**桂林电子科技大学2018-2019学年 第2学期**

**程序设计与问题求解 实验报告**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 实验名称 | **实验十 综合程序设计** | | | | | | | |  | 辅导教师意见：  成绩 教师签名： |
| 院 系 | **计算机与信息安全学院** | | | 专业 | | **计算机类** | | |
| 学 号 | **1800300938** | | | 姓名 | | **韦森强** | | |
| 实验日期 | **2019** | 年 |  | | 月 | |  | 日 |
|  |  | | | | | | | |

**一．实验目的**

1．掌握指针与内存地址的关系

2．掌握通过指针动态申请和释放内存的编程方法

3．学习和掌握单向链表的基本操作

|  |
| --- |
| **二、实验内容和步骤** |

**１.　设计和完成：简单学生成绩管理信息系统**

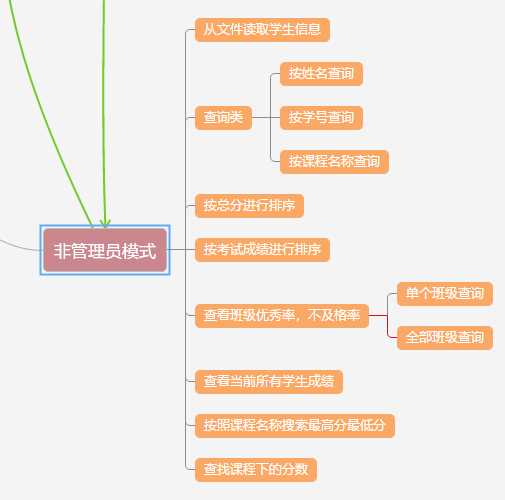
学生成绩信息包括：学号，姓名，课程名，平时成绩，实验成绩，考试成绩，总评成绩。

**２.　要求实现如下功能：**

1. 能够实现学生成绩信息的插入、删除和修改；
2. 能够实现各种查询（分别根据学生学号、姓名、课程名称等）；
3. 能够实现按照考试成绩、总评成绩进行排序；
4. 能够查询某门课程的最高分、最低分并输出相应学生信息；
5. 能够查询某门课程的优秀率（90 分及以上）、不及格率；

**三 .需求分析**



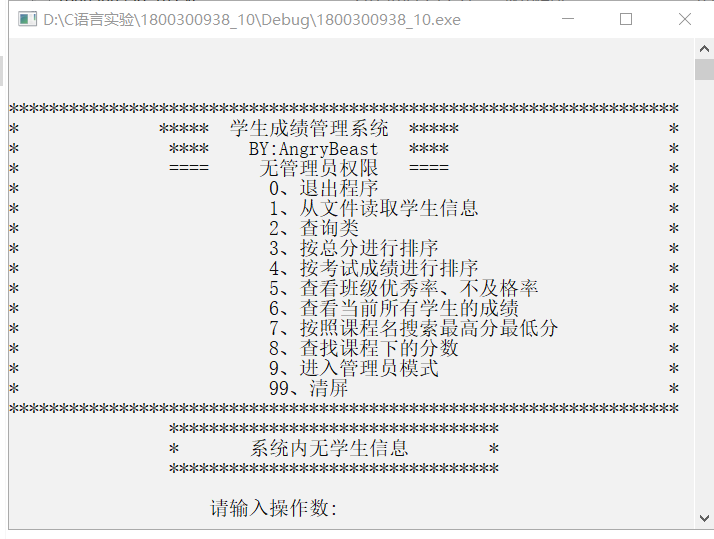


**运行结果截图**

管理员模式主菜单



非管理员模式主菜单



源代码

//学生管理系统 BY:AngryBeast 2019.6

#define \_CRT\_SECURE\_NO\_DEPRECATE //与fopen\_s,fscanf\_s相关

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int admin; //admin模式 1开启 0关闭

