《C语言程序设计实训1》报告

设 计 题 目： 基于数组的学生信息管理系统

学 院 名 称： 信息科学技术学院

专 业： 网络工程

班 级： 二班

姓 名： 李毓琪 学 号 2220191195

指 导 教 师： 张海昕

提 交 日 期：  2020年7月10日

**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。在头文件student.h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）。（同一班级的学生可以属于不同的专业，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（computer,software,network）

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

保存学生信息的文件studentInit.dat要事先创建好，不包含在菜单功能中

1. 定义函数void Input(STU \*p, int n)：从文件studentInit.dat中输入N名学生的信息。（从存储设备到内存）
2. 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。（用于各个需要输出的函数调用）
3. 定义函数STU Fetch(int studentIndex)：从文件中读取指定的第studentIndex个(0<= studentIndex <=N-1)学生的信息，并显示。

增加内容：按照学号或者姓名查询学生信息，考虑查询不到的情况

1. 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）。

查询时：输入班级，分数，高于或低于（>或者<），形式自定>0, <310显示所有该班同学信息

1. 定义函数int Max(STU \*p, int scoreIndex)：求所有学生中、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。

课程号在程序中数组下标为012，实际使用中最好用123，或者高数，英语，物理，符合日常习惯。如何避免出错！

1. 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。
2. 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（在屏幕上显示）。考虑：怎样把查询到的结果另存为新的文件？

定义局部变量：//存放部分学生的信息

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

1. 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：//存放部分学生的信息

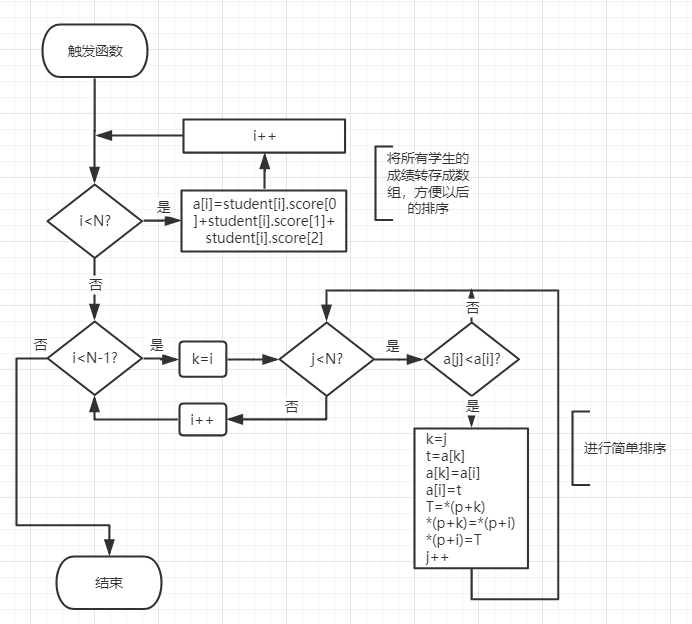
STU stu\_class\_ subject [N]; //按某门课程成绩排序后的某个专业的学生信息；

int count; //实际元素个数

1. 定义函数void Save(STU \*p,int n) ：将学生信息存入文件。例如排序后，存为abc.dat。可以使用1号功能打开它。

**三、算法流程图**

函数Sort\_select的算法流程图



**四、程序清单**（关键语句和变量加注释）

#include <stdio.h>

#include <string.h>

#include <stdlib.h>//exit()

#define N 10//定义N=10 即学生的数量

struct Student{//定义结构体Student

char num[15];//学号

char name[15];//姓名

char major[10];//专业(computer,software,network)

int Class;//班级（1-2）

int score[3];//3门课的成绩（0-2）

}student[N],stu[N];

typedef struct Student STU;//定义STU

void Save(STU \*p,int n);//保存信息

void Write(STU \*p); //编写1名同学信息

void Input(STU \*p, int n);//输入信息

void Output(STU \*p);//格式化显示同学的信息

STU Fetch(int studentIndex); //查找第n名学生的信息

void Search(STU \*p,int ClassNo,char s,int scoreSum);//查找

int Max(STU \*p, int scoreIndex);//查找最大成绩的学生

void Sort\_select(STU \*p);//简单排序

void Buble\_Sort(STU \*p);//起泡法排序

void insert\_Sort(STU \*p,int n);//直接插入排序

int Judgeread(STU \*p);

