



***Liaoning Normal University***

**操作系统实验报告**

**课程名称：操作系统**

**专 业：计算机科学与技术（师范）**

**班 级：（1） 班**

**学 号：201621012301**

**姓 名：刘雪阳**

**指导教师：张凤梅**

**2018年12月16日**

一、实验目的

理解文件控制块的作用，模拟用户磁盘文件的打开，读取，写入等操作

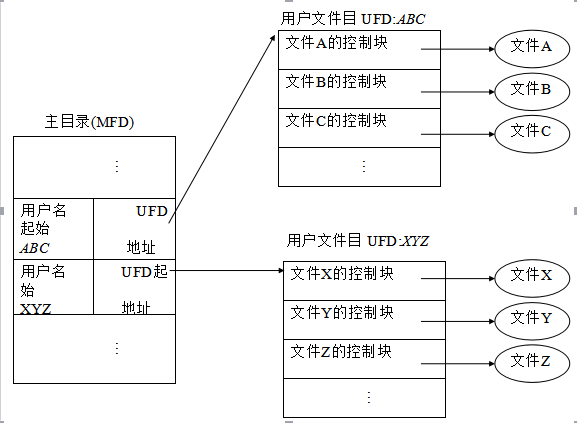
二、实验环境

Windows 10，Microsoft Visual C++

三、实验内容

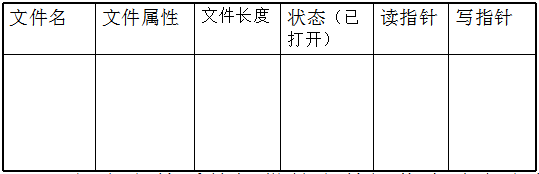
1. 数据结构

（1）采用二级目录结构时，第一级为主文件目录MFD，第二级为用户文件目录UFD，假定系统可同时管理N个用户文件，每个用户最多在磁盘上保存L个文件。约定用户把文件组织成记录式文件，文件中每个记录都是定长的，文件在磁盘上的组织形式索引，采用直接指针来记录文件创建时的数据块的地址，二级目录结构可用如下形式：



用户文件目录UFD中的文件控制块（FCB）有文件名，文件的存取权限，文件长度，文件的地址，文件的存取权限规定文件的使用权限为只读或可读可写，文件长度指组成该文件逻辑记录长度，文件地址指文件在外存磁盘上文件的首块地址，及其它块的地址（假定文件较小，一条逻辑记录恰好占用一个块）。

(2) 为每个设置一个“用户已打开文件表（UOF），用以存放该用户当前正在使用文件的情况。如果用户最多同时打开或建立S个文件，则UOF中有S个登记栏。具体如下表：



1. 实验要求

（1）显示初始的文件目录表、输入的文件操作命令和每条命令执行后的UOF文件，程序运行后打印所有用户的文件目录表。

（2）假定文件系统提供的文件操作有建立文件（CREATE），打开文件（OPEN），关闭文件（CLOSE），读文件（READ），写文件（WRITE）和删除文件（DELETE），在模拟程序中可以从键盘上输入文件操作命令来模拟各用户程序中所调用的各种文件操作，用一个结束命令（end)停止程序执行。