int lenth; //结构体长度

struct Student\* pHead; //全局头指针

void FunctionList(); //打印功能列表函数

void Admin\_FunctionList(); //admin功能列表

void Login(); //登陆admin

void AdminAccount(char Account[], char Password[]); //Admin账号读取

void TravelList(struct Student\* pHead); //遍历链表

void Admin\_Choose(); //管理员选择功能

void Singe\_print(struct Student\* p); //单个打印学生信息

void FreeAll(struct Student\* p); //删除所有

void Add\_stu(struct Student\* pHead); //增加学生

struct Student Search\_name(struct Student\* p); //搜索姓名

void Search\_num(struct Student\* p); //搜索学号

void Search\_classname(struct Student\* p); //搜索课程名称

void Excellent\_rate(struct Student\* pHead); //优秀率、不及格率

struct Student FileR(struct Student\* pHead); //文件读取

struct Student Cheak(struct Student\* p); //检查是否能申请到内存

void Search\_classname\_LandH(struct Student\* p); //搜索课程最高分最低分

void Score\_sort(struct Student\* pHead); //总分排序

void Same\_class\_score(struct Student\* p); //同课程同分

void Modify\_information(struct Student\* p); //修改学生信息

void Delete\_stu(struct Student\* p); //删除学生信息

void FileW(struct Student\* pHead); //写入文件

void Exam\_sort(struct Student\* pHead); //按考试成绩进行排序

void Choose(); //选择功能

void Delete\_data(); //删除数据文件

struct Student

{

char name[16]; //姓名

char lessonname[16]; //课程名称

int classroom; //班级

int number; //学号

int normal; //平时成绩

int exam; //考试成绩

int score; //总分

struct Student\* next;

};

int main()

{

admin = 1;

lenth = sizeof(Student);

system("color f0"); //调整背景和字体颜色

pHead = (struct Student\*)malloc(lenth);

Cheak(pHead);

pHead->next = (struct Student\*)NULL;

while (1)

{

while (!admin) //非管理员模式

{

system("color f0");

FunctionList(); //打印功能列表

Choose(); //输入操作数

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

}

while (admin) //有管理员权限时

{

system("color f5");

Admin\_FunctionList(); //打印功能列表

Admin\_Choose(); //输入操作数

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

}

}

}

void FunctionList() //打印功能列表函数

{

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

printf("\* \*\*\*\*\* 学生成绩管理系统 \*\*\*\*\* \*\n");

printf("\* \*\*\*\* BY:AngryBeast \*\*\*\* \*\n");

printf("\* ==== 无管理员权限 ==== \*\n");

printf("\* 0、退出程序 \*\n");

printf("\* 1、从文件读取学生信息 \*\n");

printf("\* 2、查询类 \*\n");

printf("\* 3、按总分进行排序 \*\n");

printf("\* 4、按考试成绩进行排序 \*\n");

printf("\* 5、查看班级优秀率、不及格率 \*\n");

printf("\* 6、查看当前所有学生的成绩 \*\n");

printf("\* 7、按照课程名搜索最高分最低分 \*\n");

printf("\* 8、查找课程下的分数 \*\n");

printf("\* 9、进入管理员模式 \*\n");

printf("\* 99、清屏 \*\n");

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

}

void Admin\_FunctionList()

{

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

printf("\* \*\*\*\*\* 学生成绩管理系统 \*\*\*\*\* \*\n");

printf("\* \*\*\*\* BY:AngryBeast \*\*\*\* \*\n");

printf("\* ======= Admin mode ======== \*\n");

printf("\* 0、退出程序 \*\n");

printf("\* 1、从文件读取学生信息 \*\n");

printf("\* 2、查询类 \*\n");

printf("\* 3、按总分进行排序 \*\n");

printf("\* 4、按考试成绩进行排序 \*\n");

printf("\* 5、查看班级优秀率、不及格率 \*\n");

printf("\* 6、查看当前所有学生的成绩 \*\n");

printf("\* 7、按照课程名搜索最高分最低分 \*\n");

printf("\* 8、查找课程下的分数 \*\n");

printf("\* 9、加入学生信息 \*\n");

printf("\* 10、从当前系统删除学生信息 \*\n");

printf("\* 11、修改学生信息 \*\n");

printf("\* 12、写入文件 \*\n");

printf("\* 13、退出管理员模式 \*\n");

printf("\* 99、清屏 \*\n");

printf("\* 998、删除数据文件 \*\n");

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

}

void Login()