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 main(){

int choice,i=0,j,n;

while(1){

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");//美化

printf("\t主菜单\n");

printf("0.重新输入学生信息到内存中\n");

printf("1.查看现内存的所有信息\n");

printf("2.保存学生信息\n");

printf("3.从文件中读取学生信息\n");

printf("4.从文件中提取第n名学生的信息\n");

printf("5.通过指定信息筛选学生\n");

printf("6.查找单科成绩最高的学生序号\n");

printf("7.对所有学生的总成绩进行简单排序\n");

printf("8.对某个班级的学生按总成绩有高到低进行起泡排序\n");

printf("9.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\n");

printf("其他.退出\n");

printf("注：7、8、9排序功能将符合条件的学生排在前面并输出，不符合条件的学生于符合条件的后面\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("请输入功能序号，运行相应功能：");

scanf("%d",&choice);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

switch (choice){

case 0://输入学生信息

while(i<N){

getchar();//避免回车当成文件名

printf("请输入第""%d""名学生的信息\n",i+1);

Write(student+i); //编写信息

printf("停止录入请按输入0，继续录入直接按回车键：");

scanf("%d",&j);//键入0时终止输入信息

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

if(j==0)

break;

i++;

}

printf("一共输入了""%d""名学生的信息,按回车键返回主菜单\n",i+1);

getchar();

break;

case 1://输出现在内存中的所有学生信息

if(student[n].Class==0)//判断是否有学生

printf("内存内无学生信息\n");

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

Output(student+i);

printf("按回车键返回主菜单");

getchar();getchar();

break;

case 2://保存学生信息

getchar();//避免回车当成文件名

Save(student,sizeof(student)/sizeof(Student) );// 保存信息

getchar();

break;

case 3://输入学生信息

getchar();

Input(student,n);//输入信息;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for(j=0; j<N; j++){//将学生信息一次输出

Output(student+j);

}

printf("按回车键返回主菜单");

getchar();

break;

case 4://从文件中查看第n名学生的信息；

printf("请输入要查看同学的序号(1<=n<=10)：");

scanf("%d",&n);

getchar();

n--;

if(student[n].Class==0){//判断该序号内是否有人

printf("!!查无此人,按回车键返回主菜单");

getchar();

break;

};

Fetch(n);//查找第n位学生的信息

printf("第%d名学生的信息为：\n",n+1);

Output(student+n);

printf("按回车键返回主菜单");

getchar();

break;

case 5://根据班级和总成绩进行查找

if(Judgeread(student)==0){

printf("!!请先打开文件,按回车键返回主菜单");

getchar();getchar();

break;

}

printf("请选择：\n1.通过班级和限定成绩筛选学生信息\n2.通过学号或者姓名筛选学生信息\n");

int choice;

scanf("%d",&choice);//选择功能

switch(choice){

case 1:

printf("请输入需要查询的班级:");

int ClassNo,scoreSum,s;

getchar();

scanf("%d",&ClassNo);

if(ClassNo!=2&&ClassNo!=1){

printf("请输入正确的班级,按回车键返回主菜单");

getchar();getchar();

break;

}

printf("请输入查询的总成绩的值:");

getchar();

scanf("%d",&scoreSum);

getchar();

Search(student,ClassNo,s,scoreSum);//查找符合要求的学生，并输出

printf("按回车键返回主菜单");

getchar(); getchar();

break;

case 2://通过学号或者姓名筛选学生信息

char Num[15],Name[15];

int b;

printf("请输入需要进行筛选的类型：1.学号；2.姓名:");

scanf("%d",&choice);//选择需要筛选的信息

getchar();

b=0;

switch(choice){

case 1:

printf("请输入需要查找的学号:");

scanf("%s",&Num);

for(j=0;j<N;j++)

if(strcmp(Num,student[j].num)==0){//通过比较学生学号输出符合要求的学生

Output(student+j);

b=1;

