**云南大学软件学院**

**Linux应用开发实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** | 何志颖 | **年级** | **2015** | **班级** | **软件工程** |
| **学号** | 20151120237 | **专业** | **软件工程** | **序号** | **31** |
| **实验名称** | **实验四 文件操作与管理实验** | | | **成绩** |  |

指导教师 (签名):

# 实验四 文件操作与管理

## **一、实验目的**

1．掌握函数stat中文件属性的应用。

2．掌握系统函数system、opendir、scandir的使用。

3．初步掌握struct dirent 的结构体变量的使用方法。

4．掌握文件阻塞与非阻塞I/O的操作。

5. 掌握文件属性的判断。

6．掌握文件锁的使用。

7. 掌握文件目录与文件的递归及深度遍历。

## 二、实验目标与要求

1.测试文件S\_IRUSR、S\_IWUSR、S\_IRGRP、S\_IROTH属性。

2.应用system函数对网站的网络连通情况进行测试。

3.应用readdir函数显示文件和子目录。

4.文件属性的判断。

5.阻塞I/O文件操作的程序设计。

6. 文件加锁、解锁的操作，文件锁属性的判断。

## 三、实验工具与准备

计算机PC机，Linux Redhat 9.0或者其他的linux操作系统

## **四、实验步骤与操作指导**

**任务一：程序设计**

设计程序应用system函数建立/home/tim文件，应用chmod函数使文件tim具有S\_IRUSR、S\_IWUSR、S\_IRGRP、S\_IROTH属性，最后应用函数stat获取文件的大小与建立的时间。

**任务二：程序设计**

设计一个程序，要求判断“/etc/passwd”的文件类型。提示：使用st\_mode属性，可以使用几个宏来判断：S\_ISLNK(st\_mode) 是否是一个连接，S\_ISREG是否是一个常规文件S\_ISDIR是否是一个目录，S\_ISCHR是否是一个字符设备，S\_ISBLK是否是一个块设备，S\_ISFIFO是否是一个FIFO文件，S\_ISSOCK是否是一个SOCKET文件。

**任务三：调试并分析下面程序的结果**

程序的功能是要求打印系统目录“/etc/rc.d”下所有的文件和子目录的名字。程序代码如下：

#include<stdio.h>

#include<sys/types.h>

#include<dirent.h>

#include<unistd.h>

int main()

{

DIR \* dir;

struct dirent \* ptr;

dir=opendir("/etc/rc.d");

printf("/etc/rc.d目录中文件或子目录有:\n");

while((ptr = readdir(dir))!=NULL)

{

printf("%s\n",ptr->d\_name);

}

closedir(dir);

}

**问题：**

（1）修改程序，要求读取“/etc”目录下所有的目录结构，并依字母顺序排列。

#include<dirent.h>

考虑以下语句：

scandir("/etc",&namelist,0,alphasort);

程序段：

while(n--)

{

printf("%s\n", namelist[n]->d\_name);

free(namelist[n]);

}

**任务四：调试并分析下面程序的结果**

程序的功能是用递归的方法列出某一目录下的全部文件的大小和文件夹及创建日期，包括子文件和子文件夹。程序代码如下：

#include<stdio.h>

#include<time.h>

#include<linux/types.h>

#include<dirent.h>

#include<sys/stat.h>

#include<unistd.h>

#include<string.h>

char \*wday[]={"日","一","二","三","四","五","六"};

void list(char \*name,int suojin)

{

DIR \*dirname;

struct dirent \*content;

struct stat sb;

struct tm \*ctime;

int i;

if((dirname=opendir(name))==NULL)

{

printf("该目录不存在\n");

return;

}

chdir(name);/\*改换工作目录\*/

while((content=readdir(dirname))!=NULL)

{

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

putchar('\t');

if(content->d\_type==4)

printf("目录\t");

else if(content->d\_type==8)

printf("文件\t");

else

printf("其他\t");

stat(content->d\_name,&sb);

ctime=gmtime(&sb.st\_mtime);

printf("%d年%d月%d日 星期%s %d:%d:%d\t",ctime->tm\_year+1900,

1+ctime->tm\_mon,ctime->tm\_mday,wday[ctime->tm\_wday],ctime->tm\_hour,

ctime->tm\_min,ctime->tm\_sec);

printf("%d\t",sb.st\_size);

printf("%s\n",content->d\_name);/\*列出目录或文件的相关信息\*/

if(content->d\_type==4&&strcmp(content->d\_name,"..")&&strcmp(content->d\_name,"."))

