实验5 文件系统

16281035 陈琦 计科1601

一、实验简介

本实验要求在模拟的I/O系统之上开发一个简单的文件系统。用户通过create,open, read等命令与文件系统交互。文件系统把磁盘视为顺序编号的逻辑块序列，逻辑块的编号为0至*L*−1。I/O系统利用内存中的数组模拟磁盘。

二、基于交互的主要函数设计

1. 登录选项操作函数

```
int LoginDisplay() //登陆选项操作函数
{
    int SELETE_1 = 0;
    do
    {
    cout<<"1、用户登陆  2、用户注册  0、退出"<<endl;
    cin>>SELETE_1;
    }while(SELETE_1<0 || SELETE_1>2);
    system("cls");
    return SELETE_1;
```

2. 用户登录及注册函数

```
bool Login(int SELETE)//用户登陆，注册函数
{
    FILE *fp,*fp1,*fp2;
    char name[12];
    switch(SELETE)
    {
    case 1://用户登陆
        if((fp = fopen("LOGIN.exe","rb")) == NULL)//打开用户注册目录管理文件
        {
            cout<<"\n错误：不能打开登陆文件。"<<endl;
            getch();system("cls");
            return false;
        }
        curuser = getspace(MFD);
        cout<<"\n*****登陆*****\n用户名：";
        cin>>name; //输入用户登陆名

        while(!feof(fp)) //检查该用户是否合法
        {
            fread(curuser,sizeof(MFD),1,fp);
            if(strcmp(curuser->username,name)==0)
                break;
        }
        if(feof(fp)) //如果没有找到跟当前登陆用户名相同的管理信息，提示出错
        {
            cout<<"\n错误：该用户不存在。"<<endl;
            fclose(fp);
            return false;
        }
        else
        {
            fclose(fp);
            return true;
        }
        break;

    case 2:  //新用户注册
        if((fp=fopen("LOGIN.exe","ab"))==NULL)//如果登陆信息管理文件不存在
            fp=fopen("LOGIN.exe","wb+");      //创建该信息管理文件
        char name[12];
        curuser = getspace(MFD);
        while(1)
        {
            cout<<"*****新用户注册*****"<<endl;
            cout<<"用户名：";
            cin>>name;       //输入用户注册名
            fp1 = fopen("LOGIN.exe","rb");
            while(!feof(fp1))//查看该用户名是否被别的用户占用
            {
                fread(curuser,sizeof(MFD),1,fp1);
                if(strcmp(curuser->username,name) == 0)//该名称已经被使用
```

```
                {
                    cout<<"该用户已经存在，请重新输入！"<<endl;
                    getch();
                    break;
                }
            }
            if(feof(fp1))//该名称没有被别的用户占用
            {
                strcpy(curuser->username,name);
                curuser->filepoint = NULL;
                fwrite(curuser,sizeof(MFD),1,fp);
                strcpy(user,curuser->username);//生成用户文件管理模块
                strcat(user,".exe");            //用于管理用户目录下的各个文件
                fp2=fopen(user,"wb+");
                fclose(fp2);
                cout<<"注册成功！"<<endl;   //提示注册成功
                fclose(fp1);
                fclose(fp);
                break;
            }
        }
        fp = fopen("LOGIN.exe","rb"); //显示当前注册用户的名称
        while(1)
        {
            fread(curuser,sizeof(MFD),1,fp);
            if(feof(fp))
                break;
            cout<<curuser->username<<endl;
            getch();
        }
        fclose(fp);
        return true;
        break;
    default:
        return false;
        break;
    }
}
```

3. 注销函数

```
void ByeFile(bool BOOL)//注销函数，调用次函数用户可以退出系统
{
    FILE *infile,*outfile;
    char out[50];
    strcpy(out,"outfilelocate.exe");
    if((infile=fopen("LOGIN.exe","rb"))==NULL)
    {
        cout<<"\n保存错误。";    //  fclose(infile);
        return;
    }
    else
```

```
    {
        if((outfile=fopen(out,"wb+"))==NULL)//申请一个缓冲区管理模块
            //存放用户更新后的全部信息
        {
            cout<<"\n保存错误。";// fclose(outfile);
            fclose(infile);return;
        }
        else
        {
            MFD *mfd = getspace(MFD);
            while(1)
            {//将旧文件管理信息读出，并保存到新的文件信息管理模块中
                fread(mfd,sizeof(MFD),1,infile);
                if(feof(infile))
                    break;
                if((strcmp(mfd->username,curuser->username))==0)
                {
                    if(BOOL)//更新当前用户信息的操作
                        fwrite(curuser,sizeof(MFD),1,outfile);
                    else continue;//如果用户想把自己的注册目录从系统中彻底删除
                    //则执行该操作
                }
                else
                    fwrite(mfd,sizeof(MFD),1,outfile);//写入新的模块
            }
            fclose(infile);fclose(outfile);
            remove("LOGIN.exe");//将旧的该用户的文件管理模块删除
            rename(out,"LOGIN.exe");//将新的用户的文件管理模块重命名为用户目录下的管理模块
        }
    }
    system("cls");
    main();
}
```

三、基于文件的主要函数设计

我设计的文件系统采用二级文件目录。设置主目录（MFD）和用户文件目录(UFD)，分别以文件的方式保存在磁盘中。在主目录中又注册用户的用户名和另一标志该用户目录下是否有文件的指针标记。用户文件目录用用户名作为文件名保存在磁盘，以便检索时方便对应。在用户文件目录中保存着该目录下所有的文件的文件名称、保护码、文件长度。