}

if(b==0){

printf("!!查无此人,按回车键返回主菜单");

break;

}

break;

case 2:

printf("请输入需要查找的姓名:");

scanf("%s",&Name);

for(j=0;j<N;j++)

if(strcmp(Name,student[j].name)==0){//通过比较学生姓名输出符合要求的学生

Output(student+j);

b=1;

}

if(b==0){

printf("!!查无此人,");

break;

}

break;

default:

printf("请选择正确的序号，按回车键返回主菜单");

getchar();

break;

}

printf("按回车键返回主菜单");

getchar(); getchar();

break;

default:

printf("请选择正确的序号，按回车键返回主菜单");

getchar(); getchar();

break;

}

break;

case 6://查找单科成绩最高的学生序号

if(Judgeread(student)==0){

printf("!!请先打开文件,按回车键返回主菜单");

getchar();getchar();

break;

}

printf("请输入需要查找的学科：（0=高数，1=线代，2=C语言）:");

int scoreIndex;

scanf("%d",&scoreIndex);

if(scoreIndex==1||scoreIndex==2||scoreIndex==0){

getchar();

Max(student,scoreIndex);//查找单科成绩最高的学生，并输出其学号和信息

}

else{

printf("!!输入错误,");

getchar();

}

printf("按回车键返回主菜单");

getchar();

break;

case 7://对所有学生进行简单排序

if(Judgeread(student)==0){

printf("!!请先打开文件,按回车键返回主菜单");

getchar();getchar();

break;

}

Sort\_select(student);

printf("排列成功！以下为所有学生按照总成绩由低到高的简单排序\n");

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

Output(student+i);

printf("按回车键返回主菜单");

getchar(); getchar();

break;

case 8://对某个班级的学生按总成绩有高到低进行起泡排序

if(Judgeread(student)==0){

printf("!!请先打开文件,按回车键返回主菜单");

getchar();getchar();

break;

}

char Major[10];

int ClassNo;

getchar();

printf("请输入需要排序的专业:network、software、computer:\n ");

gets(Major);

if(strcmp(Major,"network")!=0)

if(strcmp(Major,"software")!=0)

if(strcmp(Major,"computer")!=0){

printf("!!!输入错误,按回车键返回主菜单");

getchar();

break;

};

printf("请输入需要排序的班级：");

scanf("%d",&ClassNo);

if(ClassNo!=1&&ClassNo!=2){

printf("!!!输入错误,按回车键返回主菜单");

getchar();getchar();

break;

};

getchar();

Buble\_Sort(student);

printf("排列成功！以下为%s专业%d班级学生按总成绩的冒泡排序\n",Major,ClassNo);

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

if(strcmp(student[i].major,Major)==0)

if(student[i].Class==ClassNo)

Output(student+i);

printf("按回车键返回主菜单");

getchar();

break;

case 9://对某个专业的学生按某门课程成绩进行直接插入排序

if(Judgeread(student)==0){//判断内存中是否有文件

printf("!!请先打开文件,按回车键返回主菜单");

getchar();getchar();

break;

}

char MajorNo[10];

int ScoreNo;

getchar();

printf("请输入需要排序的专业network、software、computer：");

gets(MajorNo);//输入专业

if(strcmp(MajorNo,"network")!=0)

if(strcmp(MajorNo,"software")!=0)

if(strcmp(MajorNo,"computer")!=0){

printf("!!!输入错误,按回车键返回主菜单");

getchar();

break;

};

printf("请输入需要排序的学科(0=高数 1=线代 2=C语言)：");

scanf("%d",&ScoreNo);//选择学科

if(ScoreNo!=0&&ScoreNo!=1&&ScoreNo!=2){

printf("!!!输入错误,按回车键返回主菜单");

getchar();getchar();

break;