（3）用输入命令来模拟用户程序中调用的文件操作，这些命令的格式和和要求如下：

①建立文件：create(文件名，文件长度，文件属性）

②写文件：write(文件名，记录号）

有两种情况，第一种是在执行create后要求写，第二种是在执行open后要求写，即对一个已存在的文件进行修改。

③关闭文件：close(文件名），对某个文件不需要再读或写时，用户应关闭文件。

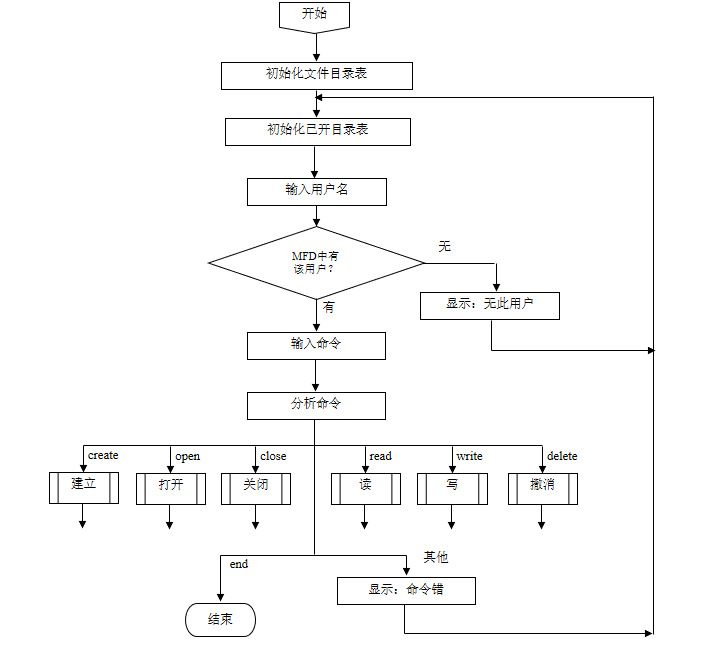
④打开文件：open(文件名，操作类型），其中操作类型指出文件打开后，用户将对文件进行读或写。约定操作类型与文件属性不符合或正处于“建立”状态的文件不允许打开。

⑤读文件：read(文件名，读长度），由于文件采用索引，可顺序读，也可随机读，自行设计流图。

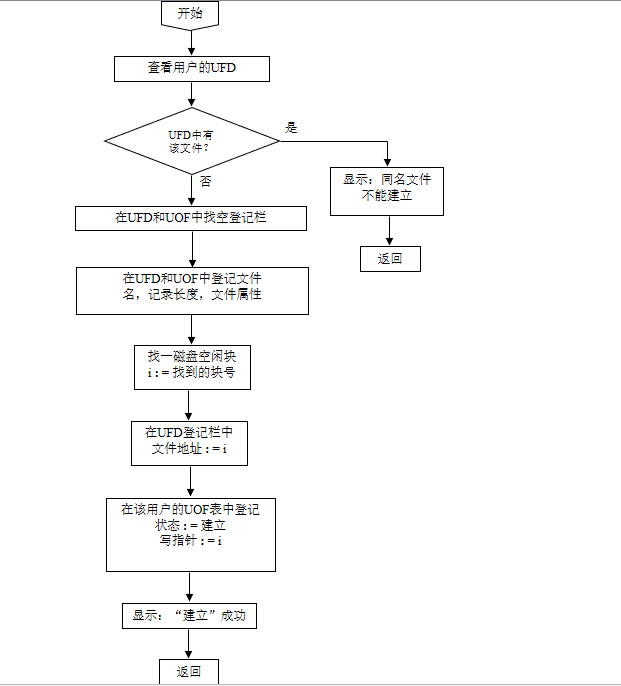
⑥删除文件 delete(文件名），从相应的用户文件目录UFD中将文件控制块删除，收回该文件占用的存储区域。自行设计流程。