我设计的系统是一个多用户、多任务的实时操作系统。

对用户和用户的文件数目并没有上限。也就是说该系统允许任何用户申请空间，而且在其目录下的文件数目并不做任何的限制。

该系统的操作命令如下：

1. 1. bye-用户注销命令。当使用该命令时，用户退出系统。命令格式：
   2. run\bye↙系统注销该用户并回到登陆界面。

2. 1. close-删除用户注册信息命令。执行该命令后，用户在系统中的所有信息，包括该用户目录下的所有文件都被删除。命令格式：run\close↙.完成后返回登陆界面。
3. 1. create-在当前目录下创建一个文件，且该文件不能跟系统中的文件重名。该文件的管理信息登录到用户文件信息管理模块中。命令格式：run\create>file1↙。其中file1为要创建的文件名称。执行完该命令后回到执行命令行。
   2. delete-删除当前用户目录下的一个文件。命令格式：run\delete>file1↙。返回命令行。
   3. list-显示当前注册目录下的所有文件信息，包括文件名、文件长度、文件操作权限。命令格式：run\list↙。
   4. chmod-改变某个文件的执行权限，但前提是该文件是该用户目录下的文件。命令格式：run\chmod>file1↙。
   5. open-在window界面下打开某个文件。命令格：run\open>file1↙。执行该命令后，文件file1将用在windows界面下的文件形式打开。用户可以在这个方式中对文件进行修改，并将修改后的内容保存。
   6. read-读文件信息。将文件信息读入并显示在终端。命令格式：run\read>file1↙。
   7. write-向某个文件写入新的信息。用户可以选择用覆盖原来内容的方式和在文件的末尾插入新信息的方式写入信息。

1. 创建 create(filename)

根据指定的文件名创建新文件。

```
void CreatFile()//在当前用户目录下创建文件
{
    FILE *fp;
    curuser->filepoint=true;

    if((fp=fopen(buffer,"r"))==NULL)//如果没有跟用户输入文件名相同的文件
    {
        if((fp=fopen(buffer,"w"))==NULL)
        {
            cout<<"\n创建文件失败！";
        //  fclose(fp);
            return;
        }
        fclose(fp);
    }
    else
    {//用户要创建的文件已经存在
        cout<<"\n该文件已经存在，创建另一个文件？Y/N";
        char ch;
        cin>>ch;
        switch(ch)
        {
        case 'Y':
        case 'y':
            cout<<"\n输入新文件名：";
            cin>>buffer;
            strcat(buffer,".txt");
            fclose(fp);
            if((fp=fopen(buffer,"w"))==NULL)
            {
                cout<<"\n创建文件失败！";
```

```
            //  fclose(fp);
                return;
            }
            fclose(fp);
            break;
        default:
            fclose(fp);
            return;
        }
    }
    strcpy(user,curuser->username);
    strcat(user,".exe");
    curfile = getspace(UFD);
    strcpy(curfile->filename,buffer);//文件名
    curfile->length=0; //该文件长度为零
    curfile->safecode=30; //设置该文件的默认权限
        //11 00，文件主有读和写权，其他用户没有读写权
    if((fp=fopen(user,"ab"))==NULL)
    {
        cout<<"\n错误：你可能不是合法用户。"<<endl;
        getch();
    }
    else
    {
        fwrite(curfile,sizeof(UFD),1,fp);//将该文件信息写入用户文件信息管理模块中
        cout<<"\n文件 "<<curfile->filename<<" 创建成功！";
    }
    fclose(fp);
}
```

2. 删除 delete(filename)

删除指定文件

```
void DeleteFile()//删除当前目录下一个文件的操作
{
    char ch;
    FILE *infile,*outfile;
    cout<<"\n确定要删除文件："<<buffer<<" Y/N"<<endl;
    cin>>ch;//提示用户确认删除
    switch(ch)
    {
    case 'Y':
    case 'y'://更新用户文件信息管理模块，这里同样使用缓冲区模块来更新
            //方法与上面将到的类似
            char out[50],in[50];
            strcpy(out,"outfilelocate.exe");
            strcpy(in,curuser->username);
            strcat(in,".exe");
            if((infile=fopen(in,"rb"))==NULL)//打开该用户的文件信息管理模块
            {
                cout<<"\n保存错误。";
                //fclose(infile);
```

```
                return;
            }
            else
            {
                if((outfile=fopen(out,"wb+"))==NULL)
                {
                    cout<<"\n保存错误。";// fclose(outfile);
                    fclose(infile);return;
                }
                else
                {
                    UFD *ufd = getspace(UFD);
                    while(1)
                    {
                        fread(ufd,sizeof(UFD),1,infile);//从旧模块读出信息
                        if(feof(infile))
                            break;
                        if((strcmp(ufd->filename,buffer))==0)//要进行更新的信息
                            continue;
                        else
                            fwrite(ufd,sizeof(UFD),1,outfile);//写入新模块
                    }
                    fclose(infile);fclose(outfile);
                    remove(in);//在磁盘移除就模块
                    rename(out,in); //新模块命名为当前用户文件信息管理模块
                }
            }
        remove(buffer);//从磁盘中删除该文件
        break;
    default:
        break;
    }
}
```

3. 打开 open(filename)

打开文件。该函数返回的索引号可用于后续的read, write, lseek,或close操作。

```
void OpenFile()//在window模式下打开该文件
{
    system(buffer);//buffer为文件名，如:file1.txt
}
```

4. 读 read(index, mem_area, count)

从指定文件顺序读入*count*个字节*mem**a**rea*指定的内存位置。读操作从文件的读写指针指示的位置开始。

```
oid ReadFile()//读文件函数
{
    if(!QueryMod(false))//查询当前用户是否有权读该文件
        return;//没有读权，则返回
```

```
    FILE *fp;
    if((fp=fopen(buffer,"r"))==NULL)//打开该文件
    {
        cout<<buffer;
        cout<<"\n该文件不存在。";
        return;
    }
    else{
        char ch;
        ch=fgetc(fp);
        while(ch!=EOF)//将该文件信息逐一输出到终端
        {
            putchar(ch);
            ch=fgetc(fp);
        }
        cout<<endl;
    }
    fclose(fp);
}
```

5. 写 write(index, mem_area, count)