{

char account[16] = { "AngryBeast" }; //管理员账户

char password[16] = { "Admin111" }; //管理员密码

char buffer\_Account[16] = { 0 };

char buffer\_Password[16] = { 0 };

int flag1, flag2, i;

flag1 = flag2 = -1;

i = 0;

printf\_s("输入账号:");

rewind(stdin);

gets\_s(buffer\_Account);

flag1 = strcmp(buffer\_Account, account);

printf\_s("输入密码:");

rewind(stdin); //清空缓冲区

gets\_s(buffer\_Password);

flag2 = strcmp(buffer\_Password, password);

if (flag1 == 0 || flag2 == 0) //检测是否是超级管理员

{

admin = 1;

return;

}

AdminAccount(buffer\_Account, buffer\_Password); //检测是否是管理员

if (admin == 0)

printf\_s("用户名或密码输入错误\n");

return;

}

struct Student Cheak(struct Student\* p)

{

if (p == NULL)

{

printf\_s("error 无法申请到内存\n");

system("pause");

exit(-1);

}

}

struct Student FileR(struct Student \*pHead)

{

int temp\_normal,temp\_exam,temp\_score,i;

char buffer[16]; //缓冲区

FILE\* fpRead = NULL;

struct Student\* p, \* tail;

errno\_t err; //与fopen相关

if ((err = fopen\_s(&fpRead, "Student.txt", "r")) != NULL) //成功返回0，不成功返回非0

{

printf("未找到Student.txt文件\n");

system("pause");

exit(-1);

}

while (pHead->next) //遍历直到链表的末尾

{

pHead = pHead->next;

}

tail = pHead; //记录尾指针

while (!feof(fpRead))

{

p = (struct Student\*)malloc(lenth);

Cheak(p);

p->next = NULL;

fgets(buffer, sizeof(buffer), fpRead); //读取学生名字

buffer[strlen(buffer) - 1] = 0; //消除换行符

strcpy(p->name, buffer);

fgets(buffer, sizeof(buffer), fpRead); //读取课程名

buffer[strlen(buffer) - 1] = 0;

strcpy(p->lessonname, buffer);

fscanf(fpRead, "%d", &p->classroom); //读取班级

fscanf(fpRead, "%d", &p->number); //读取学号

fscanf(fpRead, "%d", &temp\_normal); //将平时分取出

fscanf(fpRead, "%d", &temp\_exam); //将考试分数取出

fgets(buffer, sizeof(buffer), fpRead); //清除回车

temp\_score = (int)((0.20 \* temp\_normal) + (0.80 \* temp\_exam) + 0.5); //计算并四舍五入

p->normal = temp\_normal;

p->exam = temp\_exam;

p->score = temp\_score;

tail->next = p; //创建链表

tail = p;

}

fclose(fpRead);

printf("成功读取文件\n");

system("pause");

return \*pHead;

}

void TravelList(struct Student\* pHead) //遍历链表

{

struct Student \* p;

int count = 0;

p = pHead->next;

printf("学生名字 ");

printf("课程名称 ");

printf("班级 ");

printf("学号 ");

printf("平时成绩 ");

printf("考试成绩 ");

printf("总分\n");

while (p)

{

printf("%-16s\t", p->name);

printf("%-16s\t", p->lessonname);

printf("%-6d\t", p->classroom);

printf("%-6d\t", p->number);

printf("%-6d\t", p->normal);

printf("%-6d\t", p->exam);

printf("%-6d\t", p->score);

printf("\n");

count++;

p = p->next;

}

printf("\n\t\*\*共计%d个成绩信息\*\*",count);

return;

}

void Admin\_Choose()

{

int operation1, operation2;

struct Student\* temp = NULL;

if (pHead->next == NULL)

{

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

printf(" \*\t系统内无学生信息\t\*\n");

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

}

printf("\n 请输入操作数:");

scanf\_s("%d", &operation1);

printf("\n\n");

switch (operation1)

{

case 0:

if (pHead != NULL)

FreeAll(pHead);

exit(0);

break;

case 1:

FileR(pHead);

break;

case 2:

if (pHead->next == NULL)

{

printf("系统内无学生信息\n");

return;

}

printf("1、按姓名查询学生信息\n");

printf("2、按学号查询学生信息\n");

printf("3、按课程名称查询学生信息\n");

scanf\_s("%d", &operation2);

switch (operation2)

