**《问题求解与程序设计》课程**

**实验报告**

**学生成绩管理系统设计与实现**

学 生 姓 名： 杨功卓

专业、班级 ： 软件1901

学 号 ： 2019214257

联 系 电 话： \*\*\*\*\*\*\*\*\*\*\*

Email ： 1779249225@qq.com

完 成 日 期： 2020/04/18

# 设计任务

随着信息时代的到来，办公趋向无纸化、高效化，对于学生成绩的管理更是如此，基于此，我选择了学生成绩管理系统作为我的课题，希望能够模仿教务系统中成绩的录入与查询功能，编写自己的成绩管理系统。实现了成绩的录入、删除、修改、排序与查询功能。

# 程序设计与实现

## 程序结构

本程序面向两类用户即教师与学生，有两个入口，教师登录需密码，共有添加，删除，修改，查找，密码修改五个功能，学生权限为查看本人详细成绩和修改个人账户密码。学生成绩管理系统功能结构图如图1.所示。

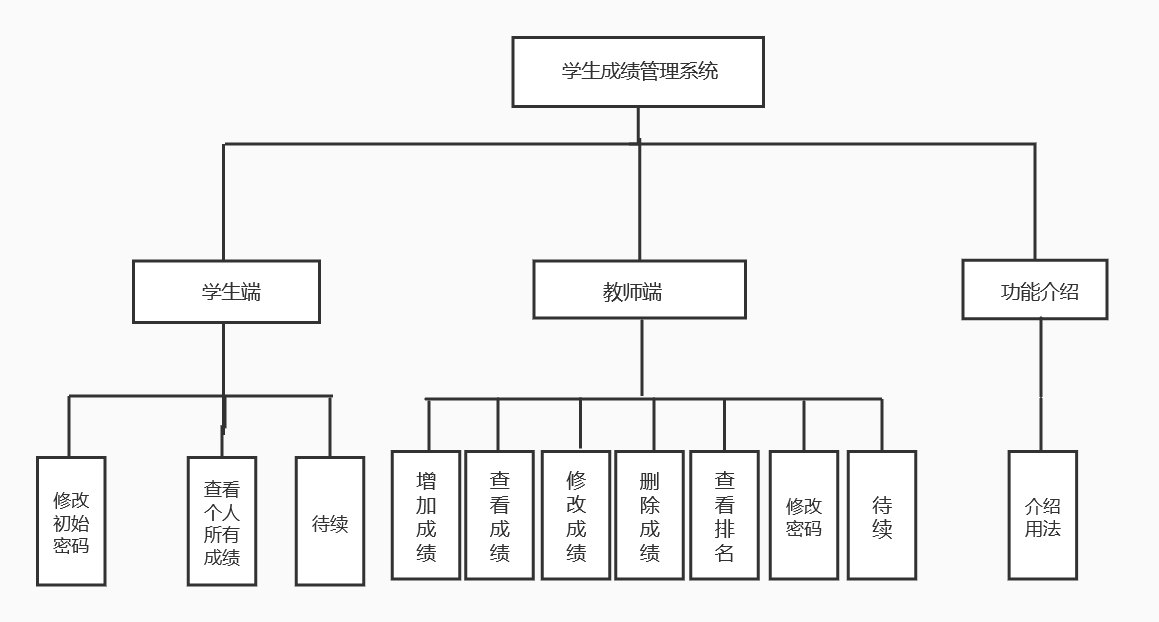


图1 学生成绩管理系统功能结构图

## 数据描述

1. 学生信息存储

代码中定义结构体类型Student来存储单个学生的信息。结构体类型定义及结构体成员含义说明如下所示。

typedef struct Student{

int id; //学生学号

char pswd[11]; //密码

char name[20]; //姓名

int c\_grade; //C语言成绩

int math\_grade;//数学成绩

int english\_grade;//英语成绩

}Student;；

学生类型的变量用链表进行存储，链表节点Node定义如下：

typedef struct node{//定义节点，数据域是一个Student类型的变量

Student stu;

struct node \*next;

}Node;