把*mem**a**rea*指定的内存位置开始的*count*个字节顺序写入指定文件。写操作从文件的读写指针指示的位置开始。

```
void WriteFile()//向文件写入信息的操作
{
    if(!QueryMod(true))//查询当前用户对该文件是否有写权
        return;//对该文件没有写权则返回
    char ch;
    int i=0;
    FILE *fp;
    if((fp=fopen(buffer,"r"))==NULL)//查询该文件是否存在
    {
        cout<<"\n该文件不存在，请创建该文件后再写入。";
    //  fclose(fp);
        return;
    }
    fclose(fp);
    cout<<"\n请选择写入方式："<<endl;
    cout<<"  1、覆盖原文件     2、在原文件末尾写入     3、取消"<<endl;
    cin>>ch;
    cout<<"开始输入正文："<<endl;
    switch(ch)
    {
    case '1'://覆盖原文件
        if((fp=fopen(buffer,"w"))==NULL)
            cout<<"\n文件打开失败。";
        else
        {
            ch=getchar();
            while(ch!='#')//将新的文件内容写入到文件的磁盘位置中
            {
```

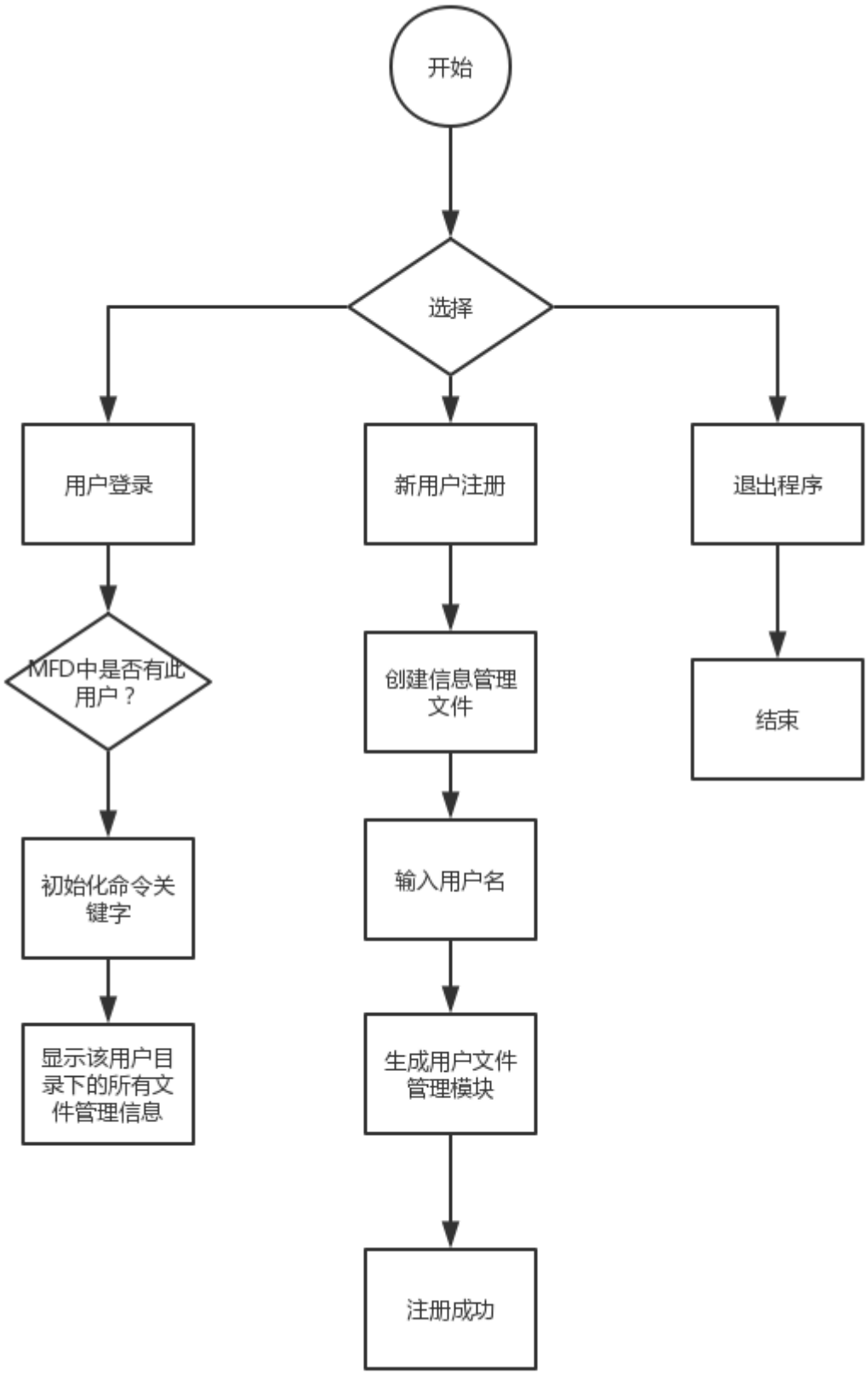
```
                i++;
                fputc(ch,fp);
                ch=getchar();
            }
        }
        fclose(fp);
        WriteLengthToFile(i,false);//将文件长度写入文件管理模块
        break;
    case '2':
        if((fp=fopen(buffer,"a"))==NULL)
            cout<<"\n文件打开失败。";
        else
        {
            ch=getchar();
            while(ch!='#')//将新的文件内容写入到文件的磁盘位置中
            {
                i++;
                fputc(ch,fp);
                ch=getchar();
            }
        }
        fclose(fp);
        WriteLengthToFile(i,true);//将文件长度写入文件管理模块
        break;
    default:
        break;
    }
}
```

6. 查看目录 ListAllFile()

列表显示所有文件及其长度。

```
void ListAllFile()//显示当前用户目录下的文件信息
{
    DisplayUFD();
}
```

四、实验流程图


开始
选择
用户登录
新用户注册
退出程序
MFD中是否有此用户？
创建信息管理文件
结束
初始化命令关键字
输入用户名
显示该用户目录下的所有文件管理信息
生成用户文件管理模块
注册成功

五、遇到的问题

1. 头文件编译出错问题

有时候我们在编写程序，特别是多个程序时我们需要写自己的头文件，这时我们用gcc编译时会出错，因为编译器不会自动的帮你添加头文件，此时我们需要手动添加自己写的头文件。

如果手动添加不成功，这时候我们就需要-I+.h文件的路径。如.h 在/root/code下的话，gcc -I/root/code main.c -o main（.h文件对应的C文件，-o mian 表示输出mian可执行文件。

