n");
}

//按平均成绩由高到低进行起泡排序。
void Sort_buble(STU *p, int classNo){
	STU stu_class_ave[N];
	int count = 0;
	// 复制原始数组到局部数组
	for (int i = 0; i < N; i++) {
		if (p[i].classNo == classNo) {
			stu_class_ave[count++] = p[i];
		}
	}
	for(int i = 0;i < N - 1;i ++){
		for (int j = i;j < N - 1 - i;j++) {
			if (stu_class_ave[j].average < stu_class_ave[j + 1].average) {
				STU temp = stu_class_ave[j + 1];
				stu_class_ave[j + 1] = stu_class_ave[j];
				stu_class_ave[j] = temp;
			}
```

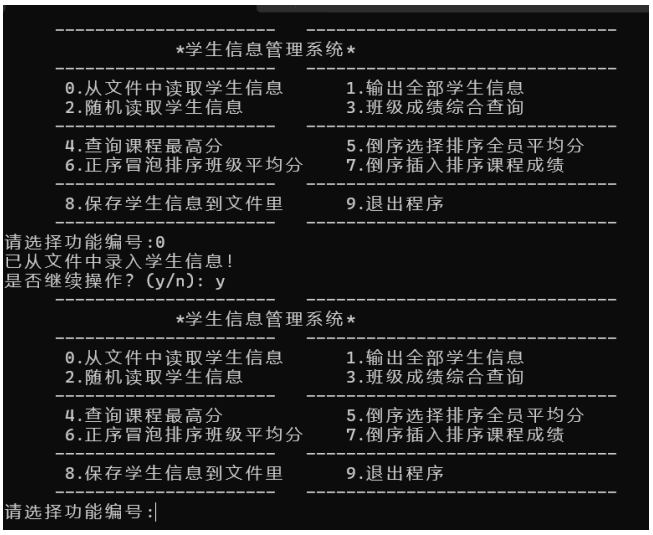
```
            }
        }
        for (int i = 0; i < count; i++) {
            Output(&stu_class_ave[i]);
        }
        printf("冒泡排序完成！\n");
}


//按某门课程成绩由低到高进行直接插入排序
void Sort_insert(STU *p, int n, char *major){
        STU stu_class_subject[N];
        int count = 0;                                          //有效的学生个数
        //特定专业的学生复制到局部数组
        for (int i = 0; i < N; i++) {
            if (strcmp(p[i].major, major) == 0) {     //比较字符串内容
                stu_class_subject[count++] = p[i];
            }
        }
        for (int i = 1; i < count; i++) {
            STU key = stu_class_subject[i];           //保存当前待插入的学生对象
            int j = i - 1;
            //从当前位置向前遍历已排序部分
            while(j >= 0 && stu_class_subject[j].score[0] > key.score[0])
{                   stu_class_subject[j + 1] = stu_class_subject[j];
                    j--;
            }
            stu_class_subject[j + 1] = key;               // 将 key 插入到已排序序列的合适位置
        }
        for (int i = 0; i < count; i++) {
            Output(&stu_class_subject[i]);
        }
        printf("排序完成！\n");
}

void AddStudent(STU *p,int n){
        FILE *fp;                                          //‘W’模式下，文件指针位于文件开头
        fp=fopen("studentInit.dat","wb");                  //只写，打开文件后，会清空文件内原有的内容。
        if(fp==NULL) {                                     //若文件不存在，则新建文件；
            printf("打开文件错误\n");
        }

        for(int i=0;i<n;i++) {
            printf("请输入第%d 位学生的学号\n",i+1);
```

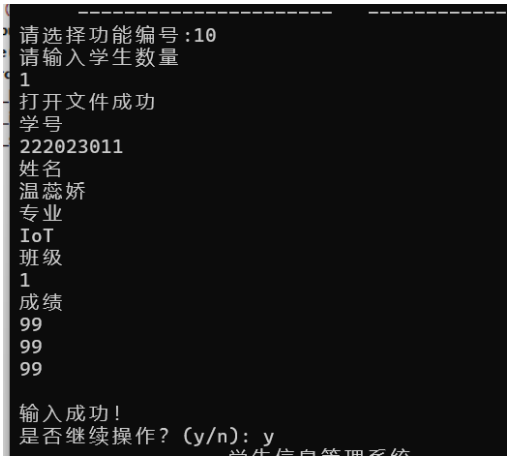
```
            scanf("%s",p[i].num);
            getchar();                          //清除输入缓冲区中的回车符或空格
            printf("请输入第%d 位学生的姓名\n",i+1);
            scanf("%s",p[i].name);
            getchar();
            printf("请输入第%d 位学生的专业\n",i+1);
            scanf("%s",p[i].major);
            getchar();
            printf("请输入第%d 位学生的班级\n",i+1);
            scanf("%d",&p[i].classNo);
            getchar();
            printf("第一门课成绩: ");
            scanf("%d", &p[i].score[0]);
            getchar();
            printf("第二门课成绩: ");
            scanf("%d", &p[i].score[1]);
            getchar();
            printf("第三门课成绩: ");
            scanf("%d", &p[i].score[2]);
            getchar();
            fwrite(&p[i], sizeof(STU), 1, fp);
        }
        fclose(fp);
        printf("\n 学生信息添加成功！ \n");
}