}

insert\_Sort(student,ScoreNo);//对所有学生对该成绩进行简单排序

printf("排列成功！以下为对%s专业的学生按照单科成绩成绩进行直接插入排序\n",MajorNo);

for(i=0,j=0;i<N;i++)

if(strcmp(student[i].major,MajorNo)==0){//选择符合条件的学生输出

Output(student+i);

stu\_class\_subject[j]=student[i];

j++;

};

int cho;

printf("请问是否要保存%s专业学生按照单科成绩排序到文件?1.是0.否",MajorNo);

scanf("%d",&cho);

if(cho==1){

getchar();

Save(stu\_class\_subject,sizeof(stu\_class\_subject)/sizeof(Student));

}

printf("按回车键返回主菜单");

getchar();

break;

default:

exit(0);

}

}

}

int Judgeread(STU \*p){

if(student[0].Class<=0)//判断内存是否有学生

return 0;

else{

return 1;

}

}

void Write(STU \*p){

printf("\n请输入学号：");

gets(p->num);

printf("\n请输入姓名: ");

gets(p->name);

printf("\n请输入专业: ");

gets(p->major);

printf("\n请输入班级：");

scanf("%d",&p->Class);

printf("\n请输入三门课的成绩，每门课程用空格隔开：\n");

scanf("%d %d %d",&p->score[0],&p->score[1],&p->score[2]);

getchar();

}

void Save(STU \*p,int n){

FILE \*fp;//打开文件

char filename[20];

int i;

printf("请输入文件名：");

gets(filename);

if((fp=fopen(filename,"wb"))==NULL){//创建二进制文件

printf("文件打开错误！\n");

fclose(fp);

return;

}

for(i=0;i<N;i++){

if(fwrite(p++,sizeof(STU),1,fp)!=1){//向文件中一次写一个结构体量值

printf("文件写入错误!\n");

fclose(fp);

return;

}

}

fclose(fp);//关闭文件

printf("保存成功!按任意键返回主菜单");

}

void Input(STU \*p,int n){

char filename[20];

FILE \*fp;

printf("要打开的文件名：");

gets(filename);

if((fp=fopen(filename,"rb"))==NULL){ //打开文件，并判断是否打开成功

printf(" 文件打开失败！\n ");

fclose(fp);

return;

}

if((fread(p,sizeof(Student),N,fp))==0){ //读取文件，并判断是否读取成功

printf(" 文件读取失败！\n");

fclose(fp);

return;

}

fclose(fp);

}

void Output(STU\*p){

if(p->num&&p->Class)//避免输出没用的空行

printf("学号:%s\t姓名：%-15s\t专业:%s\t班级:%d 高数:%d 线代:%d C语言:%d 总成绩：%d\n",p->num,p->name,p->major,p->Class,p->score[0],p->score[1],p->score[2],p->score[1]+p->score[2]+p->score[0]);

}

STU Fetch(int studentIndex){

FILE \*fp;//打开文件

char filename[20];

int i,j;

printf("请输入文件名：");

gets(filename);

if((fp=fopen(filename,"rb"))==NULL){//创建二进制文件

printf("文件打开错误！\n");

}

fseek(fp,sizeof(STU)\*studentIndex,0);//将指针后移studentIndex个结构体

if((fread(&student[studentIndex],sizeof(STU),1,fp))!= 1){ //读取该结构体的内容

printf(" 文件读取失败，请重试！");

}

fclose(fp);

}

void Search(STU \*p,int ClassNo,char s,int scoreSum){

int n,m,i,j=0,com;

printf("请输入与总成绩之间的关系：大于（1）小于（2）等于（0）:");

scanf("%d",&com);

switch(com){

case 2:

for(i=0;i<N;i++){

if(student[i].Class==ClassNo&&student[i].score[1]+student[i].score[2]+student[i].score[0]<scoreSum)//比较学生成绩和 目标成绩

