扬州大学信息工程学院

《程序设计基础II（C++）》课程设计报告

**设计题目 学生成绩管理系统**

**班 级 计算机1801**

**学 号 181604124**

**姓 名 汤礽禾**

**指导教师 严 芬**

|  |  |  |  |
| --- | --- | --- | --- |
| **《程序设计基础II（C++）》课程设计考核评估** | | | |
| 序号 | 项目名称 | 权重 | 得分 |
| 1 | 选题（复杂性） | 10% |  |
| 2 | 课设报告（规范性、完整性、合理性） | 30% |  |
| 3 | 程序实现（数据结构、编程规范、算法合理） | 30% |  |
| 4 | 演示与验收（界面友好、功能正确、极限测试） | 15% |  |
| 5 | 陈述答辩（思路清晰度、熟练度，回答正确度） | 15% |  |
|  | 课设总分 |  |  |

**验收教师签名：**

2020年 6 月

**目录**

**1.课题内容............................................2**

**（1）开发课题..............................................2**

**（2）需求..................................................2**

**（3）具体功能..............................................2**

**2.系统详细设计........................................2**

**（1）系统整体流程..........................................3**

**（2）类的设计........................................4**

**（3）函数具体功能....................................5**

1. **系统实现............................................6**

**4.课程设计小结.......................................12**

**5.附录...............................................13**

**1.课题内容：**

**（1）开发课题：**学生成绩管理系统

**（2）需求：**

学生信息管理系统，是针对学校人事处的大量业务处理工作而开发的管理软件，是典型的管理信息系统。

 它是一个教育单位不可缺少的部分，它的内容对于学校管理者来说是至关重要的，能有效的帮助学校和老师掌握学生的情况。在传统模式下利用人工进行学生信息管理，存在着较多的缺点，如：效率底，保密性差，时间一长将产生大量的文件和数据，更不便于查找，更新，维护等。诸如这些情况，令学校管理者对学生的信息管理带来了很大困难，严重影响了教育工作者的工作效率。随着科学技术的不断提高，使用日趋成熟的计算机技术来代替传统的人工模式，来实现学生信息的现代化管理，其强大的功能已为人们深刻认识，它已进入人类社会的各个领域并发挥着越来越重要的作用。作为计算机应用的一部分，使用计算机对学生信息进行管理，具有着手工管理所无法比拟的优点。例如：检索迅速、查找方便、易修改、可靠性高、存储量大、数据处理快捷、保密性好、寿命长、成本低等。这些优点能够极大地提高学生信息管理的效率，也是学校实现科学化、正规化管理的重要条件。因此，开发这样一套管理软件成为很有必要的事情。

**（3）具体功能：**

1. 从磁盘读取数据
2. 显示所有学生成绩信息
3. 添加成绩信息
4. 查询成绩信息
5. 删除成绩信息
6. 单科成绩分析
7. 修改信息
8. 将数据保存至磁盘
9. 清空所有成绩信息

**2.系统详细设计：**

系统的总体设计思想：首先设计一个学生类，然后设计一个学生管理类，作为学生类的友元类，这样便可以调用所有学生类的函数。

学生类中包含一些学生用到的基础函数，如查看单科成绩，查看个人信息等；学生管理类可以调用学生类的基础函数，以此完成查询，排序，单科分析等功能，此外还有添加，删除，保存至磁盘，从磁盘读取，清空数据等功能

**（1）系统整体流程：**

学生成绩管理系统

Read

Display

Get

Next

Get

Analyse

Swap

Get

Fortran

Get

Python

Get

Cpp

Get

Name

Get

Sno

Get

Total

添加

学

生

管

理

类

退出

修改

清空

保存

删除

排序

单科分析

显示所有学生成绩

查询

学

生

类

**（2）类的设计：**

**学生类：**

class student {

friend class studentMessage; //友元类，学生管理

private:

string sno; //学号

string name; //姓名

double score[4]; //score[0-3]记录python、c++、fortran和总分

student \* next; //链表指针，指向下一个学生类

public:

student(); //构造函数

student(string sn, string n, double sco[4]); //重载构造函数

string getName() {return name;} //返回姓名

string getSno() {return sno;} //返回学号

double getPython() {return score[0];} //返回score[0] Python成绩

double getCpp() {return score[1];} //返回score[1]C++成绩

double getFortran() {return score[2];} //返回score[3] FORTRAN成绩

double gettotal() {return score[3];} //返回总分

student\* getNext() {return next;} //返回下一个学生类的地址

static void analyse(int ,student \* ); //分析成绩

void display(); //打印成绩

void swap(); //交换对象和对象->next的信息,用于排序

};

**学生管理类：**

class studentMessage{

private:

student \* first; //头指针

student \* last; //尾指针

public:

studentMessage(); //构造函数

student\* getFirst(){return first;} //获取头指针

student\* getLast(){return last;} //获取尾指针

int getNum(){return num;} //获取表中人数

void add(); //添加

void search(); //查找

void del(); //删除

void showOne(); //分析

void showAll(); //输出所有同学所有成绩

void sort(); //排序

void read(); //从磁盘读取

void write(); //保存至磁盘

void clear(); //清空并释放空间

};

**（3）函数具体功能：**

**学生类函数：**

student::student() //构造

构造函数：每当创建一个新学生 student a时，从键盘输入学生借基础信息与成绩

student::student(string sn,string n,double sco[4]) //构造重载

重载构造函数：由于构造函数设定只能从键盘输入学生信息，局限性较大，故用此函数用于从磁盘读取学生信息，修改学生信息等。

void student::analyse(int a,student \* first)

分析函数：统计单科优（90~100），良（70~89），中（60~69），不及格（0~59）的人数，并打印学生人数和单科均分。

void student::display()

输出函数：用于输出某一学生所有成绩

void student::swap() //用于排序

交换函数：在学生管理类，交换函数中使用，交换两邻接学生类中数据。

**学生管理类函数：**

studentMessage::studentMessage()

构造函数：初始化头指针和尾指针为空

void studentMessage::add()

添加函数：向学生类链表中插入新的学生记录

void studentMessage::search()

查询函数：查询学生记录

void studentMessage::del()

删除函数：删除学生记录并释放被删除学生类的空间

void studentMessage::showOne()

分析单科函数：调用学生类中void student::analyse(int a,student \* first)函数，此函数从键盘输入需要分析的科目代号

void studentMessage::showAll()

总输出函数：调用学生类中void student::display()函数，输出所有学生记录

void studentMessage::sort()

排序函数：调用学生类中void student::swap()函数，实现单科或总分从高到低排序

void studentMessage::update()

更新函数：更新学生记录，若学号相同禁止更新。

void studentMessage::read()