（2）教师信息存储

代码中定义结构体Teacher来存储教师信息，定义结构体成员含义如下：

typedef struct teacherMesg Teacher;

struct teacherMesg{// 定义结构体：教师账号信息

char account[11] ;//账号

char pswd[11] ; //密码

char question[50]; //密保

char answer[30]; //答案

}teacher\_mesg;

教师类型的变量用链表进行存储，链表节点tNode定义如下：

typedef struct teacherNode{

Teacher teach;

struct teacherNode \*next;

}tNode;

（3）全局变量设计

定义全局变量如下：

FILE \*teacher\_file = NULL; //用来读写老师信息文件的指针

FILE \*student\_file = NULL;//指向存储学生信息的文件

char pswd1[11]; //pswd1、pswd2是注册框两个密码接收变量

char pswd2[11];

char question[41]; //注册时的密保变量

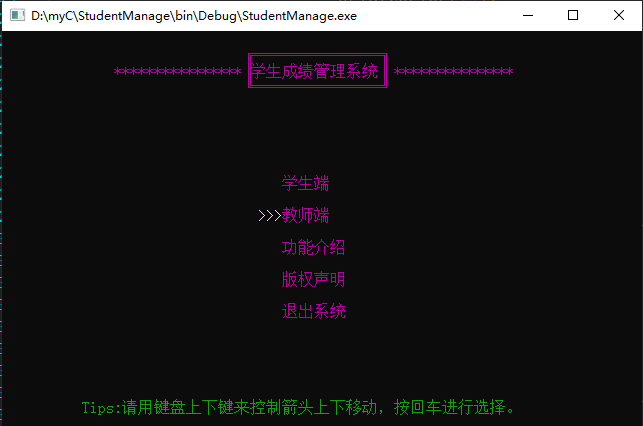
char answer[21]; //注册时的密保问题答案

int press\_times; //用来记录密码框输入字符个数，用来在密码框打印’\*’回显字符

## 功能模块设计

**本部分前三个函数是一个选择目录框架的三个主要组成部分**

（1）移动光标以及返回一个用户最后选择的目录的行坐标int值函数int choiceReact().



**这张图用的就是这个框架。**

（\*\*本函数是程序最重要的一个函数\*\*）程序中所使用的所有的目录选择功能和键盘控制箭头操作均是由一个框架所实现，框架就是一个选择函数搭配页面打印函数及一个switch循环所合成。

选择函数代码如下：（选择函数本身也是用的这种框架，由一个页面打印函数和switch循环所构成）

选择函数定义：int choiceReact(int content\_x,int content\_y,int gap,int up\_lim,int down\_lim,void(\*fOut));

\*此函数通过控制”>>>”箭头来显示用户的键盘操作情况，getch()函数获得键盘键值，用switch循环来进行箭头的反馈，最终在进行选择完毕之后返回一个反馈值，

来表示最终的选择结果，详细操作就是，透过参数获得箭头”>>>”移动的规则，根据页面的目录的行和列的坐标，以及行与行之间的间隔，最终要打印的函数的指针，

这种一系列计算来控制”>>>”箭头的仿真移动，最终的返回值也是每行目录的所在行坐标，功能函数接收行坐标之后，才能了解到用户选择了哪个目录。

下面函数所牵扯到的printColor()是打印字体的颜色，printArrow()是打印”>>>”的，

int choiceReact(int content\_x,int content\_y,int gap,int up\_lim,int down\_lim, void (\*fOut)()){// choicReact() case里的各种值都是定义的键盘键值常量（上、下、回车等）。

/\*\*\*函数有六个参数，1、2是首行目录坐标，参数3是目录间隔，参数4、5是上、下端纵坐标，参数6是指针函数\*\*\*/

//content\_x = 30; //退三格 // 在开始目录输出箭头指定玩家下一步操作，返回一个玩家的选择int值

//content\_y = 11;

int sub = down\_lim - up\_lim;

int user\_keyboard; //用户操作键盘键位值

int user\_choice = up\_lim;

printCorlor(14);

printArrow(content\_x , content\_y);

printCorlor(12);

moveNotice(10,23,"Tips:请用键盘上下键来控制箭头上下移动，按回车进行选择。");

while(1){

fOut();

if((user\_keyboard = getch())){

switch(user\_keyboard){

case UP: if(user\_choice == up\_lim){

system("cls");

fOut();

printCorlor(14);

printArrow(content\_x , content\_y + sub);

user\_choice = down\_lim;

content\_y = down\_lim;