{

list(content->d\_name,suojin+1);/\*如果是目录，则递归列出目录里的内容\*/

}

}

closedir(dirname);

chdir("..");/\*当该层目录中的文件列完后，返回父目录\*/

}

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

{

char name[256];

printf("类型\t最后修改时间\t\t\t大小\t文件名\n");

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

if(argc==1) {

printf("Enter directory name:");

scanf("%s",name);

list(name,0);

}

else

{

list(argv[1],0);

}

}

**任务五：调试下列程序**

程序主要运用一个非阻塞I/O的例子，程序打开当前终端文件/dev/tty，在打开时指定O\_NONBLOCK标志。程序运行时每隔一定时间（6秒）等待用户从终端输入，等待30秒，每次等待时屏幕都有提示“”，30秒后程序继续执行主程序，输出以下图形后结束。

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

#include <unistd.h>

#include <fcntl.h>

#include <errno.h>

#include <string.h>

#include <stdlib.h>

#define MSG\_TRY "try again\n"

#define MSG\_TIMEOUT "timeout\n"

int main(void)

{

char buf[10];

int fd, n, i, j;

fd = open("/dev/tty", O\_RDONLY|O\_NONBLOCK);

if(fd<0) {

perror("open /dev/tty");

exit(1);

}

for(i=0; i<5; i++) {

n = read(fd, buf, 10);

if(n>=0)

break;

if(errno!=EAGAIN) {

perror("read /dev/tty");

exit(1);

}

sleep(6);

write(STDOUT\_FILENO, MSG\_TRY, strlen(MSG\_TRY));

}

if(i==5)

write(STDOUT\_FILENO, MSG\_TIMEOUT, strlen(MSG\_TIMEOUT));

else

write(STDOUT\_FILENO, buf, n);

for(i=0;i<5;i++){

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

printf("%2c",'\*');

printf("\n");

}

close(fd);

return 0;

}

程序运行时，轮询等待用户的输入，等待期间如有输入，即转入主程序执行，如没有输入，30秒后执行主程序。

**问题：**

（1）修改程序，程序性质为阻塞的；

（2）修改程序，程序运行中不断地向文件/home/tim写入ps的情况，如果有输入，则测试与某网站（比如云南大学校园网）的连通情况；

**任务六：调试下面的程序**

程序在“/root”下打开一个名为“4-6file”的文件，如果该文件不存在，则创建此文件。打开后对其加上强制性的写入锁F\_WRLCK，按回车后解锁F\_UNLCK，然后加上读出锁F\_RDLCK，按回车后再解锁F\_UNLCK。程序在终端1运行后会显示程序的进程号，再打开终端2，会提示此文件处于锁定状态，此时在终端2可以多按回车，观察程序的运行结果。然后在终端1按回车，等待终端1解锁后，在终端2才可锁定此文件，你可观察到强制性锁是独占状态，当在终端2解锁后，在终端1或2可加读出锁，在读出锁状态终端1或2的运行不需要等待，因为读出锁是处于共享状态，请编写程序并测试程序运行的结果。主程序先用open函数打开文件“4-6file”，如果该文件不存在，则创建此文件；接着调用自定义函数lock\_set：先传递参数“F\_WRLCK”给文件“4-6file”加锁，并打印输出给文件加锁进程的进程号，然后先传递参数“F\_UNLCK”给文件“4-6file”解锁，并打印输出给文件解锁进程的进程号；在自定义函数lock\_set给文件上锁语句前，加上判断文件是否上锁的语句，如果文件已经被上锁，打印输出给文件上锁进程的进程号。

程序代码如下：

/\*4-6.c程序：打开“/root/4-6file”后对其加上强制性的写入锁，然后释放写入锁\*/

#include<stdio.h>

#include<stdlib.h>

#include <unistd.h>

#include <sys/file.h>

#include <sys/types.h>

#include <sys/stat.h>

void lock\_set(int fd, int type)