读取函数：从磁盘文件读取学生记录

void studentMessage::write()

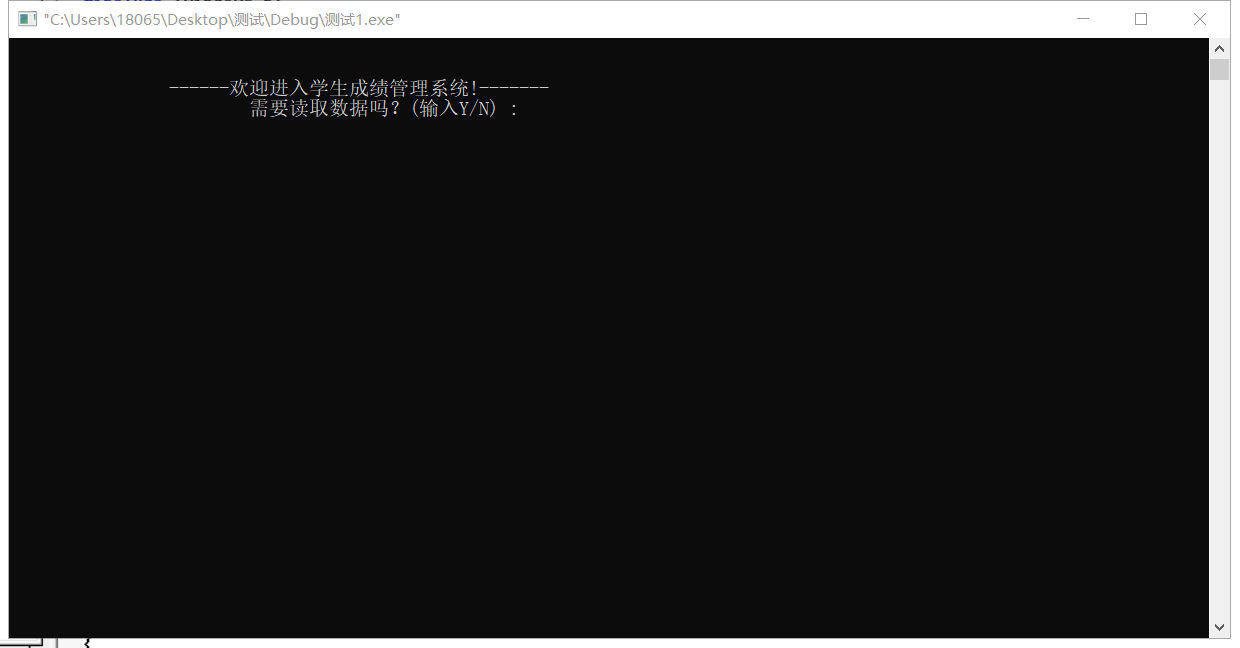
保存函数：将学生类链表写入磁盘文件

void studentMessage::clear()