{

case 1:Search\_name(pHead);break;

case 2:Search\_num(pHead);break;

case 3:Search\_classname(pHead);break;

default:printf("无效操作数\n");

}

break;

case 3:Score\_sort(pHead); break;

case 4:Exam\_sort(pHead);break;

case 5:Excellent\_rate(pHead);break;

case 6:TravelList(pHead);break;

case 7:Search\_classname\_LandH(pHead);break;

case 8:Same\_class\_score(pHead);break;

case 9:Add\_stu(pHead);break;

case 10:Delete\_stu(pHead);break;

case 11:Modify\_information(pHead);break;

case 12:FileW(pHead);break;

case 13:admin = 0;break;

case 99:system("cls");break;

case 998:Delete\_data(); break;

default:

printf("无效的操作数\n");

printf("请输入操作数:");

rewind(stdin); //清空缓冲区

scanf\_s("%d", &operation1);

system("pause");

}

}

void Search\_num(struct Student \*p)

{

int temp;

int found = 0;

rewind(stdin); //清空缓冲区

printf("请输入要搜索的学号:");

scanf\_s("%d", &temp);

while (p)

{

if (p->number == temp)

{

Singe\_print(p);

found = 1;

}

p = p->next;

if (p == NULL && !found)

{

printf("未找到该学号学生的成绩\n");

}

}

}

void Singe\_print(struct Student\* p)

{

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

printf("姓名 课程名称 班级 学号 平时成绩 考试成绩 总分\n");

printf("%s", p->name);

printf("\t%s", p->lessonname);

printf("\t %d\t", p->classroom);

printf("%d\t ", p->number);

printf("%d ", p->normal);

printf(" %d ", p->exam);

printf(" %d ", p->score);

printf("\n");

}

void FreeAll(struct Student\* p)

{

struct Student\* temp = NULL;

while (p)

{

temp = p->next;

free(p);

p = temp;

}

}

struct Student Search\_name(struct Student\* p)

{

char buffer[16];

int flag = 1; //判断是否相等

int found = 0; //判断是否找到该学生

int i = 0;

rewind(stdin); //清空缓冲区

printf("请输入要查询学生的姓名:");

gets\_s(buffer);

while (p)

{

flag = strcmp(buffer, p->name);

if (flag == 0)

{

if(found == 0)

printf("\n已找到学生%s\n\n",buffer);

Singe\_print(p);

flag = 1;

found = 1;

//return \*p;

}

p = p->next;

if (p == NULL && !found)

{

printf("未找到学生%s\n", buffer);

system("pause");

}

}

return \*pHead;

}

void Search\_classname(struct Student\* p)

{

char buffer[16]; //缓冲区

int flag = 1; //判断是否相等

int i = 0;

int found = 0;

rewind(stdin); //清空缓冲区

printf("请输入要查询课程的名称:");

gets\_s(buffer);

while (p) //开始遍历

{

flag = strcmp(buffer, p->lessonname);

if (flag == 0)

{

Singe\_print(p);

flag = 1;

found = 1;

}

p = p->next;

}

if (found == 0)

{

printf("未找到该课程\n");

}

}

void Score\_sort(struct Student\* pHead) //总成绩排序

{

struct Student\* p, \* q, \* tail;

tail = NULL;

while ((pHead->next->next) != tail)

{

p = pHead;

q = pHead->next;

while (q->next != tail)

{

if ((q->score) > (q->next->score))

{

p->next = q->next;

q->next = q->next->next;

p->next->next = q;

q = p->next;

}

q = q->next;

p = p->next;

}

tail = q;

}

printf("\t\t按总分从小到大排序成功\n");

}

void Excellent\_rate(struct Student\* pHead) //优秀率

{

int classnum[32][5] = {0}; //[32]班级数量 [5]第一位班号,第二位总人数，第三位优秀人数，第四位不及格人数

struct Student\* p;

int i,j,flag,temp;

float rate;

flag = i = j = 0;

p = pHead->next;

while (p) //先给每个班一个位置

{

for (i = 0; i < 32; i++)

{

if (p->classroom == classnum[i][0])

break;

if (i == 31)

{

classnum[j][0] = p->classroom;

j++;

}

}

p = p->next;

}

p = pHead->next; //第二次遍历

while (p)