四、实验流程图

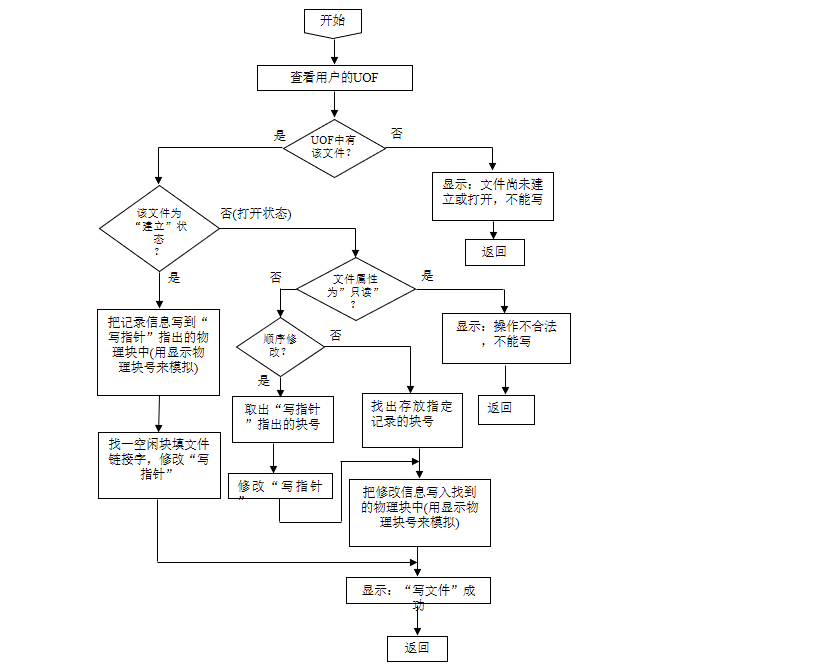
（1）主程序流程图



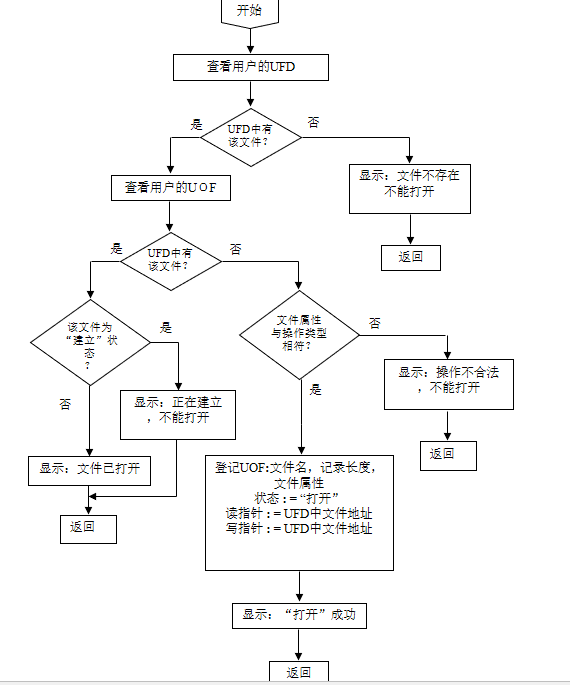
（2）建立文件



（3）写文件



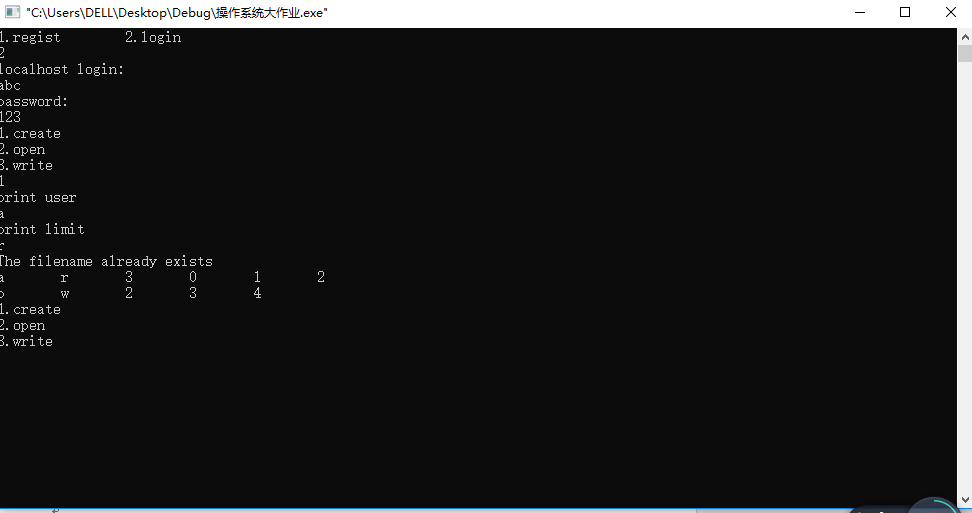
1. 打开文件



五、实验结果

1. 登录系统创建文件

a.文件已存在

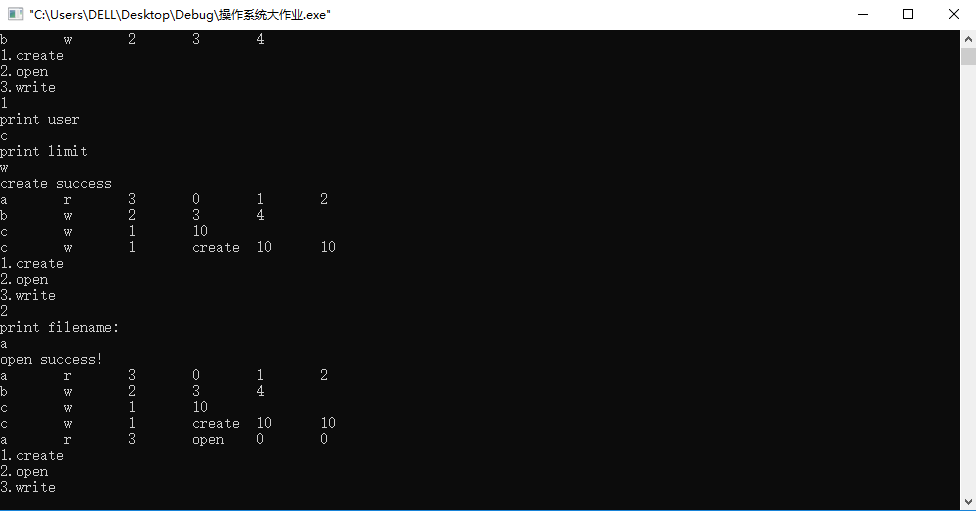


b.创建文件成功

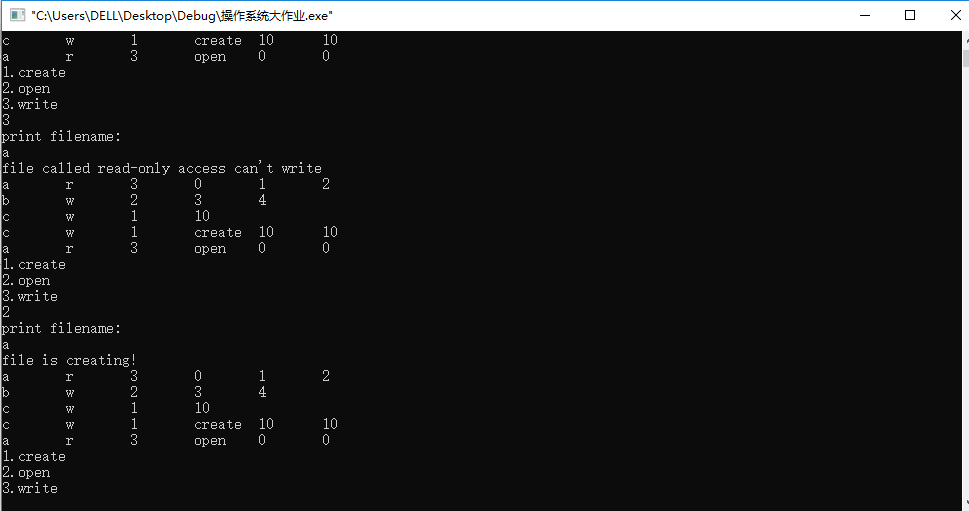


1. 打开文件

a.成功打开文件

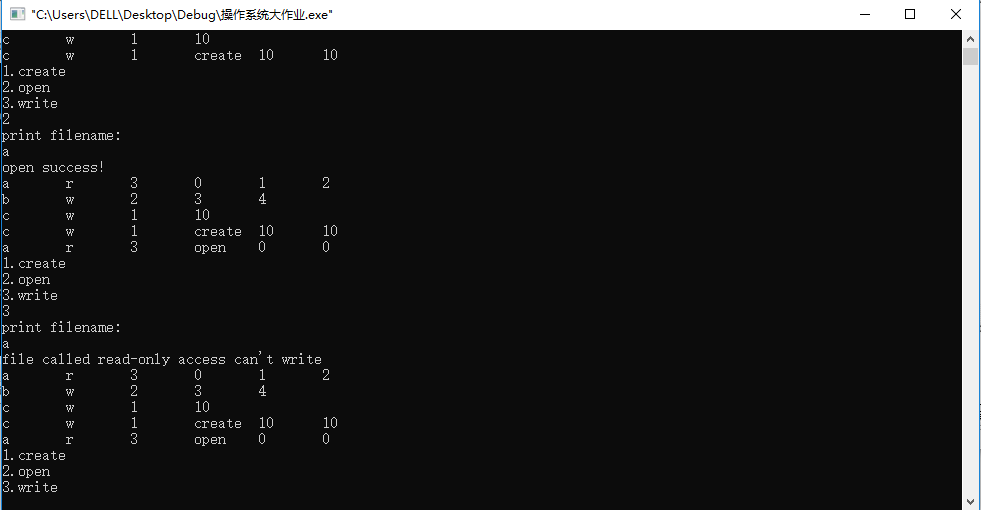


b.文件正在打开

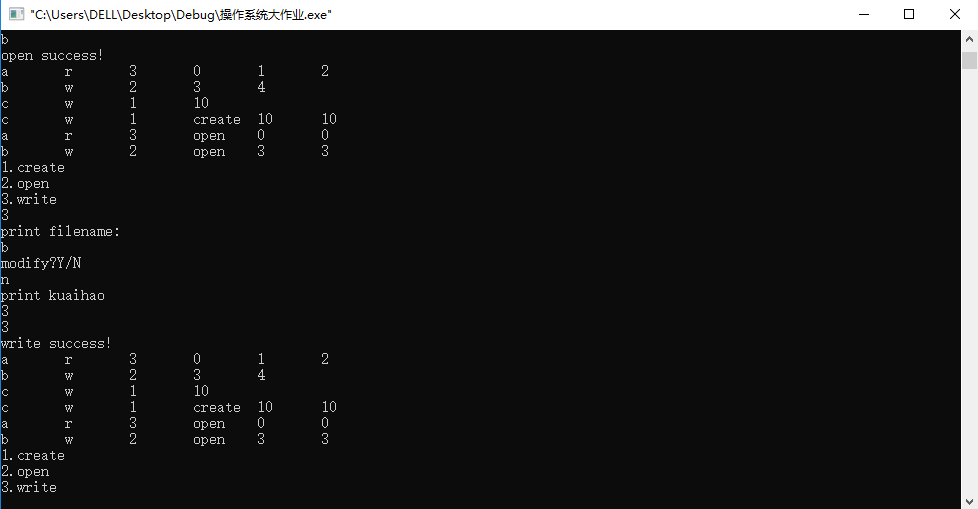


1. 写文件

a.文件只读不能写入



b.写入文件成功



六、实验代码

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int cipan[100]={1,1,1,1,1,1,1,1,1,1};

struct user

{

char name[20];

char password[20];

int id;

}user[10]={{"abc","123",0},{"xyz","123",1}};