有时候可能因为gcc版本问题还是编译不了的就把.h文件添加到gcc默认的.h文件路径中去。一般.h文件会在/usr/include中，还可以用find或者grep查看。

2. 在linux中使用getch()函数

由于在linux中没有conio.h文件，所以不能直接用getch()函数。如果不想自己写conio.h文件，则可以用以下方法：

在linux中并没有 conio.h 这个文件,要实现类似 getch()/getche() 等函数的功能，可以使用 curses库。

#include <curses.h>

使用 curses 之前要先进行初始化，用完了要注消————这些操作分别调用 initscr() endwin() 来完成.

main(){ initscr(); . . . endwin(); }

注：在编译的时候如果编译不过，可以试着添加 -lcurses 参数来引入 curses 库

六、源代码

```
#include<iostream>
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<stdbool.h>
#include <curses.h>
//#include<conio.h>
//#include<dos.h>
#define keynum 10
#define getspace(type) (type*)malloc(sizeof(type))
using namespace std;

char cmd[64];  //存放用户输入命令
char buffer[36];//
char user[32];//存放当前登陆的用户名

typedef char ALFA[12];
ALFA KWORD[keynum];
struct UFD{//用户文件管理模块
    char filename[32]; //文件名
    int  safecode;     //文件保护码
    long length;       //文件长度
}*curfile = NULL;

struct MFD{//用户登陆信息管理模块
    char username[32]; //用户名
    bool filepoint;    //用户目录下的文件指针，false表示目录为空
```

```
}*curuser = NULL,*elseuser=NULL;
typedef UFD UFD;
typedef MFD MFD;
int main();

void KeyWord()//初始化命令关键字
{
    strcpy(KWORD[ 1],"bye");    strcpy(KWORD[ 2],"chmod");
    strcpy(KWORD[ 3],"close");  strcpy(KWORD[ 4],"create");
    strcpy(KWORD[ 5],"delete"); strcpy(KWORD[ 6],"list");
    strcpy(KWORD[ 7],"open");   strcpy(KWORD[ 8],"read");
    strcpy(KWORD[ 9],"write");

}
int LoginDisplay() //登陆选项操作函数
{
    int SELETE_1 = 0;
    do
    {
    cout<<"1、用户登陆  2、用户注册  0、退出"<<endl;
    cin>>SELETE_1;
    }while(SELETE_1<0 || SELETE_1>2);
    system("cls");
    return SELETE_1;
}

bool Login(int SELETE)//用户登陆，注册函数
{
    FILE *fp,*fp1,*fp2;
    char name[12];
    switch(SELETE)
    {
    case 1://用户登陆
        if((fp = fopen("LOGIN.exe","rb")) == NULL)//打开用户注册目录管理文件
        {
            cout<<"\n错误：不能打开登陆文件。"<<endl;
            getch();system("cls");
            return false;
        }
        curuser = getspace(MFD);
        cout<<"\n*****登陆*****\n用户名：";
        cin>>name; //输入用户登陆名

        while(!feof(fp)) //检查该用户是否合法
        {
            fread(curuser,sizeof(MFD),1,fp);
            if(strcmp(curuser->username,name)==0)
                break;
        }
        if(feof(fp)) //如果没有找到跟当前登陆用户名相同的管理信息，提示出错
        {
            cout<<"\n错误：该用户不存在。"<<endl;
            fclose(fp);
```

```
            return false;
        }
        else
        {
            fclose(fp);
            return true;
        }
        break;

    case 2:  //新用户注册
        if((fp=fopen("LOGIN.exe","ab"))==NULL)//如果登陆信息管理文件不存在
            fp=fopen("LOGIN.exe","wb+");      //创建该信息管理文件
        char name[12];
        curuser = getspace(MFD);
        while(1)
        {
            cout<<"*****新用户注册*****"<<endl;
            cout<<"用户名：";
            cin>>name;       //输入用户注册名
            fp1 = fopen("LOGIN.exe","rb");
            while(!feof(fp1))//查看该用户名是否被别的用户占用
            {
                fread(curuser,sizeof(MFD),1,fp1);
                if(strcmp(curuser->username,name) == 0)//该名称已经被使用
                {
                    cout<<"该用户已经存在，请重新输入！"<<endl;
                    getch();
                    break;
                }
            }
            if(feof(fp1))//该名称没有被别的用户占用
            {
                strcpy(curuser->username,name);
                curuser->filepoint = NULL;
                fwrite(curuser,sizeof(MFD),1,fp);
                strcpy(user,curuser->username);//生成用户文件管理模块
                strcat(user,".exe");           //用于管理用户目录下的各个文件
                fp2=fopen(user,"wb+");
                fclose(fp2);
                cout<<"注册成功！"<<endl;   //提示注册成功
                fclose(fp1);
                fclose(fp);
                break;
            }
        }
        fp = fopen("LOGIN.exe","rb"); //显示当前注册用户的名称
        while(1)
        {
            fread(curuser,sizeof(MFD),1,fp);
            if(feof(fp))
                break;
            cout<<curuser->username<<endl;
            getch();
```

```
        }
        fclose(fp);
        return true;
        break;
    default:
        return false;
        break;
    }
}

void DisplayUFD()//打印用户信息，包括用户的各个文件
                 //名称、长度和操作权限的设置信息
{
    if(curuser->filepoint == false)//当前用户目录下没有任何文件存在
        cout<<"\n用户 "<<curuser->username<<" 文件夹是空的"<<endl;
    else
    {//存在文件，将所有文件信息打印在终端
        FILE *fp;
        char filename[12];
        strcpy(filename,curuser->username);
        strcat(filename,".exe");
        if((fp=fopen(filename,"rb"))==NULL)//打开用户文件信息管理模块
        {
            cout<<"\n无法打开用户："<<curuser->username<<" 的文件！"<<endl;
            getch();
            return;
        }
        else
        {//读入并将用户全部文件信息打印在终端
            cout<<"用户："<<curuser->username<<"目录下的文件："<<endl;
            UFD *ufd;
            int i=0;
            ufd = getspace(UFD); //申请存放用户文件模块的空间
            while(1)
            {
                fread(ufd,sizeof(UFD),1,fp);
                if(feof(fp))//全部输出完毕，结束
                    break;
                else//打印信息
                    cout<<ufd->filename<<"\t"<<ufd->length<<"\t"<<ufd->safecode<<endl;
            }
        }
        fclose(fp);
    }
}

void ByeFile(bool BOOL)//注销函数，调用次函数用户可以退出系统
{
    FILE *infile,*outfile;
    char out[50];
    strcpy(out,"outfilelocate.exe");
    if((infile=fopen("LOGIN.exe","rb"))==NULL)
    {
```

```
        cout<<"\n保存错误。";    //  fclose(infile);
        return;
    }
    else
    {
        if((outfile=fopen(out,"wb+"))==NULL)//申请一个缓冲区管理模块
            //存放用户更新后的全部信息
        {
            cout<<"\n保存错误。";// fclose(outfile);
            fclose(infile);return;
        }
        else
        {
            MFD *mfd = getspace(MFD);
            while(1)
            {//将旧文件管理信息读出，并保存到新的文件信息管理模块中
                fread(mfd,sizeof(MFD),1,infile);
                if(feof(infile))
                    break;
                if((strcmp(mfd->username,curuser->username))==0)
                {
                    if(BOOL)//更新当前用户信息的操作
                        fwrite(curuser,sizeof(MFD),1,outfile);
                    else continue;//如果用户想把自己的注册目录从系统中彻底删除
                    //则执行该操作
                }
                else
                    fwrite(mfd,sizeof(MFD),1,outfile);//写入新的模块
            }
            fclose(infile);fclose(outfile);
            remove("LOGIN.exe");//将旧的该用户的文件管理模块删除
            rename(out,"LOGIN.exe");//将新的用户的文件管理模块重命名为用户目录下的管理模块
        }
    }
    system("cls");
    main();
}

bool ClearUserFile()//用户要将自己的注册目录从系统彻底删除
//首先将该用户目录下的全部文件删除
{
    FILE *fp;
    char file[50];
    strcpy(file,curuser->username);
    strcat(file,".exe");
    if((fp=fopen(file,"rb"))==NULL) //打开用户文件信息管理模块
    {
    //  fclose(fp);
        cout<<"\n操作失败。";return true;
    }
    else
    {//将该用户目录下的文件逐个从磁盘删除
        UFD *ufd = getspace(UFD);
```

```
        while(1)
        {
            fread(ufd,sizeof(UFD),1,fp);
            if(feof(fp))