{

for (i = 0; i < 32; i++)

{

if (p->classroom == classnum[i][0]) //找到班级

{

classnum[i][1]++; //总人数++

//printf("%d班 总人数:%d ",classnum[i][0],classnum[i][1]);

if (p->score >= 90)

classnum[i][2]++; //优秀人数++

//printf("%d班 优秀人数:%d ", classnum[i][0], classnum[i][2]);

if (p->score < 60)

classnum[i][3]++; //不及格人数++

//printf("%d班 不及格人数:%d ", classnum[i][0], classnum[i][3]);

break;

}

}

p = p->next;

}

i = 0;

printf("1、单个班级查询\n2、全部班级\n请输入操作数:");

rewind(stdin); //清空缓冲区

scanf\_s("%d",&flag);

switch(flag)

{

case 1:

printf("请输入要查找的班级:");

rewind(stdin); //清空缓冲区

scanf\_s("%d",&temp);

while (classnum[i][0] != 0)

{

if (classnum[i][0] == temp)

{

rate = (float)classnum[i][2] / classnum[i][1]; //计算优秀率

printf("%d班优秀率为:%.2f\n", classnum[i][0], rate);

rate = (float)classnum[i][3] / classnum[i][1]; //计算不及格率

printf("%d班不及格率为:%.2f\n", classnum[i][0], rate);

return;

}

i++;

}

printf("要查找的班级不存在\n");

return;

case 2:

while (classnum[i][0] != 0) //开始遍历打印

{

rate = (float)classnum[i][2] / classnum[i][1]; //计算优秀率

printf("%d班优秀率为:%.2f\n", classnum[i][0], rate);

rate = (float)classnum[i][3] / classnum[i][1]; //计算不及格率

printf("%d班不及格率为:%.2f\n", classnum[i][0], rate);

i++;

}

break;

}

}

void Add\_stu(struct Student\* pHead)

{

struct Student\* p, \* tail;

char operation = 'Y';

char buffer[16];

int temp\_score, temp\_normal, temp\_exam;

temp\_exam = temp\_normal = temp\_score = 0;

while (pHead->next) //遍历直到链表的末尾

{

pHead = pHead->next;

}

tail = pHead; //记录尾指针

while (1)

{

switch (operation)

{

case 'Y':

p = (struct Student\*)malloc(lenth);

Cheak(p);

p->next = (struct Student\*)NULL;

printf("请输入要插入的学生的姓名:");

rewind(stdin); //清空缓冲区

gets\_s(buffer);

strcpy(p->name,buffer);

printf("请输入要插入的学生的课程名称:");

rewind(stdin); //清空缓冲区

gets\_s(buffer);

strcpy(p->lessonname, buffer);

printf("请输入要插入的学生的班级:");

rewind(stdin); //清空缓冲区

scanf\_s("%d",&p->classroom);

printf("请输入要插入的学生的学号:");

rewind(stdin); //清空缓冲区

scanf\_s("%d", &p->number);

printf("请输入要插入的学生的平时成绩:");

rewind(stdin); //清空缓冲区

scanf\_s("%d", &temp\_normal);

p->normal = temp\_normal;

printf("请输入要插入的学生的考试成绩:");

rewind(stdin); //清空缓冲区

scanf\_s("%d", &temp\_exam);

p->exam = temp\_exam;

temp\_score = (int)((0.20 \* temp\_normal) + (0.80 \* temp\_exam) + 0.5); //计算并四舍五入

p->score = temp\_score;

tail->next = p;

tail = p;

Singe\_print(p);

printf("是否继续插入？（Y/N）");

rewind(stdin); //清空缓冲区

scanf\_s("%c",&operation);

break;

case 'N':

return;

default:

printf("请输入正确的操作数\n");

system("pause");

rewind(stdin); //清空缓冲区

scanf\_s("%c", &operation);

}

}

}

void Search\_classname\_LandH(struct Student\* pHead)

{

struct Student\* p\_l; //最低位

struct Student\* p\_h; //最高位

struct Student\* p;

char buffer[16]; //缓冲区

int flag = 1; //判断是否相等

int i = 0;

int found = 0;

int temp\_score = -1;

p = pHead;

p\_l = (struct Student\*)malloc(lenth);