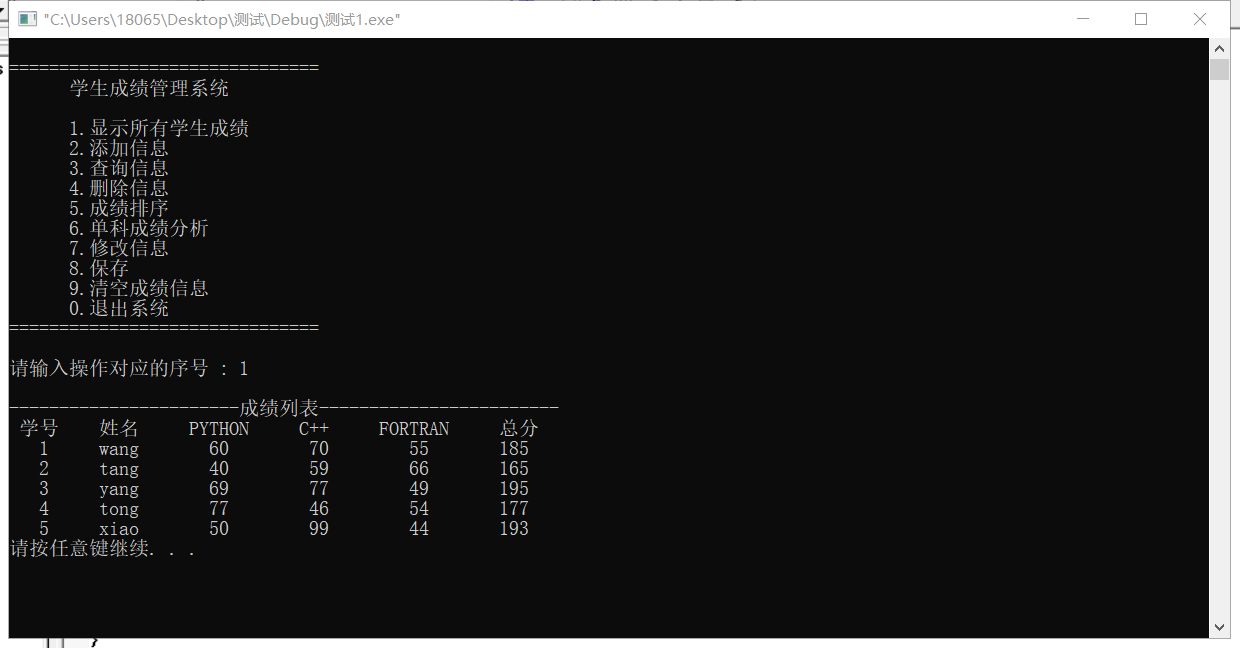
清空函数：清空所有学生类链表记录并释放空间

**3.系统实现：**

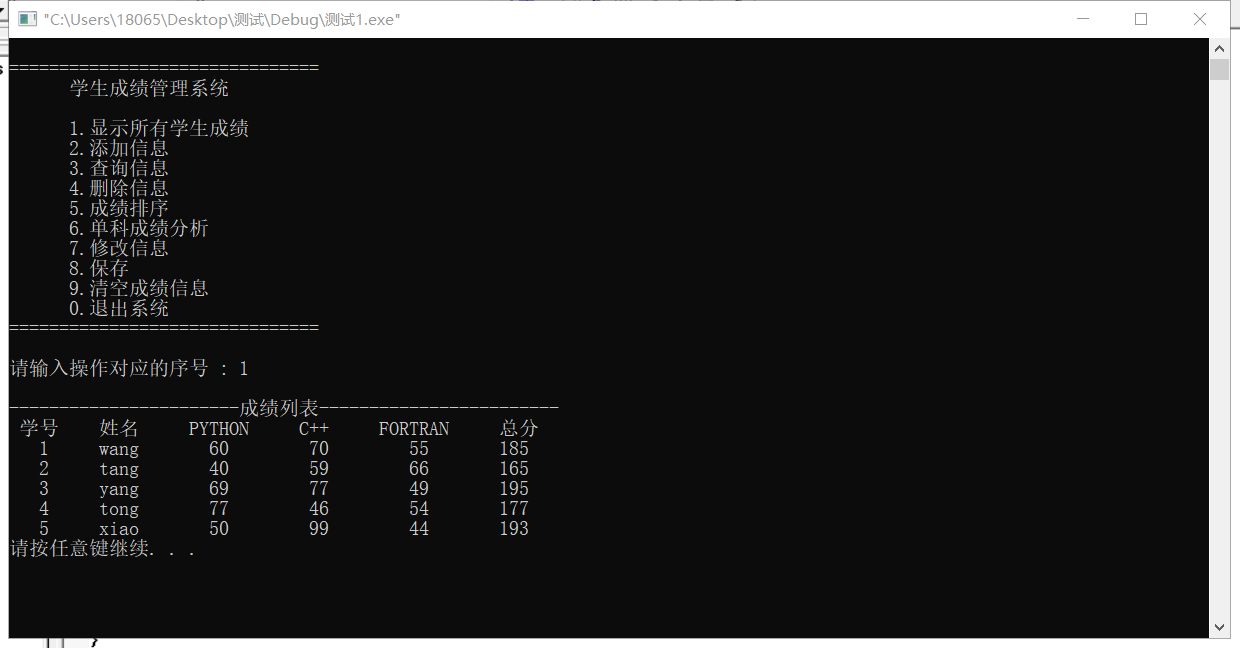
**进入界面**：选择是否从磁盘文件读取数据，这里选择的是Y（大小写皆可）



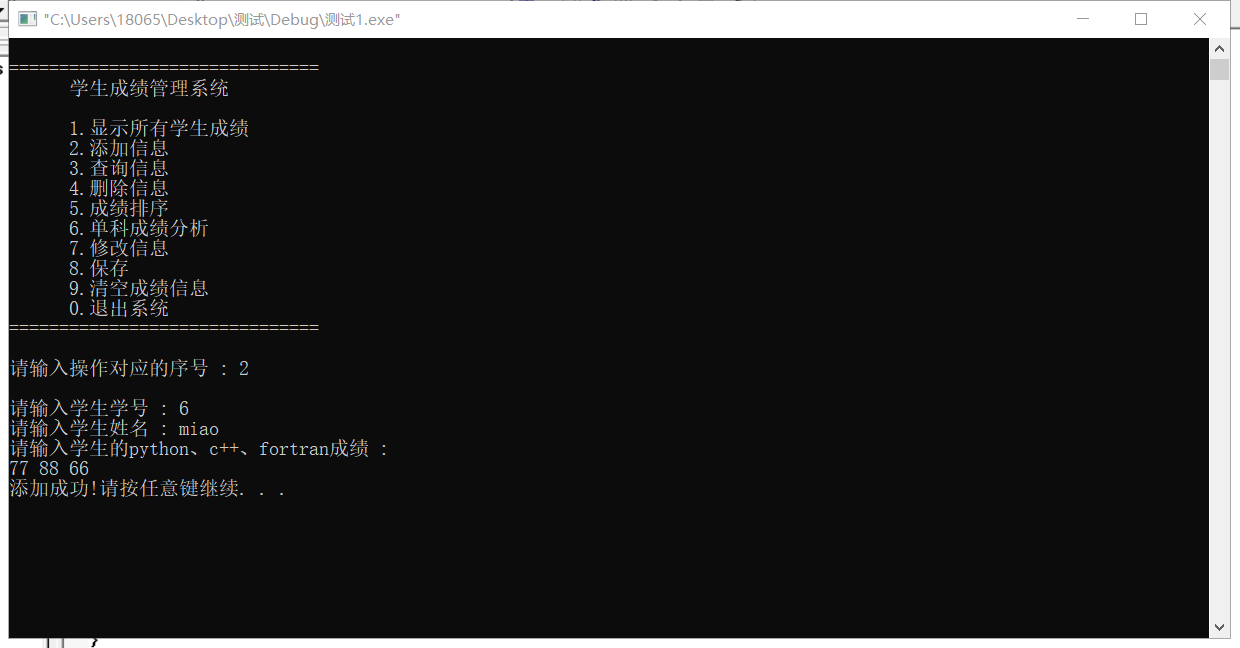