Output(student+i);

j++;

};

break;

case 1:

for(i=0;i<N;i++){

if(student[i].Class==ClassNo&&student[i].score[1]+student[i].score[2]+student[i].score[0]>scoreSum)//比较学生成绩和 目标成绩

Output(student+i);

j++;

};

break;

case 0:

for(i=0;i<N;i++){

if(student[i].Class==ClassNo&&student[i].score[1]+student[i].score[2]+student[i].score[0]==scoreSum)

Output(student+i);

j++;

};

break;

default:

printf("请输入正确的数字");

return;

}

if(j==0)

printf("无学生信息");

}

int Max(STU \*p, int scoreIndex){

int j,n,tem=0,a[N];

for(j=0;j<N;j++)

a[j]=0;

for(j=0;j<=N;j++)

if(student[j].score[scoreIndex]>student[tem].score[scoreIndex])//比较学生成绩和目标成绩

tem=j;

for(j=0;j<N;j++)

if(student[j].score[scoreIndex]==student[tem].score[scoreIndex])

a[j]=j;

printf("该学科成绩最高的学生序号为");

for(j=0;j<N;j++)

if(a[j]!=0)

printf("%d ",a[j]+1);

printf("\n");

for(j=0;j<N;j++)

if(a[j]!=0)

Output(student+a[j]);

}

void Sort\_select(STU \*p){

int a[N],i,j,k,t;//a[i]为该学生总成绩

Student T;

for(i=0;i<N;i++)//为a[i]赋值

a[i]=student[i].score[0]+student[i].score[1]+student[i].score[2];

for(i=0;i<N-1;i++){//ij是循环控制变量

k=i;//k是每一轮最大元素的下标

for(j=i+1;j<N;j++)

if(a[j]<a[i]){

k=j;//k赋值

t=a[k];//将总成绩进行交换

a[k]=a[i];

a[i]=t;

T=\*(p+k);//对结构体进行交换

\*(p+k)=\*(p+i);

\*(p+i)=T;

}

}

}

void Buble\_Sort(STU \*p){

int a[N],i,j,t;//a[i]为该学生总成绩

Student T;

for(i=0;i<N;i++)//为a[i]赋值

a[i]=student[i].score[0]+student[i].score[1]+student[i].score[2];

for(i= 0;i<N;i++)//冒泡排序

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

if(a[j]>a[j+1]){

t=a[j];//将总成绩进行交换

a[j]=a[j+1];

a[j+1]=t;

T=\*(p+j);//对结构体进行交换

\*(p+j)=\*(p+j+1);

\*(p+j+1)=T;

}

}

void insert\_Sort(STU \*p,int n){

int a[N],i=0,j=0,t=0;//a[i]为该学生该学科的成绩

Student T;

for(i=0;i<N;i++)//为a[i]赋值

a[i]=student[i].score[n];

for (i=1;i<N;i++){//进行直接插入排序

t=a[i];

T=\*(p+i);

for(j=i-1;j>=0;j--){

if(a[j]>t){

a[j+1]=a[j];

\*(p+j+1)=\*(p+j);

}

else{

break;

}

}

a[j+1]=t;

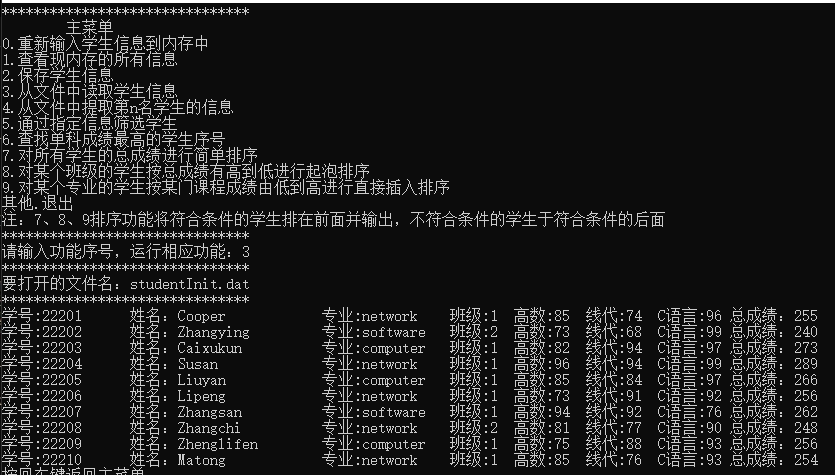
\*(p+j+1)=T;