}

else{

system("cls");

fOut();

printCorlor(14);

printArrow(content\_x , content\_y - gap);

content\_y -= gap;

user\_choice -= gap;

}

break;

case DOWN: if(user\_choice == down\_lim){

system("cls");

fOut();

printCorlor(14);

printArrow(content\_x , content\_y - sub);

user\_choice = up\_lim;

content\_y = up\_lim;

}

else{

system("cls");

fOut();

printCorlor(14);

printArrow(content\_x , content\_y + gap);

content\_y += gap;

user\_choice += gap;

}

break;

case ENTER: Sleep(400);

system("cls");

return user\_choice;

break;

}

}

}

}

1. 框架打印函数，用来打印用户选择目录，只是打印出来UI，供那些操作函数来作为标准进行动作。通过一系列的清屏，来达到动态的

效果（比如移动一次>>>箭头，就清屏一次，然后再调用打印目录的函数，再次打印，这样就达到箭头移动效果了）。

void outputContents(){// outputContents() 游戏目录 接口：玩家选择目录的数字输入

isCursorActive(0);

//gameUI(14);

printCorlor(13);

cursorGoto(14,1);

printf(" ╔════════════════╗ ");

cursorGoto(14,2);

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

cursorGoto(14,3);

printf(" ╚════════════════╝ ");

cursorGoto(35,9);

printf("学生端");

cursorGoto(35,11);

printf("教师端");

cursorGoto(35,13);

printf("功能介绍");

cursorGoto(35,15);

printf("版权声明");

cursorGoto(35,17);

printf("退出系统");

moveNotice(10,23,"Tips:请用键盘上下键来控制箭头上下移动，按回车进行选择。");

}

1. 接收choiceReact()函数的返回值来进行反应的函数（用switch来接收返回值进行操作）。

代码如下：在while循环里每次调用choiceReact()来进行箭头移动，再进行接收，从而做出最终的反应。

void WelcomeUI(){// WelcomeUI() 打开系统后选择页面

while(1){

int i;

outputContents();

i = choiceReact(32,9,2,STUSIDE,LEAVE,outputContents);

switch(i){

case STUSIDE : ///学生端

stuUserInputInterface();

break;

case TEACHSIDE : ///教师端

Sleep(1000);

userUI();

break;

case INTRODUCTION : ///功能介绍

system("cls");

break;

case DECLARE : ///声明

system("cls");

noticeCopyright();

break;

case LEAVE : ///退出

if(warningDialog("您确定要退出系统吗？","退出提醒","您已退出系统")){

exit(1);

}

break;

}

}

}

1. 打印字体颜色函数：void printCorlor(int corlor)；传递一个颜色值(int)，来控制字体之后的字体颜色
2. 控制光标移动到具体坐标函数：void cursorGoto(int x,inty);//参数是x,y坐标。
3. 控制光标是否显示函数：iscursorActive(int state);//state是光标状态，1是显示，0是隐藏
4. 对话框弹出函数：int warningDialog(const char \*text , const char \*title , const char \*opResult);三个字符常量分别是对话框内容、标题以及再进行选择之后在控制台打印的内容，返回的int值是选择的结果，yes是1，no是0
5. 打印出动态进度条函数：void gameBuffer(); 运用算法实现：在for循环中的特定时间段里用sleep()实现缓冲速度快慢。
6. 是否为密码设置回显字符函数：void saveOrNot(int is\_not);参数1是显示，0是隐藏。
7. 输入文本框的操作函数，代码如下：（输入框类函数容错率是最高的，和现有的程序中的输入框容错性能一样）

输入使用getch()函数，能及时判断输入的正确性，容错性能很好，如果是非int型数据，则会进行提示，而不会进行存储。

Ⅰ.登陆账号部分：

void goToAccount(){// goToAccount() 输入账号接口实现函数

int i = 0;

int j = 31;

char str;

isCursorActive(1);

strcpy(teacher\_mesg.account,"");

cursorGoto(31,5);

printf(" ");

while(1){

if(i == 10){

break;

