**《程序设计基础课程设计》第5周实验报告**

班级：1803012

姓名：杨煜

学号：18030100204

所做题目：一、综合系统——简单文件数据库

**第9题**

**1.原始题目及要求；**

**（1）高精度计算**

要求：用户分为管理员和客户两类，分别显示不同文本格式菜单，通过菜单项对应数字进行选择。客户菜单包括查找及点歌，可按歌名查找同名所有歌曲并显示相关信息，或按歌手名查找其所有歌曲，点歌后显示所点歌曲歌词。管理员菜单包括可添加和删除歌曲，每个歌曲的歌词用一个单独的文件存储。

命令行参数如下：

KTVsim –a(-u) xxxx

第一个参数为可执行程序名称；第二个参数为用户身份，-a表示管理员，-u表示客户；

按管理员和客户不同，显示对应的操作菜单；

**2.题目的分析**

（1）涉及知识点

文件读写、内存管理、结构体定义、基本数据结构、高级格式化输入输出

（2）题目功能理解

本题的主要难点在于文件的读写，在设计过程中应随时注意文件的打开模式，及文件的关闭时机。

**3. 题目的总体设计：设计思路、算法描述**

（1）程序模块：

（A）添加歌曲模块

为ktv添加歌曲。

（B）删除歌曲模块

为ktv删除歌曲。

（C）管理员模式模块

打开管理员模式。

（D）查找歌手模块

查找某一歌手的所演唱的歌曲。

（E）查找同名歌曲模块

查找同名的所有歌曲。

（F）用户查找歌曲选项的二级界面模块

打开用户查找歌曲选项的二级界面。

（G）播放模块

播放歌曲。

（H）用户模式模块

打开用户模式。

（I）开始模块

开始使用ktv

（2）模块调用关系

开始模块调用用户模式模块，管理员模式模块；用户模式模块调用播放模块，用户查找歌曲选项的二级界面模块；用户查找歌曲选项的二级界面模块调用查找歌手模块，查找同名歌曲模块；管理员模式模块调用添加歌曲模块，删除歌曲模块。

（3）输入输出数据说明

输入数据：形如KTVsim –a(-u) xxxx的命令行命令。

输出数据：操作结果。

（4）总体流程

输入命令行参数后根据提示进行操作即可。

**4. 各功能模块/函数的设计说明：**

（1）void admin\_add\_song()

函数名称: admin\_add\_song

函数功能: 为ktv添加歌曲

参数: 无

返回值: void

执行流程：

1）打开现有歌曲列表，若成功转2，否则提示错误后返回。

2）提示用户输入歌曲名，作者名，读取后转3。

3）与现存歌曲列表中的结果逐一比对，若找到对应的歌曲, 则设置存在标志并退

出比较，转4。

4）如果歌曲已经存在，则提示已存在，返回。否则转5。

5）以附加写入方式重新打开文件, 准备添加歌曲。转6。

6）创建新歌曲的歌词文件，成功转7，否则，提示错误并返回。

7）打开新歌曲的歌词文件，并读入歌词。转8。

8）关闭打开的各文件，返回。

可能结果：

1）添加歌曲成功

2）添加歌曲失败

（2）void admin\_delete\_song()

函数名称: admin\_delete\_song()

函数功能: 为ktv删除歌曲

参数: 无

返回值: void

执行流程：

1）打开现有歌曲列表，并创建新的歌曲列表，若成功转2，否则提示错误后返回。

2）提示用户输入歌曲名，作者名，读取后转3。

3）与现存歌曲列表中的结果逐一比对，若找到对应的歌曲, 则不写入新的歌曲列

表，否则写入，转4。

4）移除旧歌曲列表，转5。

5）移除要删除的歌曲文件的歌词文件。转6。

6）重命名新的歌曲列表的文件名，转7。

7）提示操作成功，返回。

可能结果：

1）删除歌曲成功

2）删除歌曲失败

（3）void admin( )

函数名称: admin

函数功能: 打开管理员模式

参数: 无

返回值: void

执行流程：

1）显示操作界面，转2。

2）读取用户操作选项，若非退出，根据选项做出相应操作，否则，转3。

3）返回。

可能结果：

1）执行相关操作。

（4）void user\_find\_song\_name()