// 删除指定学号的学生信息
void DeleteStudent(STU *p, int n, char *studentNum) {
        int found = 0; // 标记是否找到学生
        for (int i = 0; i < n; i ++) {
            if (strcmp(p[i].num, studentNum) == 0) {                // 找到匹配的学号
                found = 1;
                // 将后面的学生信息前移，覆盖掉要删除的学生信息
                for (int j = i; j < n - 1; j++) {
                    p[j] = p[j + 1];
                }
                n--;                                                // 更新学生数量
                printf("学生信息已删除。\n");
                break;
            }
        }
        if (!found) {
            printf("未找到学号为 %s 的学生。\n", studentNum);
        }
```

```
    // 更新文件中的学生信息
    Save(p, n);
}

//将学生信息存入文件
void Save(STU *p,int n){
    FILE *fp;                                   // 文件指针
    // 打开文件，"w"模式表示写入，如果文件不存在则创建
    fp = fopen("studentInit.dat", "wb");        //以二进制写入
    if (fp == NULL) {
        printf("文件打开错误\n");
        return;
    }
    //写入数据到文件，并检查是否发生了错误，即写入的元素数量是否少于 N
    if (fwrite(p, sizeof(STU), n, fp) != N) {
        printf( "数据录入错误.\n");
    }
    fclose(fp);
    printf("保存成功\n");
}

//计算每个同学的平均成绩
void calculate_average(STU *p, int n){
    for(int i = 0 ; i< n ; i++){
        p[i].average = (p[i].score[0]+p[i].score[1]+p[i].score[2])/3.0;
    }
}
```

五、程序测试（Input、Search、Max、Sort_buble、Sort_insert 函数的调用及执行结果的截图）

Input（）





```
请选择功能编号:3
输入班级(1或2): 1
输入比较符(>、<、=): >
输入成绩总分: 150
| 学号: 222023003
| 姓名: 张三
| 专业: IoT
| 班级: 1
| 三门课成绩: 33     86    40
| 平均成绩: 53.00
|------------------------
| 学号: 222023009
| 姓名: 林九
| 专业: automatic
| 班级: 1
| 三门课成绩: 74     48    43
| 平均成绩: 55.00
|------------------------
```

Search（）

```
     ---------------------   ------------------------------
                *学生信息管理系统*
     ---------------------   ------------------------------
      0.从文件中读取学生信息       1.输出全部学生信息
      2.随机读取学生信息           3.班级成绩综合查询
     ---------------------   ------------------------------
      4.查询课程最高分             5.倒序选择排序全员平均分
      6.正序冒泡排序班级平均分     7.倒序插入排序课程成绩
     ---------------------   ------------------------------
      8.保存学生信息到文件里       9.退出程序
     ---------------------   ------------------------------
请选择功能编号:4
输入课程序号(0-2): 1
|------------------------|
| 学号: 222023006        |
| 姓名: 刘六             |
| 专业: automatic        |
| 班级: 2                |
| 三门课成绩: 61     100   65   |
|------------------------|
是否继续操作? (y/n):
```

Max

```
请选择功能编号:6
输入班级: 1
| 学号: 222023003
| 姓名: 张三
| 专业: IoT
| 班级: 1
| 三门课成绩: 33     86    40
| 平均成绩: 53.00
|------------------------
| 学号: 222023007
| 姓名: 程七
| 专业: automatic
| 班级: 1
| 三门课成绩: 35     7     94
| 平均成绩: 45.33
|------------------------
| 学号: 222023009
| 姓名: 林九
| 专业: automatic
| 班级: 1
| 三门课成绩: 74     48    43
| 平均成绩: 55.00
|------------------------
| 学号: 222023005
| 姓名: 王五
| 专业: IoT
| 班级: 1
| 三门课成绩: 7      6     92
| 平均成绩: 35.00
|------------------------
| 学号: 222023001
| 姓名: 陈一
| 专业: IoT
| 班级: 1
| 三门课成绩: 35     1     35
| 平均成绩: 23.67
|------------------------
冒泡排序完成!
```