                break;
            else
                remove(ufd->filename);//删除文件
        }
        fclose(fp);
        return true;
    }
}

void ClearUserMes()//删除用户全部信息
{
    char name[50];
    strcpy(name,curuser->username);
    strcat(name,".exe");
    remove(name); //从磁盘中删除用户文件信息管理模块
    ByeFile(false);//更新系统的用户登陆信息管理模块
}
void DeleteUser()//删除用户注册目录的操作
{
    char ch;
cout<<"\n该操作将会是你在系统所有信息删除，下次登陆时你必须重新申请用户名！"<<endl;
    cout<<"\n你确定要删除你在系统中的注册信息吗？Y/N"<<endl;
    cin>>ch;
    switch(ch)//提示用户确认删除
    {
    case 'Y':
        if(ClearUserFile())//如果用户的全部文件已经删除了
            //则可以将该用户的文件信息管理模块也从磁盘中删除
            //以免在没完全删除文件却删了该文件信息管理模块
            //使得这些文件无法再进行管理造成磁盘空间的浪费
            ClearUserMes();//删除文件信息管理模块
        break;
    default:
        cout<<"\n你取消了此操作！";
        break;
    }
}

void CreatFile()//在当前用户目录下创建文件
{
    FILE *fp;
    curuser->filepoint=true;

    if((fp=fopen(buffer,"r"))==NULL)//如果没有跟用户输入文件名相同的文件
    {
        if((fp=fopen(buffer,"w"))==NULL)
        {
            cout<<"\n创建文件失败！";
        //  fclose(fp);
```

```
            return;
        }
        fclose(fp);
    }
    else
    {//用户要创建的文件已经存在
        cout<<"\n该文件已经存在，创建另一个文件？Y/N";
        char ch;
        cin>>ch;
        switch(ch)
        {
        case 'Y':
        case 'y':
            cout<<"\n输入新文件名：";
            cin>>buffer;
            strcat(buffer,".txt");
            fclose(fp);
            if((fp=fopen(buffer,"w"))==NULL)
            {
                cout<<"\n创建文件失败！";
            //  fclose(fp);
                return;
            }
            fclose(fp);
            break;
        default:
            fclose(fp);
            return;
        }
    }
    strcpy(user,curuser->username);
    strcat(user,".exe");
    curfile = getspace(UFD);
    strcpy(curfile->filename,buffer);//文件名
    curfile->length=0; //该文件长度为零
    curfile->safecode=30; //设置该文件的默认权限
        //11 00，文件主有读和写权，其他用户没有读写权
    if((fp=fopen(user,"ab"))==NULL)
    {
        cout<<"\n错误：你可能不是合法用户。"<<endl;
        getch();
    }
    else
    {
        fwrite(curfile,sizeof(UFD),1,fp);//将该文件信息写入用户文件信息管理模块中
        cout<<"\n文件 "<<curfile->filename<<" 创建成功！";
    }
    fclose(fp);
}

void DeleteFile()//删除当前目录下一个文件的操作
{
    char ch;
```

```
    FILE *infile,*outfile;
    cout<<"\n确定要删除文件："<<buffer<<" Y/N"<<endl;
    cin>>ch;//提示用户确认删除
    switch(ch)
    {
    case 'Y':
    case 'y'://更新用户文件信息管理模块，这里同样使用缓冲区模块来更新
            //方法与上面将到的类似
            char out[50],in[50];
            strcpy(out,"outfilelocate.exe");
            strcpy(in,curuser->username);
            strcat(in,".exe");
            if((infile=fopen(in,"rb"))==NULL)//打开该用户的文件信息管理模块
            {
                cout<<"\n保存错误。";
                //fclose(infile);
                return;
            }
            else
            {
                if((outfile=fopen(out,"wb+"))==NULL)
                {
                    cout<<"\n保存错误。";// fclose(outfile);
                    fclose(infile);return;
                }
                else
                {
                    UFD *ufd = getspace(UFD);
                    while(1)
                    {
                        fread(ufd,sizeof(UFD),1,infile);//从旧模块读出信息
                        if(feof(infile))
                            break;
                        if((strcmp(ufd->filename,buffer))==0)//要进行更新的信息
                            continue;
                        else
                            fwrite(ufd,sizeof(UFD),1,outfile);//写入新模块
                    }
                    fclose(infile);fclose(outfile);
                    remove(in);//在磁盘移除就模块
                    rename(out,in); //新模块命名为当前用户文件信息管理模块
                }
            }
        remove(buffer);//从磁盘中删除该文件
        break;
    default:
        break;
    }
}

void ListAllFile()//显示当前用户目录下的文件信息
{
    DisplayUFD();
```

```
}

void OpenFile()//在window模式下打开该文件
{
    system(buffer);//buffer为文件名，如:file1.txt
}

bool QueryModElse(bool BOOL,bool &flag)//查询其它用户目录下文件的文件
//当该文件的权限允许当前用户对其执行有关操作时，返回ture
{
    FILE *fp;
    char user[50];
    UFD *ufd = getspace(UFD);//elseuser表示除当前用户外的所有用户注册目录
    strcpy(user,elseuser->username);
    strcat(user,".exe");
    if((fp=fopen(user,"rb"))==NULL){//打开一个其它的用户文件信息管理模块
    //  fclose(fp);
        cout<<"\n操作出现错误，对此我们表示歉意！";return false;
    }
    else{
        while(1){
            fread(ufd,sizeof(UFD),1,fp);
            if(feof(fp)){
                fclose(fp);return false;
            }
            if(strcmp(ufd->filename,buffer)==0){
                if(BOOL)//该用户请求写该文件
                {
                    if(ufd->safecode== 31 || ufd->safecode== 33)
                        //1101、1111最后一位为1，有写权
                        return true;
                    else{
                        cout<<"\n你无权对文件 "<<buffer<<" 执行此操作！";
                        flag=true;return false;}
                    //flag设置为true，告诉上一层，无须再查找
                    //该文件已经找到，但用户无权执行相关操作
                }
                else //该用户请求读权
                {
                    if(ufd->safecode == 32 || ufd->safecode == 33)
                        //1110、1111倒数第二位为1，有读权
                        return true;
                    else{
                        cout<<"\n你无权对文件 "<<buffer<<" 执行此操作！";
                        flag=true;return false;}
                }
            }
        }
    }
}

bool QueryMod(bool BOOL)//查询权限
{//首先在用户目录下查找，如果找不到用户当前要进行操作的文件名
```

```
    //则在其它注册用户目录下查找
    FILE *fp,*fp1;
    bool flag=false;
    char user[50];
    UFD *ufd = getspace(UFD);
    strcpy(user,curuser->username);
    strcat(user,".exe");

    if((fp=fopen(user,"rb"))==NULL){//打开用户文件信息管理模块
    //  fclose(fp);
        cout<<"\n操作出现错误，对此我们表示歉意！";return false;
    }
    else{//查找匹配的文件名，用户目录下的文件允许用户进行读写操作
        while(1){
            fread(ufd,sizeof(UFD),1,fp);
            if(feof(fp)){//在当前用户文件管理模块中找不到匹配文件
                //则继续在其它用户注册目录下查找
                fclose(fp);
                fp1=fopen("LOGIN.exe","rb");
                elseuser = getspace(MFD);
                bool BOOL_1=false;
                while(1){
                    fread(elseuser,sizeof(MFD),1,fp1);//读其它用户信息
                    if(feof(fp1) && !BOOL_1)//全部用户都查找完
                        //但仍然没找到匹配的文件
                        return false;
                    if(elseuser != curuser){
                       if((BOOL_1=QueryModElse(BOOL,flag)))//查找
                           return true;
                       if(flag)
                           return false;
                    }
                }
            }
            if(strcmp(ufd->filename,buffer)==0){//在当前用户注册目录下
                //找到该文件，返回真值
                fclose(fp);return true;
            }
        }
    }
}

bool WriteRight(int len,bool BOOL)//查看是否已经正确地写入到该文件信息中
//是则返回真值
{

    char user[50],outfile[50];
    FILE *fp,*fp1;
    strcpy(user,elseuser->username);
    strcat(user,".exe");