函数名称: user\_find\_song\_name

函数功能: 查找同名的所有歌曲

参数: 无

返回值: void

执行流程：

1）打开现有歌曲列表，若成功转2，否则提示错误后返回。

2）提示用户输入歌曲名，读取后转3。

3）与现存歌曲列表中的结果逐一比对，若找到对应的歌曲, 则予以显示

否则, 不显示，转4。

4）提示查询结束，转5。

5）关闭打开的各文件，返回。

可能结果：

1）查找成功，显示结果。

2）查找失败，提示失败。

（5）void user\_find\_song\_singer()

函数名称: user\_find\_song\_singer

函数功能: 查找某一歌手的所演唱的歌曲

参数: 无

返回值: void

执行流程：

1）打开现有歌曲列表，若成功转2，否则提示错误后返回。

2）提示用户输入歌手名，读取后转3。

3）与现存歌曲列表中的结果逐一比对，若找到对应的歌手, 则予以显示

否则, 不显示，转4。

4）提示查询结束，转5。

5）关闭打开的各文件，返回。

可能结果：

1）查找成功，显示结果。

2）查找失败，提示失败。

（5）void user\_find\_song()

函数名称: user\_find\_song

函数功能: 用户查找歌曲选项的二级界面

参数: 无

返回值: void

执行流程：

1）显示操作界面，转2。

2）读取用户操作选项，若非退出，根据选项做出相应操作，否则，转3。

3）返回。

可能结果：

1）执行相关操作

（6）void user\_play\_song()

函数名称: user\_play\_song

函数功能: 播放歌曲

参数: 无

返回值: void

执行流程：

1）打开现有歌曲列表，若成功转2，否则提示错误后返回。

2）提示用户输入歌曲名，作者名，读取后转3。

3）与现存歌曲列表中的结果逐一比对，若找到对应的歌曲, 则设置存在标志并退

出比较，转4。

4）如果歌曲不存在，则提示不存在，返回。否则转5。

5）打开歌曲歌词文件，若失败则提示并返回。否则转6。

6）逐行显示歌词内容，转7。

7）提示点歌成功，关闭打开的文件，返回。。

可能结果：

1）播放歌曲成功

2）播放歌曲失败

（7）void ktv\_start( char user\_level, const char \*name )

函数名称: ktv\_start

函数功能: 开始使用ktv

参数: char user\_level( 操作者的身份 ), const char \*name( 操作者的姓名 )

返回值: void

执行流程：

1）根据用户身份打开对应列表。若身份不存在，则直接返回。否则转2。

2）查找对应用户是否存在，若不存在直接返回，否则转3。

3）显示操作界面，转4.

4）读取用户操作选项，若非退出，根据选项做出相应操作，否则，转5。

5）结束

可能结果：

1）执行相关操作

**5. 源程序：**

（1）ktv.c

#include"ktv.h"

#include<stdio.h>

#include<stdlib.h>

int main( int argc, char\* argv[] ){

if( argc != 3 ){

printf("参数错误\n");

return 0;

}

ktv\_start( argv[1][1], argv[2] );

return 0;

}

（2）ktv.h

#ifndef \_\_KTV\_H

#define \_\_KTV\_H

#include<stdio.h>

#include<string.h>

#include<stdbool.h>

/\* 管理员操作的相关模块 \*/

/\*

\* 函数名称：admin\_add\_song

\* 函数功能：为ktv添加歌曲

\* 参数: 无

\* 返回值：void

\*/

void admin\_add\_song(){

FILE \*songs\_list = fopen("/home/yyang/MyProgram/c\_lab/src/ktv/songs", "r");

if( !songs\_list ){

printf( "打开歌单出错，目前无法进行添加操作\n");

return;

}// 打开现有歌曲列表

printf("\n=================添加歌曲=================\n");

char name[512];// 输入的歌曲名

char singer[512];// 输入的作者名

char cur\_name[512];// 现在遍历到的歌曲名

char cur\_singer[512];// 现在遍历到的作者名

bool is\_existed = false;// 表示该作者的该歌曲已经在系统中

printf("请输入你要添加的歌曲的名称\n");

scanf( "%s", name );

printf("请输入该歌曲的歌手名称\n");

scanf( "%s", singer );

while( fscanf(songs\_list, "%s", cur\_name) != EOF && fscanf(songs\_list, "%s", cur\_singer )){