}

cursorGoto(j,5);

str = getch();

if( str >= '0' && str <= '9'){

teacher\_mesg.account[i] = str;

printf("%c",teacher\_mesg.account[i]);

i ++;

j ++;

}

else{

cursorGoto(52,5);

printCorlor(0x04);

printf("！请输入有效数字");

printCorlor(0x02);

}

}

}

Ⅱ.登录密码部分

增加了回显字符的接口。

void goToPswd(){// goToPswd() 输入密码接口实现函数

int i = 0;

int j = 31;

char str;

press\_times = 0;

isCursorActive(1);

strcpy(teacher\_mesg.pswd,"");

cursorGoto(31,8);

printf(" ");

while(1){

if(i == 10){

break;

}

cursorGoto(j,8);

str = getch();

if(str == ENTER){

//strcpy(teacher\_mesg.account,temp);

/\*strcpy(temp,teacher\_mesg.account);

int k = 0;

for(;k < 10; k ++){

teacher\_mesg.account[k] = temp[k];

}\*/

break;

}

if((str>='0' && str<='9') || ((str>='A' && str<='Z') || (str>='a' && str<='z'))){

teacher\_mesg.pswd[i] = str;

printf("%c",teacher\_mesg.pswd[i]);

i ++;

j ++;

++ press\_times;

}

else{

cursorGoto(52,8);

printCorlor(0x04);

printf("！请输入有效数字");

printCorlor(0x02);

}

}

}

1. 读取文件以及文件操作类型函数，整个程序中的文件操作和改写的基本思想就是，先将文件内容按一定格式读取到指定类型的链表中，再对链表进行操作（增删改等），然后将处理过的链表重新写入文件。

代码如下：

Ⅰ.文件读出。

Node\* readAndWrite(Node \*head,const char \*file\_name){// readAndWrite() 将将文件读入链表中 两个参数分别是链表头节点和文件名

FILE \*file = NULL;

Node \*this\_head = head;

Node \*temp = NULL;

this\_head = (Node\*)malloc(sizeof(Node));

Node \*now = this\_head;

file = fopen(file\_name,"r");

if(this\_head == NULL){

printf("Read\_Write\_NullPointer\_Error\n");

exit(-1);

}else{

int id = 0;

int c\_gra = 0;

int m\_gra = 0;

int e\_gra = 0;

char name[20];

char pswd[11];

while((fscanf(file,"%d %s %s %d %d %d\n",&id,pswd,name,&c\_gra,&m\_gra,&e\_gra)) != EOF){

temp = (Node\*)malloc(sizeof(Node));

temp->next = NULL;

now->next = temp;

now = temp;

now->stu.id = id;

strcpy(now->stu.pswd,pswd);

strcpy(now->stu.name,name);

now->stu.c\_grade = c\_gra;

now->stu.math\_grade = m\_gra;

now->stu.english\_grade = e\_gra;

}

}

fclose(file);

return this\_head;

}

Ⅱ.文件写入函数。

Node\* readAndWrite(Node \*head,const char \*file\_name){// readAndWrite() 将链表读进指定文件中 两个参数分别是链表头节点和文件名

FILE \*file = NULL;

Node \*this\_head = head;

Node \*temp = NULL;

this\_head = (Node\*)malloc(sizeof(Node));

Node \*now = this\_head;

file = fopen(file\_name,"r");

if(this\_head == NULL){

printf("Read\_Write\_NullPointer\_Error\n");

exit(-1);

}else{

int id = 0;

int c\_gra = 0;

int m\_gra = 0;

int e\_gra = 0;

char name[20];

char pswd[11];

while((fscanf(file,"%d %s %s %d %d %d\n",&id,pswd,name,&c\_gra,&m\_gra,&e\_gra)) != EOF){

temp = (Node\*)malloc(sizeof(Node));

temp->next = NULL;

now->next = temp;

now = temp;

now->stu.id = id;

strcpy(now->stu.pswd,pswd);

strcpy(now->stu.name,name);

now->stu.c\_grade = c\_gra;

now->stu.math\_grade = m\_gra;

now->stu.english\_grade = e\_gra;