    if((fp=fopen(user,"rb"))==NULL){
            return false;
    }
```

```
    else{
        UFD *ufd = getspace(UFD);
        while(1){//在此用户目录下查找匹配文件
            fread(ufd,sizeof(UFD),1,fp);
            if(feof(fp)){
                fclose(fp);return false;
            }
            if((strcmp(ufd->filename,buffer))==0){//找到要写入新的长度的文件

                strcpy(outfile,"outfilelocate.exe");
                if((fp1=fopen(outfile,"wb+"))==NULL){
                    cout<<"\n错误：写入文件长度出错_3。";
                //  fclose(fp1);
                    fclose(fp);return false;
                }
                else{
                    fclose(fp);
            fp=fopen(user,"rb");//文件指针从新指向此用户文件信息管理模块开头
                    while(1){
                        fread(ufd,sizeof(UFD),1,fp);
                        if(feof(fp))
                            break;
                        if(strcmp(ufd->filename,buffer)==0){//找到匹配的文件
                            if(BOOL) ufd->length+=len; //在文件末追加内容的操作
                            else ufd->length =len;     //覆盖原文件内容
                        }
                        fwrite(ufd,sizeof(UFD),1,fp1);
                    }
                    fclose(fp);fclose(fp1);
                    remove(user);
                    rename(outfile,user);
                    return true;
                }
            }
        }
    }
}
void WriteLengthToFile(int Len,bool BOOL)//将文件长度写入文件管理模块中
{//因为当前用户可以对其它用户的文件进行操作（只要权限允许）
    //所以应该在整个文件系统目录下查找该文件的位置
    FILE *fp;
    if((fp=fopen("LOGIN.exe","rb"))==NULL){//不能打开文件
        cout<<"\n写入文件长度错误_1！";
    //  fclose(fp);
        return;
    }
    else{
        elseuser = getspace(MFD);
        while(1){
            fread(elseuser,sizeof(MFD),1,fp);
            if(feof(fp))
                break;
            else{
```

```
                if(WriteRight(Len,BOOL)){//查看是否已经正确地写入到该文件信息中
                    fclose(fp);return;
                }
            }
        }
        cout<<"\n写入文件长度错误_2！";
        fclose(fp);return;
    }
}

void WriteFile()//向文件写入信息的操作
{
    if(!QueryMod(true))//查询当前用户对该文件是否有写权
        return;//对该文件没有写权则返回
    char ch;
    int i=0;
    FILE *fp;
    if((fp=fopen(buffer,"r"))==NULL)//查询该文件是否存在
    {
        cout<<"\n该文件不存在，请创建该文件后再写入。";
    //  fclose(fp);
        return;
    }
    fclose(fp);
    cout<<"\n请选择写入方式："<<endl;
    cout<<"  1、覆盖原文件     2、在原文件末尾写入     3、取消"<<endl;
    cin>>ch;
    cout<<"开始输入正文："<<endl;
    switch(ch)
    {
    case '1'://覆盖原文件
        if((fp=fopen(buffer,"w"))==NULL)
            cout<<"\n文件打开失败。";
        else
        {
            ch=getchar();
            while(ch!='#')//将新的文件内容写入到文件的磁盘位置中
            {
                i++;
                fputc(ch,fp);
                ch=getchar();
            }
        }
        fclose(fp);
        WriteLengthToFile(i,false);//将文件长度写入文件管理模块
        break;
    case '2':
        if((fp=fopen(buffer,"a"))==NULL)
            cout<<"\n文件打开失败。";
        else
        {
            ch=getchar();
            while(ch!='#')//将新的文件内容写入到文件的磁盘位置中
```

```
            {
                i++;
                fputc(ch,fp);
                ch=getchar();
            }
        }
        fclose(fp);
        WriteLengthToFile(i,true);//将文件长度写入文件管理模块
        break;
    default:
        break;
    }
}

void ReadFile()//读文件函数
{
    if(!QueryMod(false))//查询当前用户是否有权读该文件
        return;//没有读权，则返回
    FILE *fp;
    if((fp=fopen(buffer,"r"))==NULL)//打开该文件
    {
        cout<<buffer;
        cout<<"\n该文件不存在。";
        return;
    }
    else{
        char ch;
        ch=fgetc(fp);
        while(ch!=EOF)//将该文件信息逐一输出到终端
        {
            putchar(ch);
            ch=fgetc(fp);
        }
        cout<<endl;
    }
    fclose(fp);
}

void ChangeMod()//修改某文件的执行权限
{
    int mod=40;
    FILE *fp,*infile,*outfile;

    char in[50],out[50];
    UFD *ufd = getspace(UFD);
    strcpy(in,curuser->username);
    strcat(in,".exe");
    strcpy(out,"outfilelocate.exe");

    if((fp=fopen(in,"rb"))==NULL){
    //  fclose(fp);
        cout<<"\n操作出现错误，对此我们表示歉意！";return;
    }
```

```
    else{
        while(1){//查看该文件是否在当前用户的注册目录下
            //任何用户无权修改不是自己目录下的文件的权限值
            fread(ufd,sizeof(UFD),1,fp);
            if(feof(fp)){//在当前目录下找不到该文件，说明该用户无权修改该文件权限
                cout<<"\n你没有权限对文件 "<<buffer<<" 执行该操作！";
                fclose(fp);return;
            }
            if(strcmp(ufd->filename,buffer)==0){//找到该文件，继续操作
                fclose(fp);break;
            }
        }
    }
    bool flag1=true;
    while(flag1)
    {
        cout<<"\n输入文件  "<<buffer<<" 的新的权限值：";
        cin>>mod;//输入权限值
        if(mod<30 || mod>33)
        {//确保输入的权限值正确
            cout<<"\n错误：权限值必须在30~33之间";
            continue;
        }
        else{
            char ch;
            switch(mod){//告诉用户对该文件权限修改的结果，以便用户确认
            case 30:
            cout<<"\n当前权限设置：其他用户对"<<buffer<<"既没读权也没写权！";
                break;
            case 31:
                cout<<"\n当前权限设置：其他用户对"<<buffer<<"没读权但有写权！";
                break;
            case 32:
                cout<<"\n当前权限设置：其他用户对"<<buffer<<"有读权但没写权！";
                break;
            case 33:
            cout<<"\n当前权限设置：其他用户对"<<buffer<<"既有读权也有写权！";
                break;
            default: break;
            }
            cout<<"\n确认按'Y'，取消按'N':";
            cin>>ch;
            switch(ch){
            case 'Y':
            case 'y':flag1=false;break;
            default: flag1=true;
            }
        }
    }
    //更新文件信息管理模块，相关操作类似上面，不在赘述
    if((infile=fopen(in,"rb"))==NULL){
        cout<<"\n操作出现错误，对此我们表示歉意！";fclose(infile);
        return;
```

```
    }
    else{
        if((outfile=fopen(out,"wb+"))==NULL){
            cout<<"\n操作出现错误，对此我们表示歉意！";
            fclose(infile);//fclose(outfile);
            return;
        }
        else{
            while(1)
            {
                fread(ufd,sizeof(UFD),1,infile);
                if(feof(infile))
                    break;
                if((strcmp(ufd->filename,buffer))==0)
                    ufd->safecode=mod;
                fwrite(ufd,sizeof(UFD),1,outfile);
            }
            fclose(infile);fclose(outfile);
            remove(in);
            rename(out,in);
        }
    }
}
void Execute(int i,int len,int cmdset)//执行命令函数
{
    int j=0;
    for(;i<len;i++)
    {
        if(cmd[i]=='>'||cmd[i]==' ')
            break;
        buffer[j]=cmd[i];j++;
    }
    buffer[j]='\0';
    strcat(buffer,".txt");
    switch(cmdset)
    {
    case 1: //退出
        ByeFile(true);
        break;
    case 2: //改变文件操作权限
        if((strcmp(buffer,".txt"))==0){
            cout<<"\n输入命令出错！";
            return;
        }
        ChangeMod();
        break;
    case 3: //删除用户
        DeleteUser();
        break;
    case 4: //创建文件
        if((strcmp(buffer,".txt"))==0){
            cout<<"\n输入命令出错！";
            return;
```