// 与现存歌曲列表中的结果逐一比对

// 若找到对应的歌曲, 则设置存在标志并退出比较

if( strcmp( cur\_singer, singer ) == 0 && strcmp( cur\_name, name ) == 0 ){

is\_existed = true;

break;

}

}

if( is\_existed ){// 如果已经存在

printf("该歌手的该歌曲已经存在曲库之中，无法重复添加\n");

fclose( songs\_list );

return;

}

fclose( songs\_list );

songs\_list = fopen("/home/yyang/MyProgram/c\_lab/src/ktv/songs", "a");

// 以附加写入方式重新打开文件, 准备添加歌曲

if( !songs\_list ){

printf( "打开歌单出错，目前无法进行添加操作\n");

return;

}

char directory[1024]="/home/yyang/MyProgram/c\_lab/src/ktv/";

size\_t pre\_len = strlen( directory );

strcpy( directory+pre\_len, singer );

\*(directory+pre\_len+strlen(singer) ) = '\_';

strcpy( directory+pre\_len+strlen(singer)+1, name );

// 新添加的歌曲的歌词文件所存放的路径名

//printf("%s\n", directory);

FILE \*song = fopen(directory, "w+");

if( !song ){

printf( "创建歌曲出错，目前无法进行添加操作\n");

fclose( songs\_list );

return;

}// 打开新歌曲的歌词文件

printf( "请输入这首歌的歌词，歌词以由仅含有一个\'#\'的新一行为结束标志\n");

char txt[512];

while( scanf("%s",txt) && txt[0] != '#' ){

fprintf(song, "%s\n", txt );

}

fprintf(songs\_list, "%s %s\n", name, singer );

printf( "歌曲添加成功\n" );

fclose(songs\_list);

fclose( song );

return;

}

/\*

\* 函数名称：admin\_delete\_song

\* 函数功能：为ktv删除歌曲

\* 参数: 无

\* 返回值：void

\*/

void admin\_delete\_song(){

FILE \*songs\_list = fopen("/home/yyang/MyProgram/c\_lab/src/ktv/songs", "r");

// 未执行删除时的歌曲列表

FILE \*new\_song\_list = fopen("/home/yyang/MyProgram/c\_lab/src/ktv/songs.temp", "w");

// 执行删除后的歌曲列表

if( !songs\_list || !new\_song\_list ){

printf( "打开歌单出错，目前无法进行删除操作\n");

return;

}

printf("\n=================删除歌曲=================\n");

char name[512];// 输入的歌曲名

char singer[512];// 输入的作者名

char cur\_name[512];// 现在遍历到的歌曲名

char cur\_singer[512];// 现在遍历到的作者名

bool is\_existed = false;// 表示该作者的该歌曲已经在系统中

printf("请输入你要删除的歌曲的名称\n");

scanf( "%s", name );

printf("请输入该歌曲的歌手名称\n");

scanf( "%s", singer );

while( fscanf(songs\_list, "%s", cur\_name) != EOF && fscanf(songs\_list, "%s", cur\_singer )){

// 与现存歌曲列表中的结果逐一比对

// 若找到对应的歌曲, 则不写入新的歌曲列表

// 否则,写入新的歌曲列表

if( strcmp( cur\_singer, singer ) == 0 && strcmp( cur\_name, name ) == 0 ){

continue;

}

fprintf(new\_song\_list, "%s %s\n", cur\_name, cur\_singer );

}

char directory[1024]="/home/yyang/MyProgram/c\_lab/src/ktv/";

size\_t pre\_len = strlen( directory );

strcpy( directory+pre\_len, singer );

\*(directory+pre\_len+strlen(singer) ) = '\_';

strcpy( directory+pre\_len+strlen(singer)+1, name );

// 删除的歌曲的歌词文件所存放的路径名

fclose( songs\_list );

fclose( new\_song\_list );

remove( "/home/yyang/MyProgram/c\_lab/src/ktv/songs" );// 移除旧歌曲列表

remove( directory );// 移除要删除的歌曲文件的歌词文件

rename( "/home/yyang/MyProgram/c\_lab/src/ktv/songs.temp", "/home/yyang/MyProgram/c\_lab/src/ktv/songs" );