//主文件目录

struct MFD

{

char filename[20]; //文件名

char mode; //文件存取权限

int length; //文件长度

int address[4]; //文件地址

}mfd[10][20]={{{"a",'r',3,{0,1,2}},{"b",'w',2,{3,4}}},{{"x",'r',3,{5,6,7}},{"y",'w',2,{8,9}}}};

//用户已打开文件表

struct UOF

{

char filename[20];//文件名

char mode;//文件属性

int length;//文件长度

char status[10];//状态

int read;//读指针

int write;//写指针

}uof[10];

//注册

void regist(){

int i;

for(i=0;i<10;i++)

{

if(strlen(user[i].name)==0){

scanf("%s",&user[i].name);

scanf("%s",&user[i].password);

scanf("%d",&user[i].id);

return;

}

}

}

int login(){

int i;

char name[20];

char password[20];

printf("localhost login:\n");

scanf("%s",&name);

printf("password:\n");

scanf("%s",&password);

for(i=0;i<10;i++)

{

if(strcmp(user[i].name,name)==0)

{

if(strcmp(user[i].password,password)==0)

return user[i].id;

}

else

{

printf("password error\n");

return -1;

}

}

printf("no user\n");

return -1;

}

void create(int id){

int i,j,k;

char mode;

char filename[10];

printf("print user\n");

scanf("%s",&filename);

getchar();

printf("print limit\n");

scanf("%c",&mode);

for(i=0;i<20;i++)

{

//UFD中有该文件

if(strcmp(mfd[id][i].filename,filename)==0)

{

printf("The filename already exists\n");

return;

}

}

for(i=0;i<20;i++)