Cheak(p\_l);

p\_h = (struct Student\*)malloc(lenth);

Cheak(p\_h);

p\_l->score = 101;

p\_h->score = -1;

rewind(stdin); //清空缓冲区

printf("请输入要查询课程的名称:");

gets\_s(buffer);

while (buffer[i] != 0)

{

i++;

}

buffer[i] = 0; //加上字符串结束符

p = p->next;

while (p) //开始遍历

{

flag = strcmp(buffer, p->lessonname);

if (flag == 0)

{

//Singe\_print(p); //找到后打印出来

flag = 1; //重装载

found = 1;

if (p->score > p\_h->score)

{

p\_h = p;

}

if (p->score < p\_l->score)

{

p\_l = p;

}

}

p = p->next;

}

if (found == 0)

{

printf("未找到该课程\n");

return;

}

//利用查询课程名称和成绩函数打印所有同分

printf("\n\n"); //先打印高分

printf("该课程的最高分为\n");

p = pHead; //回到头指针

p = p->next;

temp\_score = p\_h->score;

while (p) //开始遍历

{

flag = strcmp(buffer, p->lessonname);

if (flag == 0)

{

flag = 1;

if (p->score == temp\_score) //读取分数

{

found = 1;

Singe\_print(p);

}

}

p = p->next;

}

if (found == 0)

{

printf("High error\n");

system("pause");

}

printf("\n\n"); //打印低分

printf("该课程的最低分为\n");

p = pHead; //回到头指针

p = p->next;

temp\_score = p\_l->score;

while (p) //开始遍历

{

flag = strcmp(buffer, p->lessonname);

if (flag == 0)

{

flag = 1;

if (p->score == temp\_score) //读取分数

{

found = 1;

Singe\_print(p);

}

}

p = p->next;

}

if (found == 0)

{

printf("Low error\n");

system("pause");

}

}

void Same\_class\_score(struct Student\* p)

{

char buffer[16]; //缓冲区

int flag = 1; //判断是否相等

int i = 0;

int found = 0;

int temp\_score = -1;

rewind(stdin); //清空缓冲区

printf("请输入要查询课程的名称:");

gets\_s(buffer);

while (buffer[i] != 0)

{

i++;

}

buffer[i] = '\n'; //配对最后一位的换行符

buffer[i + 1] = 0; //加上字符串结束符

printf("请输入要查找的分数:");

rewind(stdin); //清空缓冲区

scanf\_s("%d",&temp\_score);

p = p->next;

while (p) //开始遍历

{

flag = strcmp(buffer, p->lessonname);

if (flag == 0)

{

flag = 1;

if (p->score == temp\_score) //读取分数

{

found = 1;

Singe\_print(p);

}

}

p = p->next;

}

if (found == 0)

{

printf("未找到该课程下的该分数的同学\n");

}

}

void Delete\_stu(struct Student\* p)

{

struct Student\* temp;

int temp\_num;

printf("请输入要删除的学生的学号:");

rewind(stdin); //清空缓冲区

scanf\_s("%d",&temp\_num);

while (p->next)

{

if (p->next->number == temp\_num) //找到要删除的节点

{

temp = p->next->next;

free(p->next);

p->next = temp;

printf("删除成功\n");

return;

}

p = p->next;

}

printf("未找到该学号的学生\n");

return;

}

void Modify\_information(struct Student\* p) //修改学生信息

{

char buffer[16];

int mode,temp,tmep\_score;

temp = mode = -1;

printf("1、修改学生的班级\n");

printf("2、修改学生的学号\n");

printf("3、修改学生的平时成绩\n");

printf("4、修改学生的考试成绩\n");

printf("选择是:");

rewind(stdin); //清空缓冲区

scanf\_s("%d",&mode);

if (mode >= 1 && mode <= 4)

{

printf("该学生的学号是:");

rewind(stdin); //清空缓冲区

scanf\_s("%d",&temp);

}

else

{

printf("错误的操作数\n"); //错误的操作数直接返回，一点面子都不给（防止继续进行报错）

system("pause");

return;

}

while (p) //找到该学号的学生

{

if (p->number == temp)

{

printf("已找到该学生\n");

break;

}

p = p->next;

if (p == NULL)

{

printf("未找到该学生\n");

system("pause");

return;