}

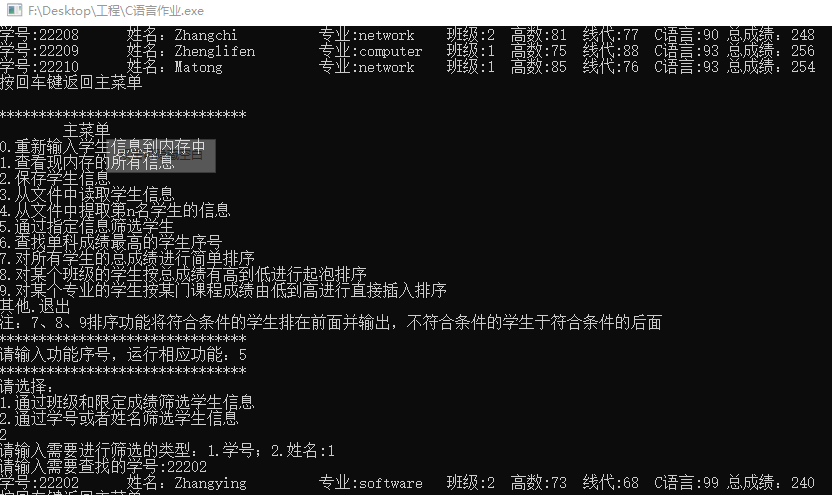
}

**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）

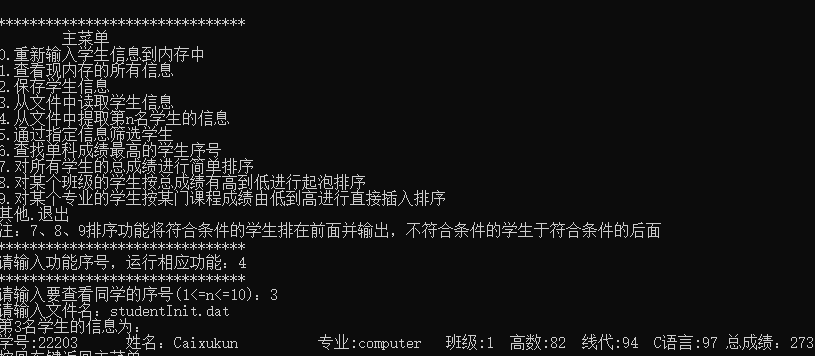
调用Input函数，并且使用Output函数进行输出



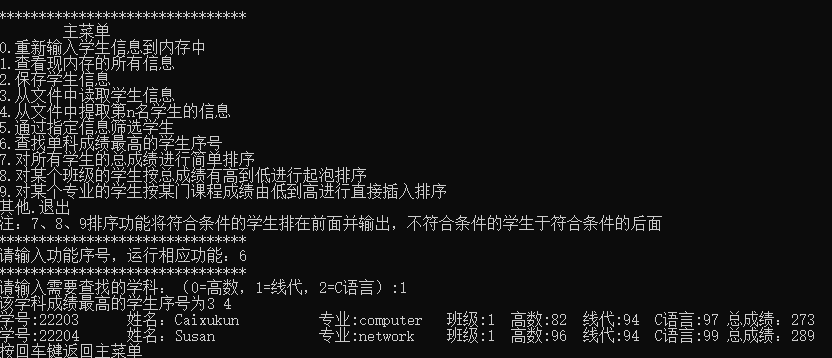
调用Search函数，查找指定学生信息（例：通过学生学号进行查找，学号：22202）