```
请选择功能编号：7
输入专业：IoT
| 学号：222023005
| 姓名：王五
| 专业：IoT
| 班级：1
| 三门课成绩：7      6     92
| 平均成绩：35.00
|------------------------
| 学号：222023004
| 姓名：李四
| 专业：IoT
| 班级：2
| 三门课成绩：21     48    61
| 平均成绩：43.33
|------------------------
| 学号：222023003
| 姓名：张三
| 专业：IoT
| 班级：1
| 三门课成绩：33     86    40
| 平均成绩：53.00
|------------------------
| 学号：222023001
| 姓名：陈一
| 专业：IoT
| 班级：1
| 三门课成绩：35     1     35
| 平均成绩：23.67
|------------------------
| 学号：222023002
| 姓名：黄二
| 专业：IoT
| 班级：2
| 三门课成绩：75     31    84
| 平均成绩：63.33
|------------------------
排序完成！
是否继续操作？(y/n):
```

Sort_insert（）

六、实验总结（实验过程中遇到的具体问题，如何解决，不要说空话、套话，雷同扣分）

0. 这是一个信息管理系统，但如果每次运行都手动输入很多数据，或者只能从文件读取数据，都不符合需求，于是为了方便管理信息，选择了两种输入模式，功能 0 是单独的读取文件数据，而 10 则是单独输入数据并保存到文件里。

1.在选择了一次系统功能后，运行就会自动结束：利用 do while 循环，并加入一个判断交互，让用户选择是否继续菜单模式，来判断是否循环。

2.文件内容正确，读取却出现乱码：在查阅资料后得知 dat 格式的文件为二进制文件，“r”“w”是对应文本文件的，需要变成“rb”“wb”，例如：FILE *file = fopen("studentInit.dat", "rb");

3.在编写 search 函数时，因为要求综合查询，还要遍历所有信息，循环套循环再套几个判断语句太混乱，于是选择更换写法，用了&&和||语句，减少多次循环加判断的麻烦，思路也清晰了不少。

4.查找对应要求的姓名与专业时，系统会报错：通过把报错黏贴至搜索引擎，知道了要用字符串比较，且 strcmp()函数的输出为 0/1，则改写代码为 if (strcmp(p[i].major, major) == 0)。

5.排序函数中有些需要根据平均分排序，那么输出结果的时候为了可视化也要输出平均分，但是原结构体并没有平均分，如果单独在函数中加上对平均分的输出，与学生信息对不上号，且容易报错，没成功，于是单独在原结构体上增加了 float average，并并在导入学生信息的时候，就让同步计算了平均分并写入该结构体变量，以此达到输出平均分的结果。

6.运行结果显示文件打开正确但是无法输出对应文件内容：先运行导入数据功能，在运行其他功能函数，即要先有 input 才能做后面的操作。

7. 多个 scanf()无法输入：报错后上网查阅资料，了解到是由于该函数的缓冲区导致的，有两种方法解决，一是在%d 前面加空格，二是在 scanf 后用 getchar 在写下一个 scanf。

8.switch case 在每一个 case 都会报错：在 case 的冒号后加上{}并 break。

《C/C++程序设计实践》成绩评价表

课程名称	C/C++程序设计实践			
题目名称	基于结构体的学生信息管理系统			
学生姓名	温蕊娇	学号	2220232283	
序号	评价项目	指　　标	满分	评分
1	**课堂表现**	预习充分、无迟到、实验认真不做与实验无关活动；设计方案分析充分，思路清晰，逻辑正确，有创新特色；针对设计方案，能够合理分工，任务清晰，及时解决问题；在设计与试验过程中，能够全面考虑系统设计过程中所面临的人力、物力、财务的问题。	40	
2	**实践报告**	报告格式规范，语句通顺，图表程序正确，分析论证充分，实验结果表述清晰、正确；在报告中，能够全面分析系统设计过程中所面临的人力、物力、财务的问题。	30	
3	**实践**	答辩时能系统介绍设计方案，思路清晰，逻辑正确，有创新特色；汇报规范，语句通顺，图表程序正确，分析论证充分，实验结果表述清晰、正确。	30	
总分				
评语：				

指导教师：（签名）　　年　　月　　日