**菜单界面：**



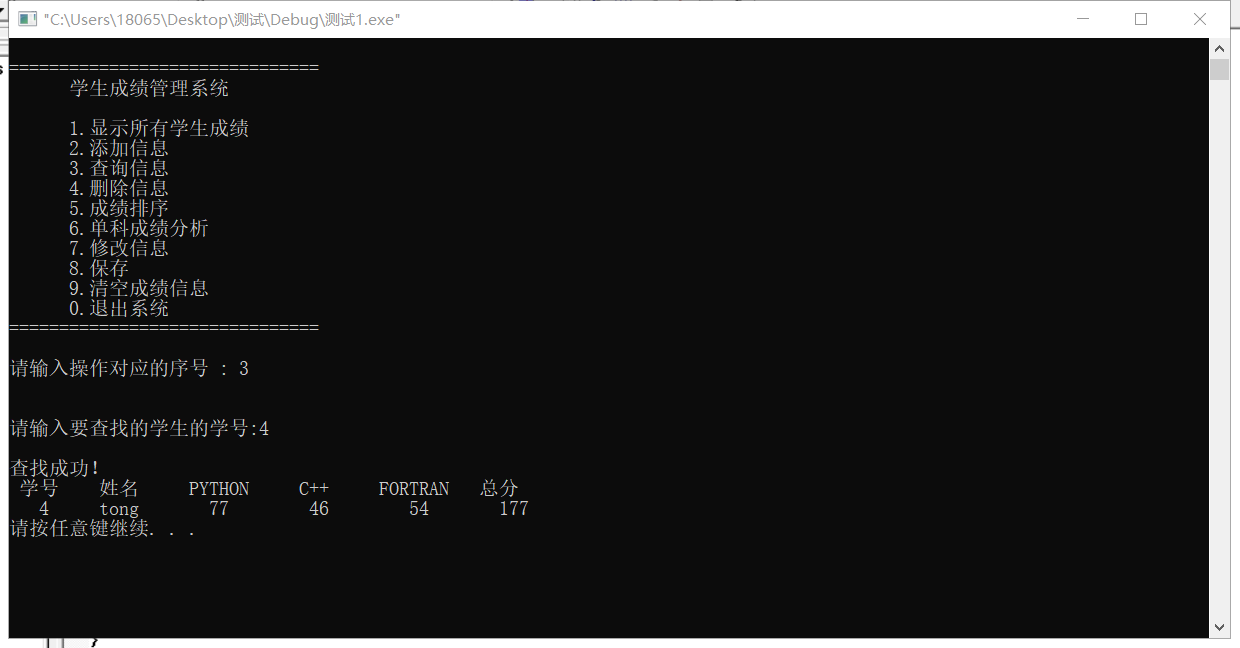
**选项1.显示所有学生成绩：**



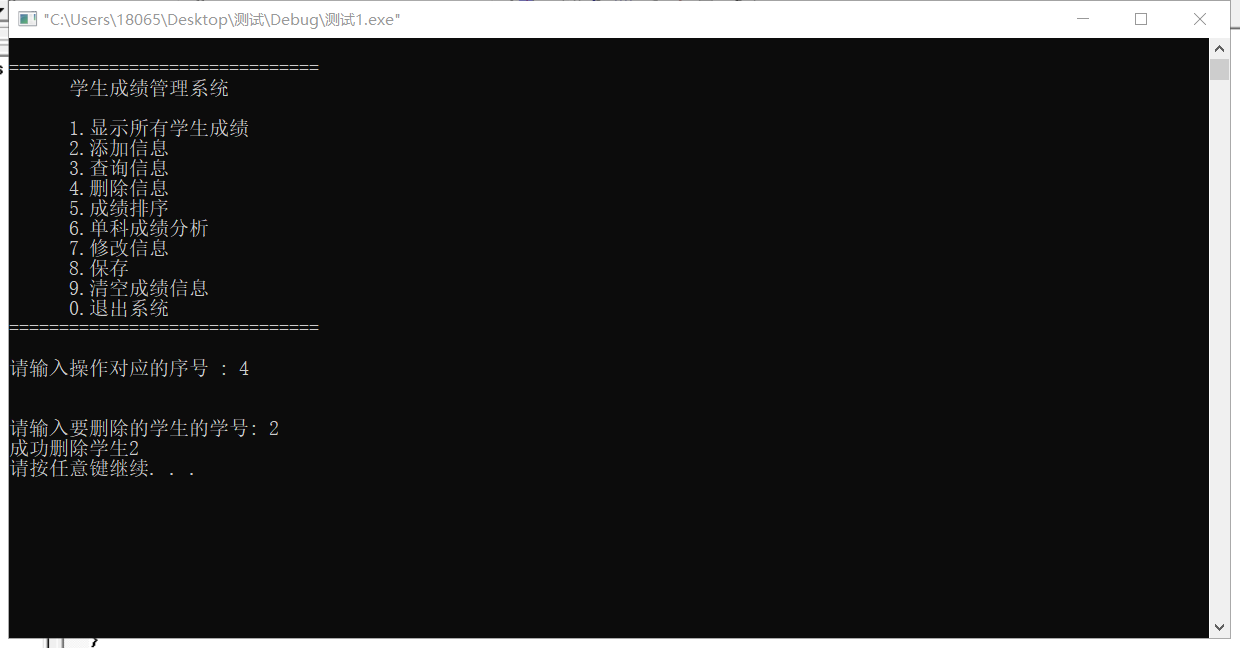
**选项2.添加信息（无重复学号）：**



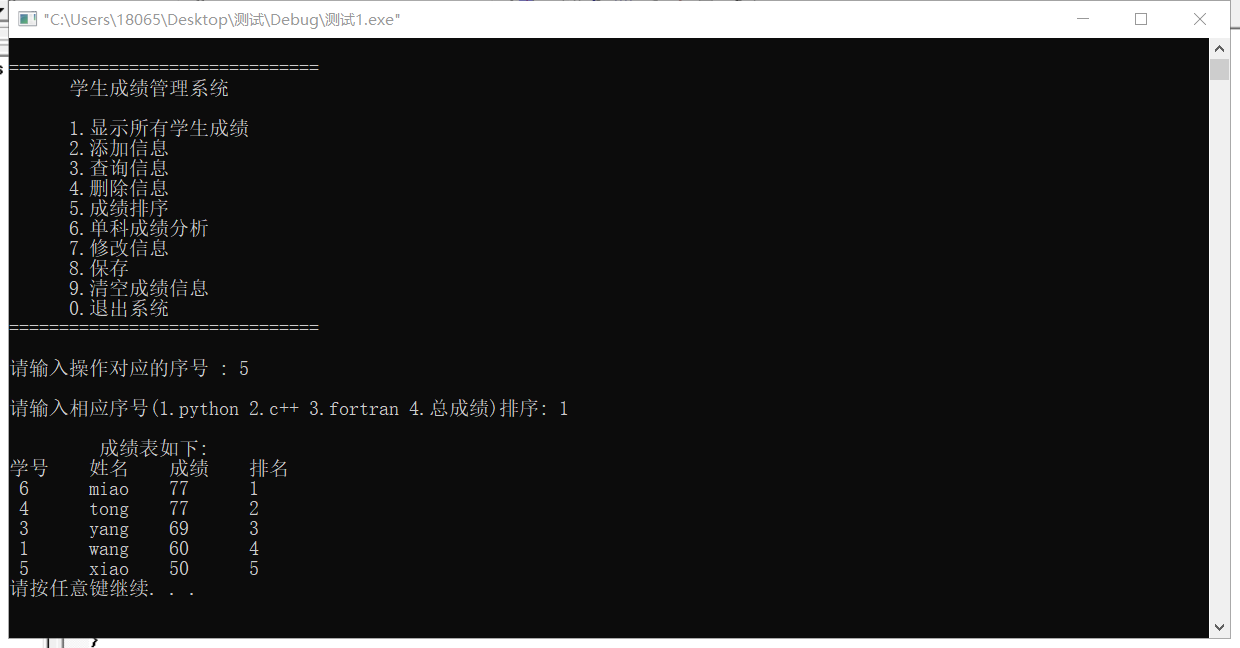
**选项3.查询：**



**选项4.删除：**

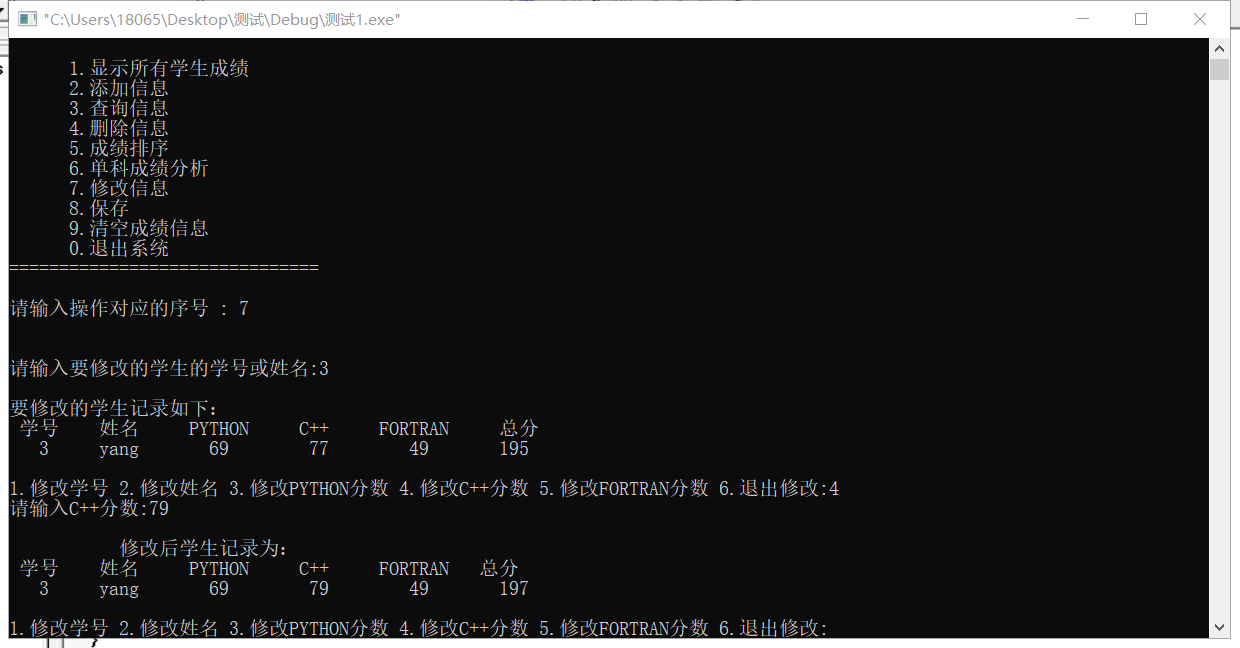