{

if(strlen(mfd[id][i].filename)==0)

{

for(j=0;j<100;j++)

{

if(cipan[j]==0)

break;

}

//登记文件名，文件属性，长度，地址

strcpy(mfd[id][i].filename,filename);

mfd[id][i].mode=mode;

mfd[id][i].length=1;

mfd[id][i].address[0]=j;

cipan[j]=1;

for(i=0;i<10;i++)

{

//找一磁盘空闲块

if(strlen(uof[i].filename)==0)

{

strcpy(uof[i].filename,filename);

uof[i].length=1;

uof[i].mode=mode;

strcpy(uof[i].status,"create");

uof[i].write=j;

uof[i].read=j;

printf("create success\n");

return;

}

}

}

}

}

void write(int id)

{

int i,j,ms,kuaihao;

char filename[10];

printf("print filename:\n");

scanf("%s",&filename);

getchar();

//UFD中已有同名文件

for(i=0;i<20;i++)

{

if(strcmp(mfd[id][i].filename,filename)==0)

break;

}

//UOF中有无该文件

for(j=0;j<10;j++)

{

if(strcmp(uof[j].filename,filename)==0)

{

//文件是已建立状态

if(strcmp(uof[j].status,"create")==0)

{

printf("%d\n",uof[j].write);

printf("write success!\n");

return;

}

else

{

//文件属性只读

if(uof[j].mode=='r')

{

printf("file called read-only access can't write\n");

return;

}

else

{

printf("modify?Y/N\n");

scanf("%c",&ms);

//顺序修改

if(ms=='Y')

{

printf("%d\n",uof[j].write);

printf("write success\n");

return;

}

else

{

printf("print kuaihao\n");

scanf("%d",&kuaihao);

printf("%d\n",kuaihao);

printf("write success!\n");

return;

}

}

}

}

}

//UOF中没有该文件

printf("file not create or open!\n");

return;

}

void open(int id)

{

int i,j;

char filename[10];

printf("print filename:\n");

scanf("%s",&filename);

for(i=0;i<20;i++)

{

if(strcmp(mfd[id][i].filename,filename)==0)

{

for(j=0;j<10;j++)

{

if(strcmp(uof[j].filename,filename)==0)

{

//该文件正在建立

if(strcmp(uof[i].status,"create")==0)

{

printf("file is creating!\n");

return;

}

strcpy(uof[j].status,"open");

printf("file is opening!\n");

return;

}

}

for(j=0;j<10;j++)

{

if(strlen(uof[j].filename)==0)

{

//登记uof

strcpy(uof[j].filename,filename);

uof[j].length=mfd[id][i].length;

uof[j].mode=mfd[id][i].mode;

//文件属性为打开

strcpy(uof[j].status,"open");

uof[j].write=mfd[id][i].address[0];

uof[j].read=mfd[id][i].address[0];

printf("open success!\n");

return;

}

}

}

}

printf("files does not exit!\n");

return;

}

void output(int id)

{

int i,j;

for(i=0;i<20;i++)

{

if(strlen(mfd[id][i].filename)==0);

else

{

printf("%s\t",mfd[id][i].filename);

printf("%c\t",mfd[id][i].mode);

printf("%d\t",mfd[id][i].length);

for(j=0;j<mfd[id][i].length;j++)

{

printf("%d\t",mfd[id][i].address[j]);

}

printf("\n");

}

}

for(i=0;i<10;i++){

if(strlen(uof[i].filename)==0);

else{

printf("%s\t",uof[i].filename);

printf("%c\t",uof[i].mode);

printf("%d\t",uof[i].length);

printf("%s\t",uof[i].status);

printf("%d\t",uof[i].read);

printf("%d\t",uof[i].write);

printf("\n");

}

}

}

main(){

int id,choice,n=1;

printf("1.regist\t2.login\n");

scanf("%d",&choice);

if(choice==1)

regist();

while((id=login())==-1);

while(1){

printf("1.create\n");

printf("2.open\n");

printf("3.write\n");

scanf("%d",&choice);

switch(choice){

case 1:create(id);output(id);break;

case 2:open(id);output(id);break;

case 3:write(id);output(id);break;

}

}

}