{

struct flock lock;

lock.l\_whence = SEEK\_SET;

lock.l\_start = 0;

lock.l\_len =0;

while(1){

lock.l\_type = type;

if((fcntl(fd,F\_SETLK,&lock))==0){/\*根据不同的type值给文件加锁或解锁\*/

if( lock.l\_type == F\_RDLCK ) /\*F\_RDLCK为共享锁，表示读取锁或建议性锁\*/

printf("加上读取锁的是： %d\n",getpid());

else if( lock.l\_type == F\_WRLCK ) /\*F\_WRLCK为排斥锁，表示强制性锁\*/

printf("加上写入锁的是： %d\n",getpid());

else if( lock.l\_type == F\_UNLCK )

printf("释放强制性锁： %d\n",getpid());

return;

}

fcntl(fd, F\_GETLK,&lock); /\*读取文件锁的状态\*/

if(lock.l\_type != F\_UNLCK){

if( lock.l\_type == F\_RDLCK )

printf("文件已经加上了读取锁，其进程号是： %d\n",lock.l\_pid);

else if( lock.l\_type == F\_WRLCK )

printf("文件已加上写入锁，其进程号是： %d\n",lock.l\_pid);

getchar();

}

}

}

int main ()

{

int fd;

fd=open("/root/4-6file",O\_RDWR | O\_CREAT, 0666);

if(fd < 0)

{

perror("打开出错");

exit(1);

}

lock\_set(fd, F\_WRLCK);

getchar();

lock\_set(fd, F\_UNLCK);

getchar();

lock\_set(fd, F\_RDLCK);

getchar();

lock\_set(fd, F\_UNLCK);

close(fd);

exit(0);

}

**问题：**

（1）在不同的终端调试程序，分析程序的运行情况；

（2）修改程序，如果把文件名作一个房号，请描述宾馆客人入住的情况。

**五、实验算法及程序**

任务一：

#include<stdlib.h>

#include<sys/types.h>

#include<sys/stat.h>

#include<unistd.h>

#include<stdio.h>

int main(){

struct stat buf;

system("sudo mkdir /home/tim");

chmod("/home/tim",S\_IRUSR | S\_IWUSR | S\_IRGRP | S\_IROTH);

system("sudo ls -l /home");

stat("/home/tim",&buf);

printf("/home/tim 的文件大小为：%d\n",(int)(buf.st\_size));

return 0;

}

任务二：

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/stat.h>

void what\_is( int);

int main()

{

struct stat info;

if( stat("/etc/passwd", &info) == -1 )

perror("etc/passwd");

else

what\_is( info.st\_mode );

printf("last access time is %s\n", ctime(&info.st\_mtime)); //>以字符串的形式打印时间；

return 0;

}

void what\_is( int mode )

{

if( S\_ISDIR(mode) ) printf("/etc/passwd is Directory file.\n"); //stat\_mode返回的是条件中的字符串；

if( S\_ISFIFO(mode) ) printf("/etc/passwd is Fifo file.\n");

if( S\_ISLNK(mode) ) printf("/etc/passwd is link file.\n");

if( S\_ISREG(mode) ) printf("/etc/passwd is regular file.\n");

if( S\_ISSOCK(mode) ) printf("/etc/passwd is socket file.\n");

}

任务三：

1. （1）修改程序，要求读取“/etc”目录下所有的目录结构，并依字母顺序排列。

#include <stdio.h>

#include <stdlib.h>

#include <dirent.h>

int main(){

struct dirent \*\*namelist;

int i,total;

total = scandir("/",&namelist,0,alphasort);

if(total < 0){

perror("scandir");

}else{

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

printf("%s\n",namelist[i]->d\_name);

}

printf("total = %d\n",total);

}

return 0;

}

任务四：

程序代码如下：

#include<stdio.h>

#include<time.h>

#include<linux/types.h>

#include<dirent.h>

#include<sys/stat.h>

#include<unistd.h>

#include<string.h>

char \*wday[]={"日","一","二","三","四","五","六"};

void list(char \*name,int suojin)