}

}

rewind(stdin); //清空缓冲区

switch (mode)

{

case 1:

printf("该学生新的班级是:");

scanf\_s("%d",&temp);

p->classroom = temp;

break;

case 2:

printf("该学生新的学号是:");

scanf\_s("%d",&temp);

p->number = temp;

break;

case 3:

printf("该学生的平时成绩修改为:");

scanf\_s("%d", &temp);

p->normal = temp;

break;

case 4:

printf("该学生的考试成绩修改为:");

scanf\_s("%d", &temp);

p->exam = temp;

break;

}

if (mode == 3 || mode == 4)

{

p->score = (float)p->normal \* 0.2 + (float)p->exam \* 0.8 + 0.5; //四舍五入

}

printf("修改成功\n");

}

void FileW(struct Student\* pHead)

{

char buffer[16]; //建立缓冲区

struct Student\* p;

FILE\* fpWrite = NULL;

errno\_t err; //与fopen相关

if ((err = fopen\_s(&fpWrite, "Student.txt", "w")) != NULL) //成功返回0，不成功返回非0

{

printf("未找到Student.txt文件\n");

system("pause");

exit(-1);

}

p = pHead->next;

while (p)

{

strcpy(buffer, p->name); //写入名字

strcat(buffer, "\n");

fputs(buffer,fpWrite);

strcpy(buffer, p->lessonname); //写入课程名称

strcat(buffer,"\n");

fputs(buffer,fpWrite);

fprintf(fpWrite,"%d",p->classroom);

fputc('\0', fpWrite); //作为数据的隔断

fprintf(fpWrite, "%d", p->number);

fputc('\0', fpWrite);

fprintf(fpWrite, "%d", p->normal);

fputc('\0',fpWrite);

fprintf(fpWrite, "%d", p->exam);

if(p->next)

fputs("\n", fpWrite);

p = p->next;

}

fclose(fpWrite);

}

void AdminAccount(char Account[],char Password[])

{

char buffer\_Account[16];

char buffer\_Password[16];

int flag = -1;

FILE\* fpRead;

errno\_t err; //与fopen相关

if ((err = fopen\_s(&fpRead, "Account.txt", "r")) != NULL) //成功返回0，不成功返回非0

{

printf("未找到Account.txt文件\n");

system("pause");

exit(-1);

}

while (!feof(fpRead))

{

fgets(buffer\_Account, sizeof(buffer\_Account), fpRead); //读取Admin账号

//puts(buffer\_Account);

buffer\_Account[strlen(buffer\_Account) - 1] = 0; //消除换行符

flag = strcmp(buffer\_Account, Account);

if (flag != 0)

continue;

fgets(buffer\_Password, sizeof(buffer\_Password), fpRead); //读取Admin密码

//puts(buffer\_Password);

buffer\_Password[strlen(buffer\_Password) - 1] = 0; //消除换行符

flag = strcmp(buffer\_Password, Password);

if (flag != 0)

continue;

admin = 1;

return;

}

printf("账号密码错误\n");

system("pause");

return;

}

void Exam\_sort(struct Student\* pHead) //总成绩排序

{

struct Student\* p, \* q, \* tail;

tail = NULL;

while ((pHead->next->next) != tail)

{

p = pHead;

q = pHead->next;

while (q->next != tail)

{

if ((q->exam) > (q->next->exam))

{

p->next = q->next;

q->next = q->next->next;

p->next->next = q;

q = p->next;

}

q = q->next;

p = p->next;

}

tail = q;

}

printf("\t\t按考试成绩从小到大排序成功\n");

}

void Choose()

{

int operation1, operation2;

struct Student\* temp = NULL;

if (pHead->next == NULL)

{

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

printf(" \*\t系统内无学生信息\t\*\n");

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

}

printf("\n 请输入操作数:");

scanf\_s("%d", &operation1);

printf("\n\n");

switch (operation1)

{

case 0:

if (pHead != NULL)

FreeAll(pHead);

exit(0);

break;

case 1:FileR(pHead);break;

case 2:

if (pHead->next == NULL)

{

printf("系统内无学生信息\n");

return;

}

printf("1、按姓名查询学生信息\n");

printf("2、按学号查询学生信息\n");

printf("3、按课程名称查询学生信息\n");

scanf\_s("%d", &operation2);

switch (operation2)