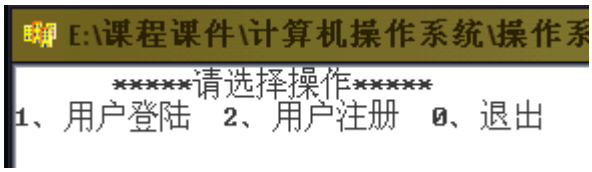
```
        }
        CreatFile();
        break;
    case 5: //删除文件
        if((strcmp(buffer,".txt"))==0){
            cout<<"\n输入命令出错！";
            return;
        }
        DeleteFile();
        break;
    case 6: //列出该用户所有文件清单
        ListAllFile();
        break;
    case 7: //打开文件
        if((strcmp(buffer,".txt"))==0){
            cout<<"\n输入命令出错！";
            return;
        }
        OpenFile();
        break;
    case 8: //读文件
        if((strcmp(buffer,".txt"))==0){
            cout<<"\n输入命令出错！";
            return;
        }
        ReadFile();
        break;
    case 9: //写文件
        if((strcmp(buffer,".txt"))==0){
            cout<<"\n输入命令出错！";
            return;
        }
        WriteFile();
        break;
    default:
        break;
    }
}
void Command()//读取用户输入的命令，并将其转换成系统能识别的命令
{
    int len = 0,i,j;
    int cmdset;
    while(1)
    {
        cmdset = 0;
        cout<<"\n执行2";
        cin>>cmd;
        len = strlen(cmd);
        i=0;j=0;
        while(cmd[i]=='>'||cmd[i]==' '){i++;}//过滤空格键和'>'
        for(;i<len;i++)
        {
```

```
            if(cmd[i]=='>' || cmd[i]==' ' || i==len-1)
            {
                if(cmd[i]=='>' || cmd[i]==' ')
                    buffer[j] = '\0';
                else
                    if(i==len-1)
                    {
                        buffer[j]=cmd[i];
                        buffer[j+1]='\0';
                    }
                i++;
                j=0;
                int low=1,mid,high=keynum-1;
                bool BOOL = false;
                while(low<=high){//找到该命令关键字的内部识别码
                    mid=(low+high)/2;
                    if (strcmp(buffer,KWORD[mid])<=0) high=mid-1;
                    if (strcmp(buffer,KWORD[mid])>=0) low=mid+1;
                    if(strcmp(buffer,KWORD[mid])==0){
                            BOOL = true;
                            break;
                    }
                }
                if(!BOOL)
                {
                    cout<<"\n"<<buffer<<"不是系统定义的命令...";

                    cmdset = 0; break;
                }
                else {cmdset = mid;break;}
            }
            else{
                buffer[j] = cmd[i];
                j++;
            }
        }
        if(cmdset == 0) continue;
        while(cmd[i]=='>'||cmd[i]==' '){i++;}//过滤空格键和'>'
        buffer[0]='\0';
        Execute(i,len,cmdset);  //执行该命令
    }
}

int main()
{
    initscr();
    while(1){
        int SELETE = LoginDisplay();
        if(SELETE==0)
            exit(0);
        bool BOOL = Login(SELETE);//用户登陆，或者注册函数
        if(BOOL)
        {
```