// 重命名新的歌曲列表的文件名

printf("歌曲删除成功\n");

}

/\*

\* 函数名称：admin

\* 函数功能：打开管理员模式

\* 参数: 无

\* 返回值：void

\*/

void admin( ){

int choice;// 表示用户所选操作

while(1){

printf("\n================管理员模式================\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*功能列表\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("1.添加歌曲\n");

printf("2.删除歌曲\n");

printf("3.退出系统\n");

printf("请输入对应序号选择功能，按回车键确认选择\n");

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

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

scanf( "%d", &choice );

switch( choice ){

case 1: admin\_add\_song(); break;

case 2: admin\_delete\_song(); break;

case 3: printf("管理员操作结束\n"); return; break;

default: printf("输入选项不存在，请核对后重新选择\n"); break;

}

}

}

/\* 用户操作的相关模块 \*/

/\*

\* 函数名称：user\_find\_song\_singer

\* 函数功能：查找某一歌手的所演唱的歌曲

\* 参数: 无

\* 返回值：void

\*/

void user\_find\_song\_singer(){

FILE \*songs\_list = fopen("/home/yyang/MyProgram/c\_lab/src/ktv/songs", "r");

if( !songs\_list ){

printf( "打开歌单出错，目前无法进行查询操作\n");

return;

}

printf("\n=================查找歌曲=================\n");

char singer[512];// 输入的作者名

char cur\_name[512];// 现在遍历到的歌曲名

char cur\_singer[512];// 现在遍历到的作者名

printf("请输入你要查询的歌手的名称\n");

scanf( "%s", singer );

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*查询结果\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

while( fscanf(songs\_list, "%s", cur\_name) != EOF && fscanf(songs\_list, "%s", cur\_singer )){

// 与现存歌曲列表中的结果逐一比对

// 若找到对应歌手的歌曲, 则予以显示

// 否则, 不显示

if( strcmp( cur\_singer, singer ) == 0 ){

printf("%s - %s\n", cur\_name, cur\_singer );

}

}

printf("查询结束\n");

fclose( songs\_list );

}

/\*

\* 函数名称：user\_find\_song\_singer

\* 函数功能：查找同名的所有歌曲

\* 参数: 无

\* 返回值：void

\*/

void user\_find\_song\_name(){

FILE \*songs\_list = fopen("/home/yyang/MyProgram/c\_lab/src/ktv/songs", "r");

if( !songs\_list ){

printf( "打开歌单出错，目前无法进行查询操作\n");

return;

}

printf("\n=================查找歌曲=================\n");

char name[512];// 输入的歌曲

char cur\_name[512];// 现在遍历到的歌曲名

char cur\_singer[512];// 现在遍历到的作者名

printf("请输入你要查询的歌曲的名称\n");

scanf( "%s", name );

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*查询结果\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

while( fscanf(songs\_list, "%s", cur\_name) != EOF && fscanf(songs\_list, "%s", cur\_singer )){

// 与现存歌曲列表中的结果逐一比对

// 若找到对应歌名的歌曲, 则予以显示

// 否则, 不显示

if( strcmp( cur\_name, name ) == 0 ){

printf("%s - %s\n", cur\_name, cur\_singer );

}

}

printf("查询结束\n");

fclose( songs\_list );

}

/\*

\* 函数名称：user\_find\_song

\* 函数功能：用户查找歌曲选项的二级界面

\* 参数: 无

\* 返回值：void

\*/

void user\_find\_song(){

int choice;

while(1){

printf("\n================查找歌曲================\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*功能列表\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("1.按歌名查找\n");

printf("2.按歌手查找\n");

printf("3.返回上级\n");

printf("请输入对应序号选择功能，按回车键确认选择\n");

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

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

scanf( "%d", &choice );

switch( choice ){

case 1: user\_find\_song\_name(); break;

case 2: user\_find\_song\_singer(); break;

case 3: printf("正在返回上一级\n"); return; break;

default: printf("输入选项不存在，请核对后重新选择\n"); break;