**选项5.成绩排序**

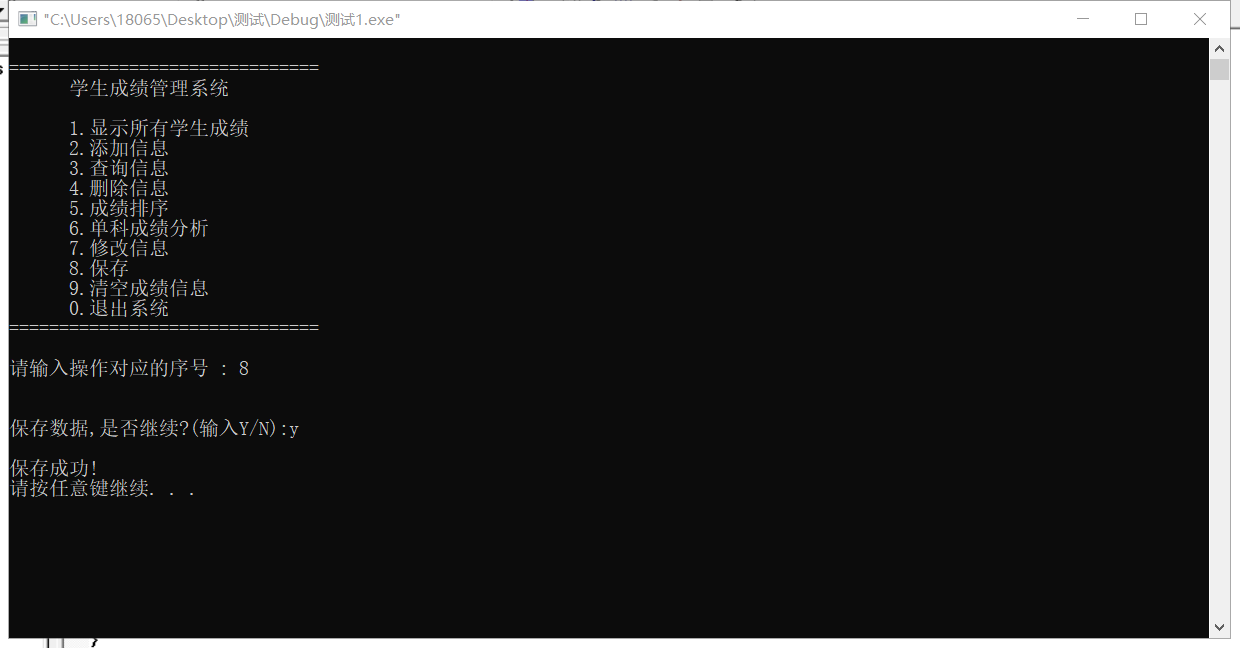


**选项6.成绩分析：**  

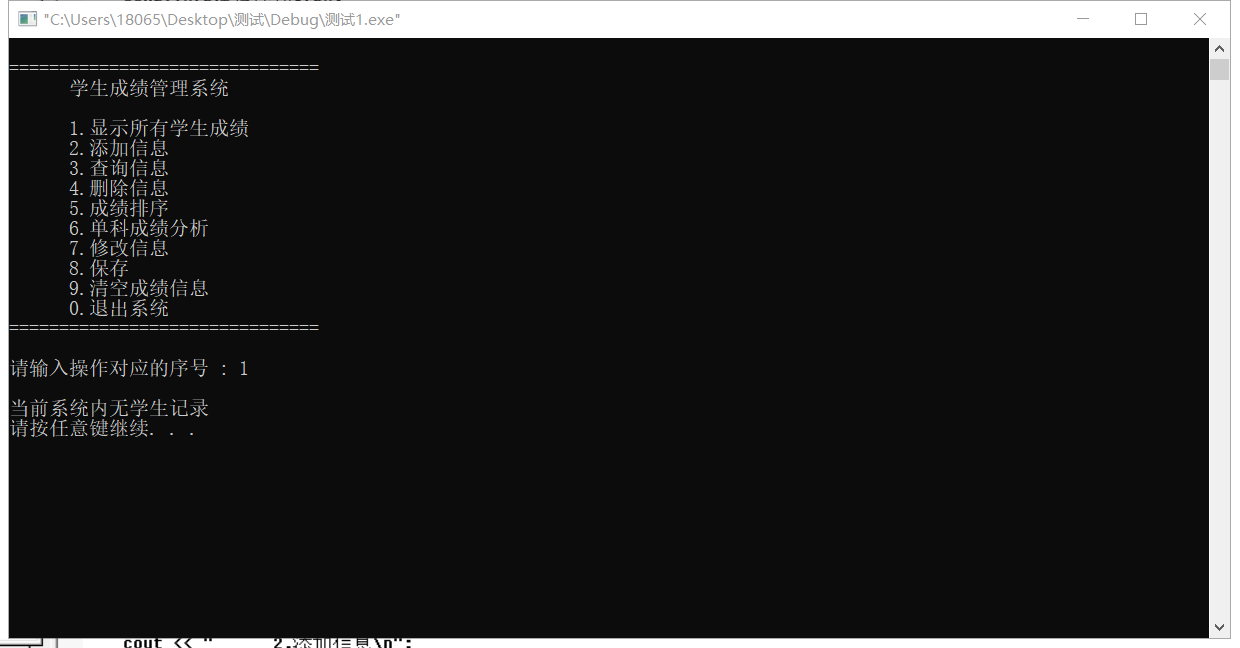
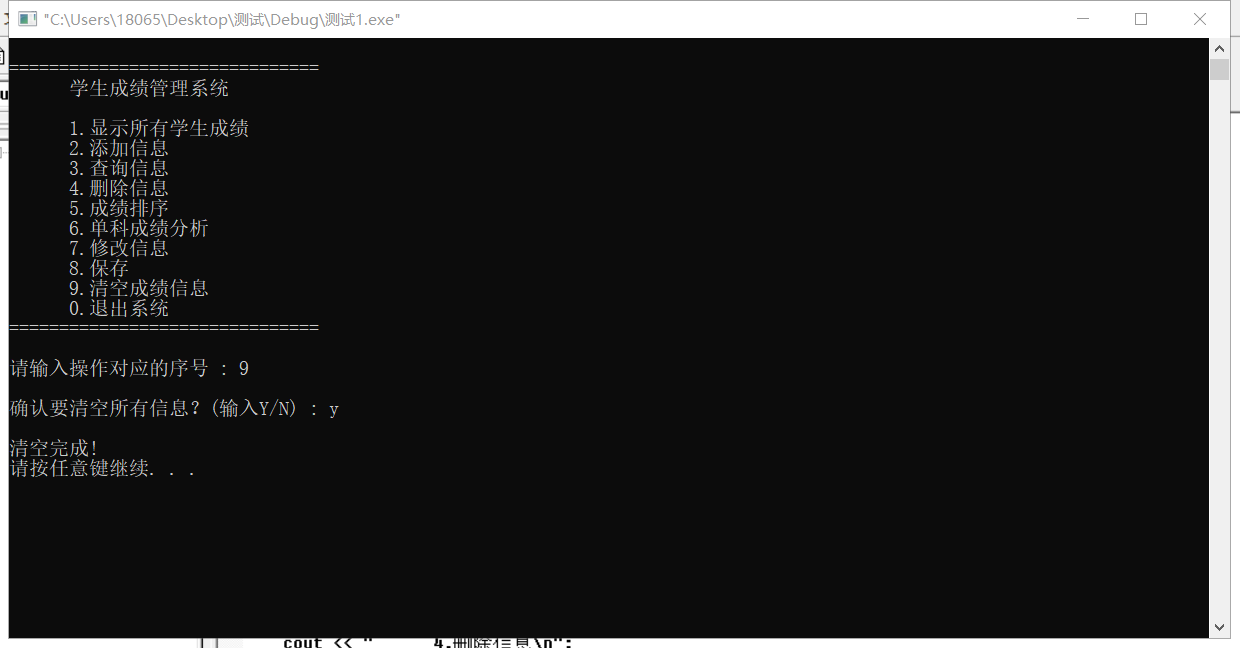

**选项7.修改信息（无重复学号）：**



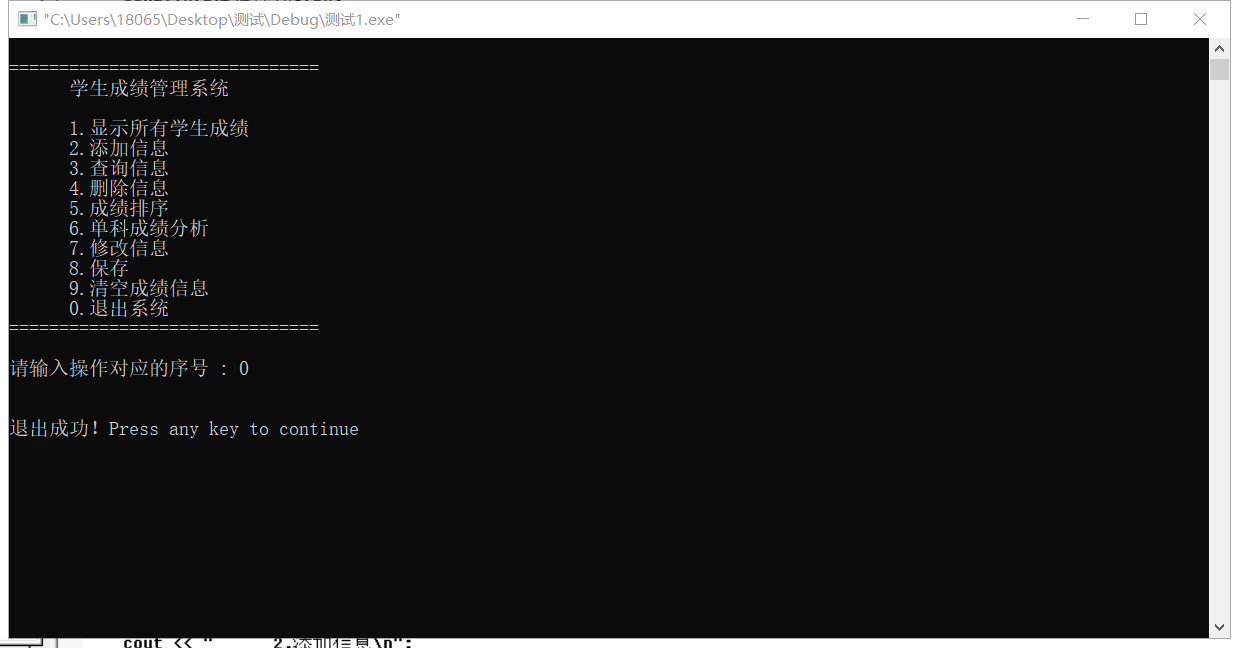
**选项8.保存**



**选项9.清空**



**选项0.退出**



**4.课程设计小结：**

此次课程设计加深了我对友元类的理解，熟悉了友元类的使用，友元类是非常方便的，可以直接访问另一个类中的私有成员和受保护成员，提高效率，方便编程。但从网上查阅的信息来看，友元类破坏了类的封装性，提供了安全漏洞。由于我现阶段只在扫盲，对于友元类的缺点理解不够深刻，希望日后能加深友元类缺点的理解。