}

}

fclose(file);

return this\_head;

}

# 设计难点与解决方法

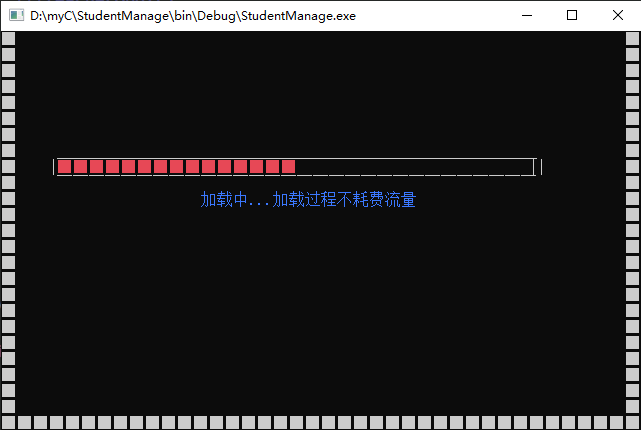
（1）在教师端登录进去的时候，会自动生成班级，我原本的代码意思是登录后检查有没有班级信息文件（和账户名称相同的txt文件，例如：1234567890.txt）如果没有此文件就创建一个，然后在每次登录之后，就对该文件排序一次，但是，再过了几次编译之后，程序突然出现了奇怪的bug，如果班级信息文件为空，就会直接程序终止，解决办法只能在此账户登录之前保证班级信息文件不为空，或在刚注册新账户时候，对其进行手动添加学生信息。