}

}

}

/\*

\* 函数名称：user\_play\_song

\* 函数功能：播放歌曲

\* 参数: 无

\* 返回值：void

\*/

void user\_play\_song(){

FILE \*songs\_list = fopen("/home/yyang/MyProgram/c\_lab/src/ktv/songs", "r");

if( !songs\_list ){

printf( "打开歌单出错，目前无法进行点播操作\n");

return;

}

printf("\n=================点播歌曲=================\n");

char name[512];// 输入的歌曲名

char singer[512];// 输入的作者名

char cur\_name[512];// 现在遍历到的歌曲名

char cur\_singer[512];// 现在遍历到的作者名

printf("请输入你要点播的歌曲的名称\n");

scanf( "%s", name );

printf("请输入该歌曲的歌手名称\n");

scanf( "%s", singer );

bool is\_existed = false;// 是否存在对应的歌曲

while( fscanf(songs\_list, "%s", cur\_name) != EOF && fscanf(songs\_list, "%s", cur\_singer )){

// 与现存歌曲列表中的结果逐一比对

// 若找到对应歌手的歌曲, 则设置存在标志

if( strcmp( cur\_singer, singer ) == 0 && strcmp( cur\_name, name ) == 0 ){

is\_existed = true;

break;

}

}

if( !is\_existed ){

printf("所点歌曲不在曲库中，点歌失败\n");

fclose(songs\_list);

return;

}

char directory[1024]="/home/yyang/MyProgram/c\_lab/src/ktv/";

size\_t pre\_len = strlen( directory );

strcpy( directory+pre\_len, singer );

\*(directory+pre\_len+strlen(singer) ) = '\_';

strcpy( directory+pre\_len+strlen(singer)+1, name );

// 要播放的歌曲的歌词文件所存放的路径名

FILE \*song = fopen(directory, "r");

if( !song ){

printf( "无法打开该歌曲歌词，点歌失败\n");

fclose( songs\_list );

return;

}

char buf[512];

while( fgets(buf, 512,song ) != NULL ){

printf( "%s", buf );

}// 逐行显示歌词

fclose( songs\_list );

fclose( song );

printf("点歌成功\n");

}

/\*

\* 函数名称：user

\* 函数功能：打开用户模式

\* 参数: 无

\* 返回值：void

\*/

void user( ){

int choice;

while(1){

printf("\n================用户模式================\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*功能列表\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("1.查找歌曲\n");

printf("2.点播歌曲\n");

printf("3.退出系统\n");

printf("请输入对应序号选择功能，按回车键确认选择\n");

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

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

scanf( "%d", &choice );

switch( choice ){

case 1: user\_find\_song(); break;

case 2: user\_play\_song(); break;

case 3: printf("用户操作结束\n"); return; break;

default: printf("输入选项不存在，请核对后重新选择\n"); break;

}

}

}

/\* 开始模块 \*/

/\*

\* 函数名称：ktv\_start

\* 函数功能：开始使用ktv

\* 参数: char user\_level( 操作者的身份 ), const char \*name( 操作者的姓名 )

\* 返回值：void

\*/

void ktv\_start( char user\_level, const char \*name ){

if( user\_level == 'a' ){// 管理员模式

FILE \*admin\_list = fopen( "/home/yyang/MyProgram/c\_lab/src/ktv/admin", "r" );

if( !admin\_list ){

printf("打开管理员列表失败，无法进行验证，程序结束运行。\n");

return;

}// 打开管理员名称列表, 准备进行验证

char buf[512];// 临时缓冲区

while( fscanf(admin\_list,"%s", buf) != EOF ){

// 与管理员列表中的名称逐一比对

// 若发现该名管理员, 则开启管理员界面, 操作结束后退出系统

if( strcmp( buf, name ) == 0 ){

printf("验证通过，正在进入系统\n");

admin();// 打开管理员操作界面

printf("已退出系统，欢迎下次使用。\n");

fclose(admin\_list);

return;

}

}

printf("该管理员未登记入系统中，请核对名称后重试。\n");

fclose(admin\_list);

}

else if( user\_level == 'u' ){// 用户模式

FILE \*user\_list = fopen( "/home/yyang/MyProgram/c\_lab/src/ktv/user", "r" );

if( !user\_list ){

printf("打开用户列表失败，无法进行验证，程序结束运行。\n");

return;