{

case 1:Search\_name(pHead);break;

case 2:Search\_num(pHead);break;

case 3:Search\_classname(pHead);break;

default:printf("无效操作数\n");

}

break;

case 3:Score\_sort(pHead); break;

case 4:Exam\_sort(pHead); break;

case 5:Excellent\_rate(pHead); break;

case 6:TravelList(pHead); break;

case 7:Search\_classname\_LandH(pHead); break;

case 8:Same\_class\_score(pHead); break;

case 9:Login(); break;

case 99:system("cls"); break;

default:

printf("无效的操作数\n");

printf("请输入操作数:");

rewind(stdin); //清空缓冲区

scanf\_s("%d", &operation1);

system("pause");

}

}

void Delete\_data()

{

char key = 0;

printf("确定要删库跑路？ Y/N \n");

rewind(stdin); //清空缓冲区

scanf\_s("%c",&key);

if (key == 'Y')

{

if (\_unlink("Student.txt") == 0)

printf("成功删库，赶紧跑路！\n");

system("pause");

}

else

printf("没有删除\n");

rewind(stdin); //清空缓冲区

}

部分功能截图

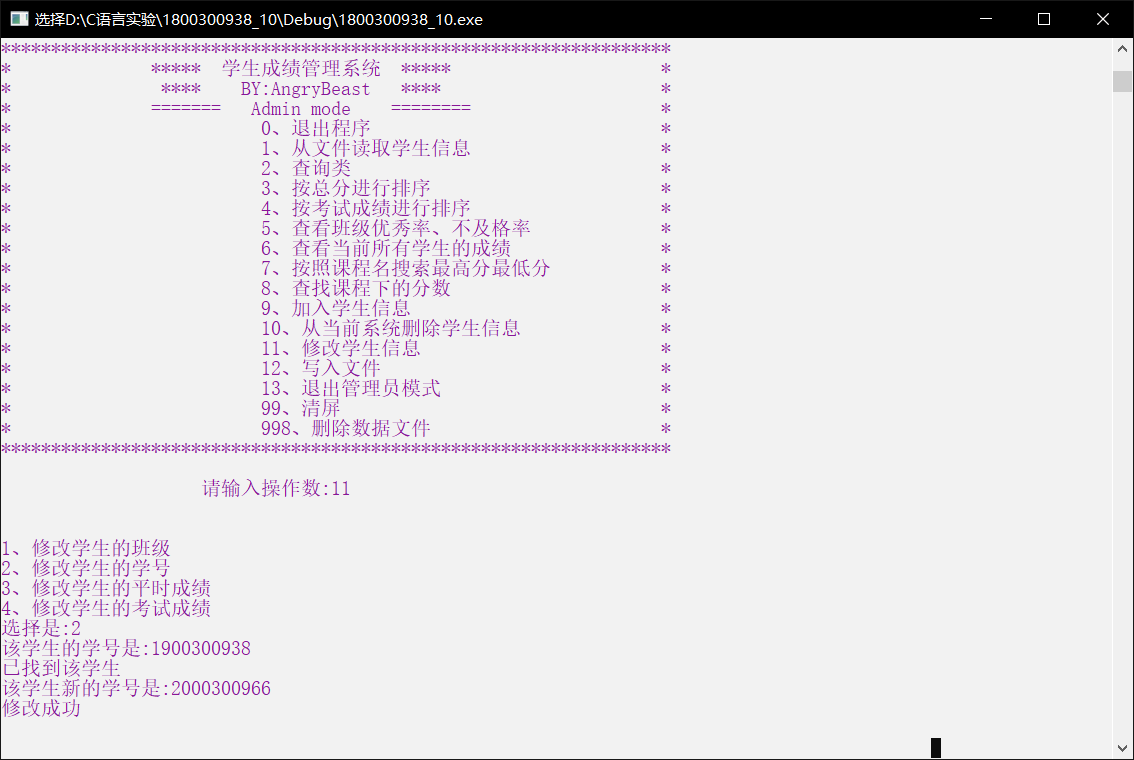
加入学生信息



按照课程名称搜索最高分最低分



修改学生成绩



登陆管理员账号



|  |
| --- |
| **四．实验小结** |