# UI设计效果

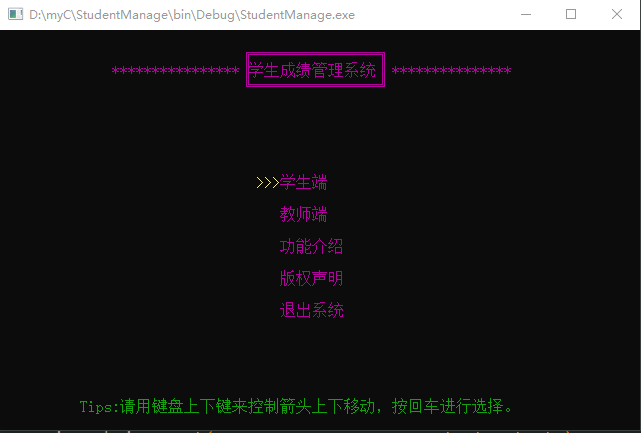
佛祖保佑：



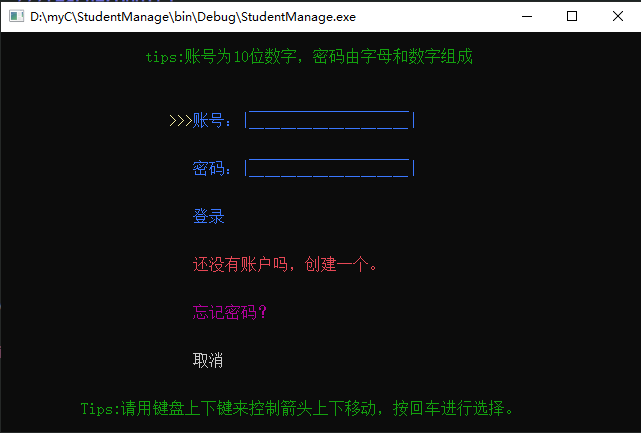
4.0模拟缓冲效果



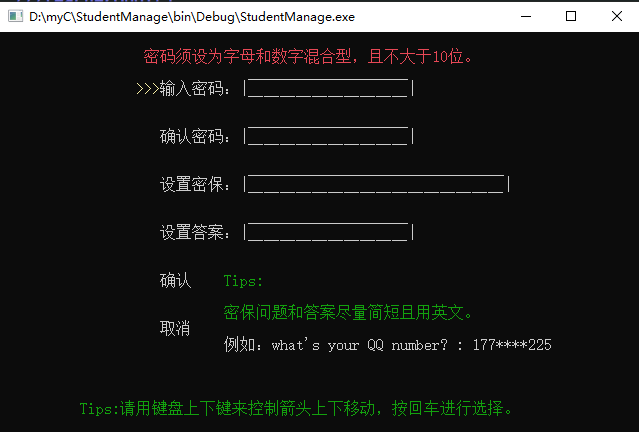
4.1 系统启动界面



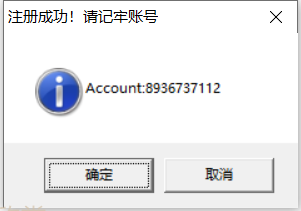
4.2 教师端登录页面



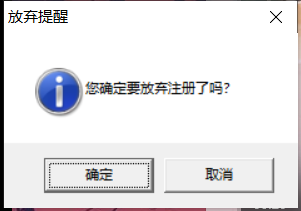
4.3 创建账户页面



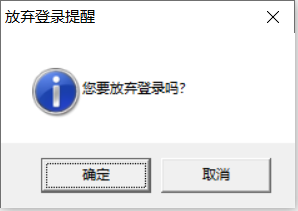
4.4注册成功对话框



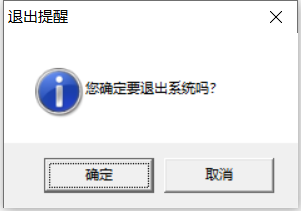
4.5放弃注册对话框



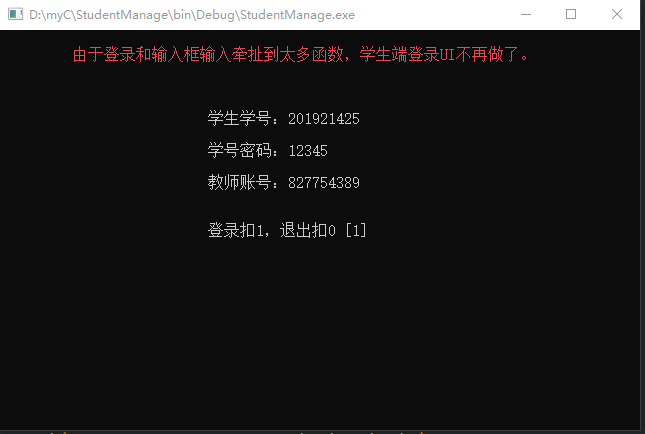
4.6放弃登录对话框



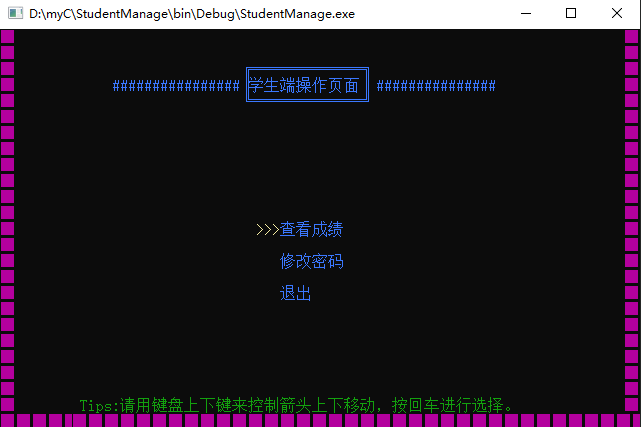
4.7退出系统提醒



4.8学生登录界面



4.9学生登录后界面



4.4 项目到目前为止存在的不足及改正措施：

（1）数据输出不整齐

（2）部分功能未实现

# 单元测试

5.1 xxx函数

5.1.1 测试用例设计

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 测试用例 | 覆盖路径 | 期望输出 | 实际输出 | 是否通过 |  |
|  |  |  |  |  |  |  |

5.1.2 测试代码

# 总结

自己在做项目的时候遇到的最严重的问题就是不能把控住项目的进度，项目完成日期不可控。另外一点就是软件工程过程很重要，需求分析、各种模型等，我就在这方面吃了大亏，项目进度和项目发展很混乱，代码质量不良，很是浪费时间，希望在课下多多了解这方面的知识，对做项目帮助很大。

# 附件（程序代码）

程序源代码

7.1 main.c