}// 打开用户名称列表, 准备进行验证

char buf[512];

while( fscanf(user\_list,"%s", buf) != EOF ){

// 与用户列表中的名称逐一比对

// 若发现该名用户, 则开启用户界面, 操作结束后退出系统

bool flag = strcmp( buf, name ) == 0;

if( flag ){

printf("验证通过，正在进入系统\n");

user();// 打开用户操作界面

printf("已退出系统，欢迎下次使用。\n");

fclose(user\_list);

return;

}

}

fclose(user\_list);

printf("该用户未登记入系统中，请核对名称后重试。\n");

}

else{// 若非以上两者, 说明存在错误

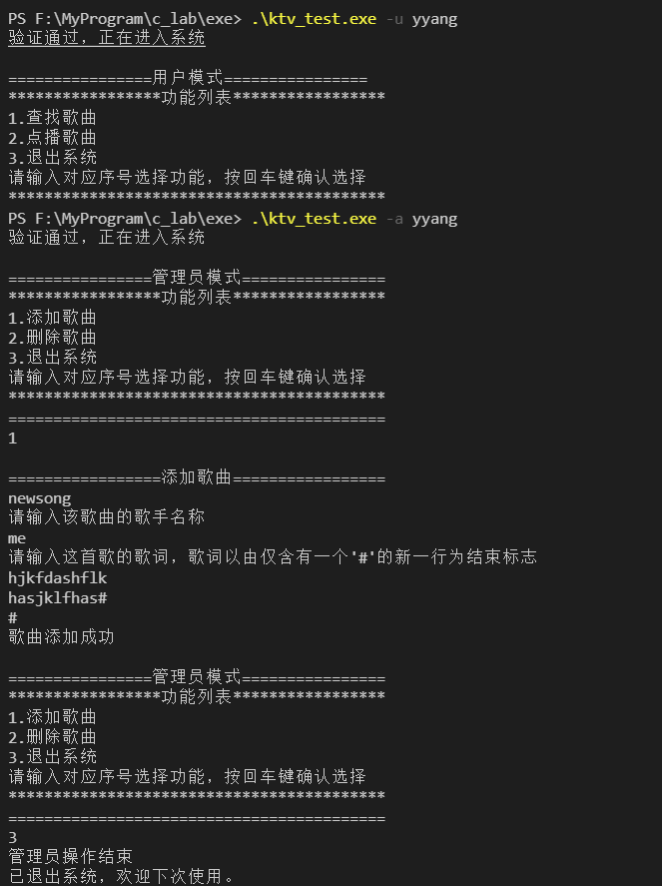
printf("用户类型错误，请核对输入后重新启动程序。\n");

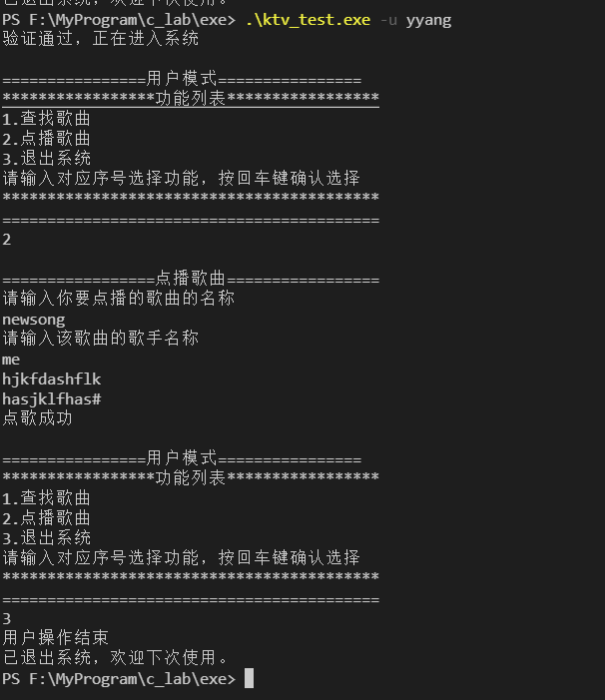
}

return;

}

#endif

**6. 测试数据（输入、输出）：**



**7. 小结：**

本次实验圆满完成，实验结果符合实验要求。

实验过程中，我复习到了c语言中文件读写、内存管理、结构体定义、基本数据结构、高级格式化输入输出。