其次，此次课设代码设计有很多冗余，由于一开始参考了一些csdn的文章，我将学生类的构造函数定义为直接从键盘键入数据，导致后面的从磁盘读取，修改等功能无法实现，只能先删除，再重新输入正确的数据，局限性非常大，于是我增加了一个重载构造函数，解决了磁盘读取，数据修改等功能缺失。

**4.附录：**

#include<iostream>

#include <stdlib.h>

#include <windows.h>

#include<fstream>

#include<string>

#include<iomanip>

using namespace std;

int num; //总学生人数

class student {

friend class studentMessage;

private:

string sno;

string name;

double score[4]; //score[0-3]记录python、c++、fortran和总分

student \* next;

public:

student();

student(string sn, string n, double sco[4]);

string getName() {return name;}

string getSno() {return sno;}

double getPython() {return score[0];}

double getCpp() {return score[1];}

double getFortran() {return score[2];}

double gettotal() {return score[3];}

student\* getNext() {return next;}

static void analyse(int ,student \* ); //分析成绩

void display(); //打印成绩

void swap(); //交换对象和对象->next的信息,用于排序

};

student::student() //构造

{

cout<<"请输入学生学号 : ";cin>>sno;

cout<<"请输入学生姓名 : ";cin>>name;

cout<<"请输入学生的python、c++、fortran成绩 : "<<endl;

int a,b,c;

cin>>a;

while(a<0||a>100)

{

cout<<"PYTHON成绩输入有误，请重新输入: "<<endl;

cin>>a;

}

cin>>b;

while(b<0||b>100)

{

cout<<"C++输入成绩输入有误，请重新输入: "<<endl;

cin>>b;

}

cin>>c;

while(c<0||c>100)

{

cout<<"FORTRAN成绩输入有误，请重新输入: "<<endl;

cin>>c;

}

score[0]=a;

score[1]=b;

score[2]=c;

score[3]=a+b+c;

next=NULL;

}

student::student(string sn,string n,double sco[4]) //构造重载

{ //这个函数本没必要，因为一开始没考虑保存功能，上头那个构造函数只能从键盘输入，

//故不能从文件写入，也不能修改，故此重载构造函数诞生了

sno = sn;

name = n;

for(int i = 0; i <= 3; i++)

{

score[i] = sco[i];

}

next=NULL;

}

void student::analyse(int a,student \* first)

{

int x;

if(a==1) x = 0;

else if(a ==2) x = 1;

else x = 2;

int you=0,liang=0,zhong=0,bujige=0;

student \* t=first;

double average=0; //平均分

while(t)

{

int b = int(t->score[x]/10);

switch(b)

{

case 10:

case 9:you++; break;

case 8:

case 7:liang++; break;

case 6:zhong++; break;

default:bujige++;

}

average+=t->score[x];

t=t->next;

}

average=average/num;

cout<<" 考生总人数 : "<<num<<" 平均分 : "<<average<<endl;

cout<<"\n 优 良 中 不及格 "<<endl;

cout<<" "<<you<<" \t"<<liang<<" \t"<<zhong<<"\t"<<bujige<<endl;

}

void student::display()

{

cout<<" "<<setw(3)<<sno<<" "<<setw(3)<<name<<" ";

for (int i=0;i<=3;i++)

{

cout<<setw(3)<<score[i]<<" ";

}

cout<<endl;

}

void student::swap() //用于排序

{

int j;

string na,id;

double sco[4];

na=name;name=next->name;next->name=na;

id=sno;sno=next->sno;next->sno=id;

for(j=0;j<=3;j++)

{

sco[j]=score[j];

score[j]=next->score[j];

next->score[j]=sco[j];

}

}

class studentMessage{

private:

student \* first;

student \* last;

public:

studentMessage();

student\* getFirst(){return first;}

student\* getLast(){return last;}

int getNum(){return num;}

void add(); //添加

void search(); //查找

void del(); //删除

void update();

void showOne(); //分析

void showAll(); //输出所有同学所有成绩

void sort(); //排序

void read(); //读

void write(); //写

void clear(); //清空并退出

};

studentMessage::studentMessage()

{

first=NULL; //头结点

last=NULL; //尾结点

num=0;

}

void studentMessage::add()

{

student \* t =new student;

student \*p = first;

while(p)

{

if(p->sno==t->sno)

{

cout<<"\n学号输入错误或该学生成绩已经存在！"<<endl;

Sleep(2000);

system("cls");

return;

}

p=p->next;

}

num++;

if(first==NULL)

{

first=last=t;

}

else

{

last->next=t;

last=last->next;

}

cout<<"添加成功!";

system("pause");system("cls");

}

void studentMessage::search()

{

string a;

cout<<"\n请输入要查找的学生的学号:";cin>>a;

student \*t = first;

while(t)

{

if(t->sno==a)

break;

t=t->next;

}

if(!t)

{

cout<<"\n未找到要查找学生！"<<endl;

system("pause");system("cls");

return;

}

cout<<"\n查找成功！"<<endl;

cout << " 学号 姓名 PYTHON C++ FORTRAN 总分" << endl;

t->display();

system("pause");system("cls");

}

void studentMessage::del()

{

string a;

cout<<"\n请输入要删除的学生的学号: ";cin>>a;

student \*t = first;

student \*p=NULL;

while(t){

if(t->sno==a)break;

p=t;

t=t->next;

}

if(!t)

{

cout<<"\n未找到要删除学生！"<<endl;

system("pause");system("cls");

return;

}

if(!p)

{

first=first->next;

cout<<"\n成功删除学生"<<a<<endl;

delete t;

}

else

{

p->next=t->next;

cout<<"成功删除学生"<<a<<endl;

delete t;

}

num--;

system("pause");system("cls");

}

void studentMessage::showOne()

{

int a;

while(1)

{

cout<<"\n想要分析哪一门成绩？请输入学科序号(1:PYTHON 2:C++ 3:FORTRAN): ";cin>>a;

if(a!=1 && a!=2 && a!=3)

cout<<"\n输入序号或名称有误，请重新输入！"<<endl;

else break;

}

cout<<"\n\t成绩分析如下: \n"<<endl;

student::analyse(a,first);

system("pause");system("cls");

}

void studentMessage::showAll()

{

if(!first)

{

cout<<"当前系统内无学生记录"<<endl;

system("pause");

system("cls");

return;

}

cout << "-----------------------成绩列表------------------------"<< endl;

cout << " 学号 姓名 PYTHON C++ FORTRAN 总分" << endl;

student \*t = first;

while(t){

t->display();

t=t->next;