{

DIR \*dirname;

struct dirent \*content;

struct stat sb;

struct tm \*ctime;

int i;

if((dirname=opendir(name))==NULL)

{

printf("该目录不存在\n");

return;

}

chdir(name);/\*改换工作目录\*/

while((content=readdir(dirname))!=NULL)

{

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

putchar('\t');

if(content->d\_type==4)

printf("目录\t");

else if(content->d\_type==8)

printf("文件\t");

else

printf("其他\t");

stat(content->d\_name,&sb);

ctime=gmtime(&sb.st\_mtime);

printf("%d年%d月%d日 星期%s %d:%d:%d\t",ctime->tm\_year+1900,

1+ctime->tm\_mon,ctime->tm\_mday,wday[ctime->tm\_wday],ctime->tm\_hour,

ctime->tm\_min,ctime->tm\_sec);

printf("%d\t",sb.st\_size);

printf("%s\n",content->d\_name);/\*列出目录或文件的相关信息\*/

if(content->d\_type==4&&strcmp(content->d\_name,"..")&&strcmp(content->d\_name,"."))

{

list(content->d\_name,suojin+1);/\*如果是目录，则递归列出目录里的内容\*/

}

}

closedir(dirname);

chdir("..");/\*当该层目录中的文件列完后，返回父目录\*/

}

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

{

char name[256];

printf("类型\t最后修改时间\t\t\t大小\t文件名\n");

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

if(argc==1) {

printf("Enter directory name:");

scanf("%s",name);

list(name,0);

}

else

{

list(argv[1],0);

}

}

任务五：

（1）程序代码如下：

#include <stdio.h>

#include <fcntl.h>

#include <errno.h>

#include <string.h>

#include <stdlib.h>

#define MSG\_TRY "try again\n"

#define MSG\_TIMEOUT "timeout\n"

int main(void)

{

char buf[10];

int fd, n, i, j;

fd = open("/dev/tty", O\_RDONLY & ~O\_NONBLOCK);

if(fd<0) {

perror("open /dev/tty");

exit(1);

}

for(i=0; i<5; i++) {

n = read(fd, buf, 10);

if(n>=0)

break;

if(errno!=EAGAIN) {

perror("read /dev/tty");

exit(1);

}

sleep(6);

write(STDOUT\_FILENO, MSG\_TRY, strlen(MSG\_TRY));

}

if(i==5)

write(STDOUT\_FILENO, MSG\_TIMEOUT, strlen(MSG\_TIMEOUT));

else

write(STDOUT\_FILENO, buf, n);

for(i=0;i<5;i++){

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

printf("%2c",'\*');

printf("\n");

}

close(fd);

return 0;

}

（2）修改程序，程序运行中不断地向文件/home/tim写入ps的情况，如果有输入，则测试与某网站（比如云南大学校园网）的连通情况；

程序代码如下：

System(“ping -c 5 [www.ynu.edu.cn](http://www.ynu.edu.cn) >> /home/tim”);

任务六：

（1）程序代码如下：

/\*4-6.c程序：打开“/root/4-6file”后对其加上强制性的写入锁，然后释放写入锁\*/

#include<stdio.h>

#include<stdlib.h>

#include <unistd.h>

#include <sys/file.h>

#include <sys/types.h>

#include <sys/stat.h>

void lock\_set(int fd, int type)

{

struct flock lock;

lock.l\_whence = SEEK\_SET;

lock.l\_start = 0;

lock.l\_len =0;

while(1){

lock.l\_type = type;

if((fcntl(fd,F\_SETLK,&lock))==0){/\*根据不同的type值给文件加锁或解锁\*/

if( lock.l\_type == F\_RDLCK ) /\*F\_RDLCK为共享锁，表示读取锁或建议性锁\*/

printf("加上读取锁的是： %d\n",getpid());

else if( lock.l\_type == F\_WRLCK ) /\*F\_WRLCK为排斥锁，表示强制性锁\*/

printf("加上写入锁的是： %d\n",getpid());

else if( lock.l\_type == F\_UNLCK )

printf("释放强制性锁： %d\n",getpid());

return;