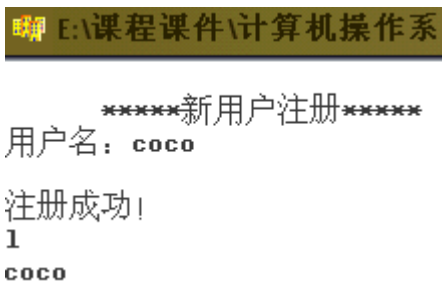
```
            KeyWord();  //初始化命令关键字
            DisplayUFD();//打印用户目录下的文件
            Command();   //命令行操作
        }
    }
    endwin();
    return 0;
```

五、测试

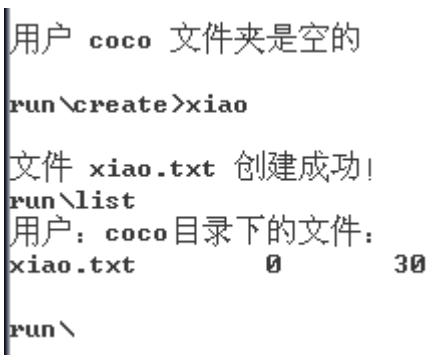
1）系统界面如下



2)创建新用户：如果你以前还没有注册，则可以先选择2创建一个用户



3)创建用户后,可以用create>xiao命令建立一个名为xiao的text文件并可以用list查看文件的信息(xiao.txt是文件名，0表示文件当前的长度为0，30用二进制表示为1100，表示当前用户有读写权，而其它用户没有读权也没有写权)。



4)用write>xiao命令向这个文件写信息,选择1或是2,输入信息如下（注意：所有信息输入完后要以'#'号键作为结束标志。）

```
请选择写入方式：
  1、覆盖原文件      2、在原文件末尾写入     3、取消
1
开始输入正文：
xiao,你好!欢迎使用本系统!

00
3
#
```

5)用read>xiao命令读文件中的内容

```
run\read>xiao

xiao,你好!欢迎使用本系统!

00
3


run\
```

6)此时再write>xiao命令向这个文件写信息,选择1,输入信息,并用read>xiao查看,结果如下

```
请选择写入方式：
  1、覆盖原文件      2、在原文件末尾写入     3、取消
1
开始输入正文：
再次登陆文件#

run\read>xiao

再次登陆文件

run\
```

7)还可以用open>xiao这个命令打开文件,发现文件内容变为