}

system("pause");

system("cls");

}

void studentMessage::sort()

{

int a,n=0; //n--排名

cout<<"请输入相应序号(1.python 2.c++ 3.fortran 4.总成绩)排序: ";

while(1)

{

cin>>a;

if (a!=1&&a!=2&&a!=3&&a!=4)

cout<<"\n输入序号有误，请重新输入 : ";

else break;

}

student \*t=first;

student \*p=t;

/\*student \*tail=last;

for(;t->next!=NULL;t=t->next)

for(student \* q=t->next;q!=NULL;q=q->next)

if(t->next->score[a-1]>=t->score[a-1])

{

t->swap();

}\*/

while(p->next)

{

t=first;

while(t->next)

{

if(t->next->score[a-1]>=t->score[a-1])

{

t->swap();

}

t=t->next;

}

p=p->next;

}

t = first;

cout<<"\n\t 成绩表如下: "<<endl;

cout<<"学号 姓名 成绩 排名"<<endl;

while(t)

{

n++;

cout<<" "<<t->getSno()<<" \t"<<t->getName()<<" \t";

switch(a)

{

case 1 :cout<<t->getPython()<<" \t"<<n<<endl;break;

case 2 :cout<<t->getCpp()<<" \t"<<n<<endl;break;

case 3 :cout<<t->getFortran()<<" \t"<<n<<endl;break;

case 4 :cout<<t->gettotal()<<" \t"<<n<<endl;break;

}

t=t->next;

}

system("pause");system("cls");

}

//

void studentMessage::update()

{

string a;

cout<<"\n请输入要修改的学生的学号或姓名:";cin>>a;

student \*t = first;

while(t)

{

if(t->sno==a || t->getName()==a) break;

t=t->next;

}

if(!t)

{

cout<<"\n未找到要修改学生！"<<endl;

return;

}

cout<<"\n要修改的学生记录如下："<<endl;

cout << " 学号 姓名 PYTHON C++ FORTRAN 总分" << endl;

t->display();

int x=0,flag=0;

string na;

student \*m;

while(x!=6)

{

cout<<"\n1.修改学号 2.修改姓名 3.修改PYTHON分数 4.修改C++分数 5.修改FORTRAN分数 6.退出修改:"; cin>>x;

switch(x)

{

case 1:

{

cout<<"请输入学号:";

cin>>na;

m=first;

while(m)

{

if(m->sno==na)

{

cout<<"\n学号输入错误或该学生成绩已经存在!"<<endl;flag=1;

break;

}

m=m->next;

}

if(!flag) t->sno=na;

break;

}

case 2:cout<<"请输入姓名:"; cin>>t->name;break;

case 3:cout<<"请输入PYTHON分数:"; cin>>t->score[0];t->score[3]=t->score[1]+t->score[0]+t->score[2];break;

case 4:cout<<"请输入C++分数:"; cin>>t->score[1];t->score[3]=t->score[1]+t->score[0]+t->score[2];break;

case 5:cout<<"请输入FORTRAN分数:"; cin>>t->score[2];t->score[3]=t->score[1]+t->score[0]+t->score[2];break;

case 6:break;

default:cout<<"输入错误，请重新选择修改项"<<endl;

}

cout<<"\n\t 修改后学生记录为：\n";

cout << " 学号 姓名 PYTHON C++ FORTRAN 总分" << endl;

t->display();

}

system("pause");

system("cls");

}

void studentMessage::read()

{

ifstream fs;

student \*s=first;

string sno,name;

double score[4];

fs.open("Student.dat",ios::in|ios::binary );

fs>>sno>>name>>score[0]>>score[1]>>score[2]>>score[3];

while(fs.good())

{

s= new student(sno,name,score);

student \*s2 = first;

if(first) //若已经存在结点

{

student \*s2 = first;

while(s2->next) //查找尾结点

{

s2=s2->next;

}

s2->next=s; //连接

}

else //若不存在结点(表空)

{

first=s; //连接

}

fs>>sno>>name>>score[0]>>score[1]>>score[2]>>score[3];

num++;

last=s;

}

fs.close();

cout<<"\n读取成功!\n";

}

void studentMessage::write()

{

ofstream fs("Student.dat",ios::out|ios::binary );

char c;

cout<<"\n保存数据,是否继续?(输入Y/N):";

cin>>c;

if(toupper(c)!='Y')

return;

student \*s=first;

while(s)

{

fs<<s->sno<<" "<<s->name<<" "<<s->score[0]<<" "<<s->score[1]<<" "<<s->score[2]<<" "<<s->score[3]<<endl;

s=s->next;

}

fs.close();

cout<<"\n保存成功!\n";

system("pause");

system("cls");

}

void studentMessage::clear()

{

char x;

cout<<"确认要清空所有信息？(输入Y/N) : ";cin>>x;

if (toupper(x)!='Y')

{

cout<<"\n取消清空\n"<<endl;

return;

}

student \*t = first;

student \*p;

while(t){

p=t;

t=t->next;

delete p;

}

last=first=NULL;

cout<<"\n清空完成!\n";

system("pause");system("cls");

}

void showMenu()

{

cout << " \n";

cout << "===============================\n";

cout << " 学生成绩管理系统\n\n";

cout << " 1.显示所有学生成绩\n";

cout << " 2.添加信息\n";

cout << " 3.查询信息\n";

cout << " 4.删除信息\n";

cout << " 5.成绩排序\n";

cout << " 6.单科成绩分析\n";

cout << " 7.修改信息\n";

cout << " 8.保存\n";

cout << " 9.清空成绩信息\n";

cout << " 0.退出系统\n";

cout << "===============================\n";

cout << " \n";

}

int main()

{

char x;

int h;

studentMessage stuM=studentMessage();

cout<<"\n\n\t\t------欢迎进入学生成绩管理系统!-------"<<endl;

cout<<"\t\t\t需要读取数据吗？(输入Y/N) : ";cin>>x;

if (toupper(x)=='Y')

{

stuM.read();

}

system("pause");

system("cls");

while(1)

{

showMenu();

cout << "请输入操作对应的序号 : ";

cin >>h;

cout<<endl;

switch(h)

{

case 1: stuM.showAll(); break;

case 2: stuM.add(); break;

case 3: stuM.search(); break;

case 4: stuM.del(); break;

case 5: stuM.sort();break;

case 6: stuM.showOne(); break;

case 7: stuM.update();break;

case 8: stuM.write();break;

case 9: stuM.clear();break;

case 0: cout<<"\n退出成功！";return 0;

default:cout<<"\n输入序号有误！请输入正确的序号。"<<endl;

}

}

}