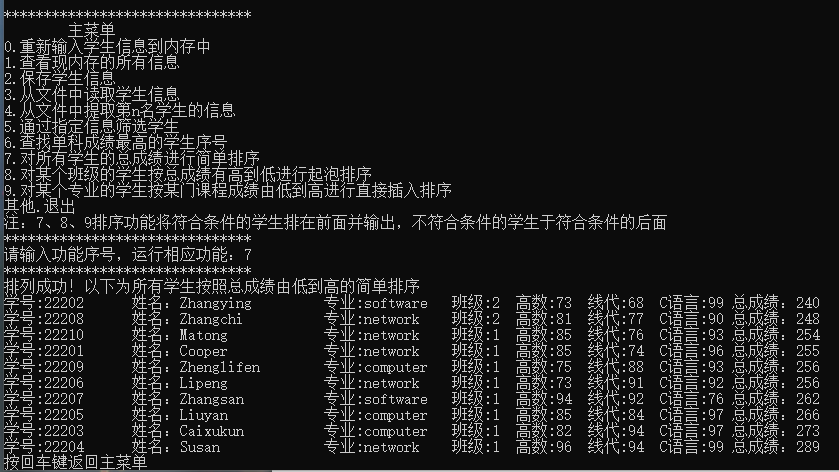
Fetch函数的调用，查询文件中第n名学生的信息（例：从studentInit.dat文件中查询第3名学生的信息）



Max函数的调用，查找单科成绩最高的学生信息（例：查找所有学生中线代分数最高的学

生序号和信息）因成绩最高的学生有两人故输出两个人的序号和信息。

Sort\_select函数调用，即简单排序，对所有学生按照总成绩由低到高进行简单排序并输出。

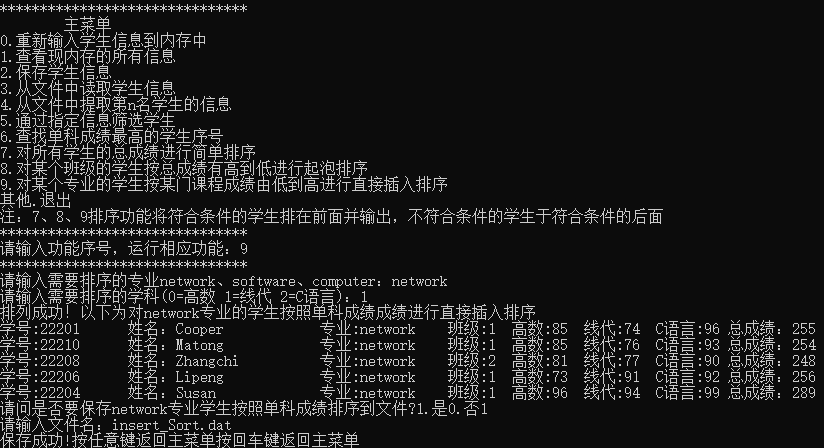


Buble\_Sort函数的调用，即起泡法排序，对某个班级的学生按总成绩有高到低进行排序。

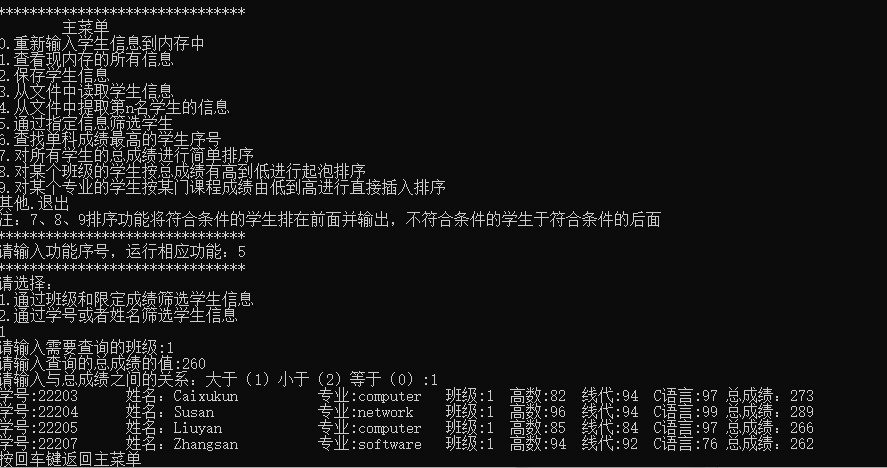
（例：network专业1班的学生进行排序）



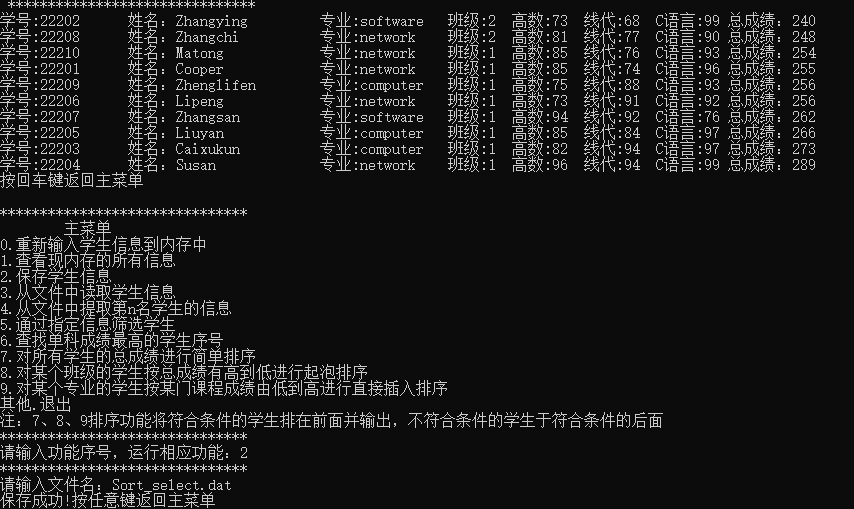
insert\_Sort函数的调用 ，即直接插入排序，对某个专业的学生按某门课程成绩由低到高进行直接插入排序（例：将network专业对线代学科成绩由低到高进行直接插入排序）最后可以进行保存操作。



通过班级和限定成绩筛选学生信息（例：筛选1班成绩高于260分的学生信息）



对先内存中的所有同学的信息进行保存



**六、实验总结**

**设计优化：**

为方便操作者的使用，若未打开文件时，进行其他的操作（除从文件中提取一个生的信息除外），会提醒操作者未打开程序，并返回主菜单。其次，为避免操作者在进 行功能 选择的时候进行错误操作，如果进行错误选择，会进行提示，并返回主菜单。在每个功能运行结束之后不会立即返回主菜单，可以给操作者以足够的时间去查看信息。因操作界面有限，将通过班级和成绩进行筛选和通过学号或者姓名进行查找进行归类。最后进行直接插入排序后可以对排序后的学生信息进行另存，方便日后的查看。

**遇到问题：**

1.因学号为字符型，无法直接用“==”进行比较，只能通过strcmp进行比较， 需要头 文件void <string>。

1. 在个别地方，进行scanf后的gets输入时会跳过，因为gets会获取scanf结束的回车键。通过添加getchar()消掉回车的影响。
2. 在输出a[i]时未定义数值的位置输出乱码，因为数组未置零导致数组的值为系统默认的值，使得输出结果错误。
3. 因为单科成绩最高的学生不止有一名，而进行比较之后只会输出序号靠前的一个人的信息。此时可以从所有学生的该科成绩中进行查找相同的成绩，同时声明一个数组，记录成绩相同的学生的位置。再将符合条件的学生进行依次输出。
4. 输出时各个学生的信息不对齐。因学生的姓名长度不同，个别学生姓名的长度超过一个制表符，在输出的时候将姓名%s变成%-15s,可以使学生姓名进行对齐，并且不会使后面的信息凌乱。