}

fcntl(fd, F\_GETLK,&lock); /\*读取文件锁的状态\*/

if(lock.l\_type != F\_UNLCK){

if( lock.l\_type == F\_RDLCK )

printf("文件已经加上了读取锁，其进程号是： %d\n",lock.l\_pid);

else if( lock.l\_type == F\_WRLCK )

printf("文件已加上写入锁，其进程号是： %d\n",lock.l\_pid);

getchar();

}

}

}

int main ()

{

int fd;

fd=open("/root/4-6file",O\_RDWR | O\_CREAT, 0666);

if(fd < 0)

{

perror("打开出错");

exit(1);

}

lock\_set(fd, F\_WRLCK);

getchar();

lock\_set(fd, F\_UNLCK);

getchar();

lock\_set(fd, F\_RDLCK);

getchar();

lock\_set(fd, F\_UNLCK);

close(fd);

exit(0);

}

（2）修改程序，如果把文件名作一个房号，请描述宾馆客人入住的情况。

#include<stdio.h>

#include<stdlib.h>

#include <unistd.h>

#include <sys/file.h>

#include <sys/types.h>

#include <sys/stat.h>

void lock\_set(int fd, int type)

{

struct flock lock;

lock.l\_whence = SEEK\_SET;

lock.l\_start = 0;

lock.l\_len =0;

while(1){

lock.l\_type = type;

if((fcntl(fd,F\_SETLK,&lock))==0){/\*根据不同的type值给文件加锁或解锁\*/

if( lock.l\_type == F\_RDLCK ) /\*F\_RDLCK为共享锁，表示读取锁或建议性锁\*/

printf("编号为%d的客人在查看房间101的使用情况！\n",getpid());

else if( lock.l\_type == F\_WRLCK ) /\*F\_WRLCK为排斥锁，表示强制性锁\*/

printf("编号为%d的客人占用了房间101！\n",getpid());

else if( lock.l\_type == F\_UNLCK )

printf("编号为 %d的客人退房成功！\n",getpid());

return;

}

fcntl(fd, F\_GETLK,&lock); /\*读取文件锁的状态\*/

if(lock.l\_type != F\_UNLCK){

if( lock.l\_type == F\_RDLCK )

printf(" 房间101已被编号为%d的客人占用，请查看其它房间！\n",lock.l\_pid);

else if( lock.l\_type == F\_WRLCK )

printf("编号为%d的客人在查看房间101的使用情况！\n",lock.l\_pid);

getchar();

}

}

}

int main ()

{

int fd;

fd=open("房间101",O\_RDWR | O\_CREAT, 0666);

if(fd < 0)

{

perror("该房间不存在哦！");

exit(1);

}

lock\_set(fd, F\_WRLCK);

getchar();

lock\_set(fd, F\_UNLCK);

getchar();

lock\_set(fd, F\_RDLCK);

getchar();

lock\_set(fd, F\_UNLCK);

close(fd);

exit(0);

}

1. **实验结果记录及分析**

**任务一**



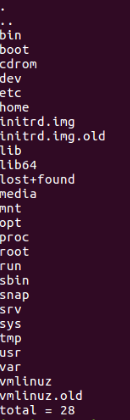


**任务二**



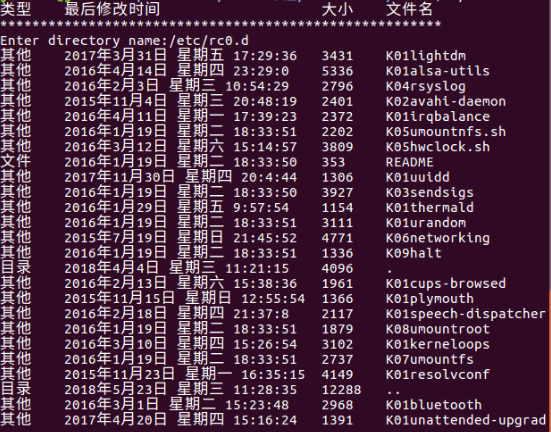
**任务三**



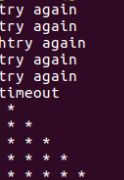


**任务四**





**任务五**





**任务六**

**(1)**

**A**



原因：源代码中访问的文件不存在，因此，显示“打开出错”的信息。

修改打开的文件路径为：“4-6file”,此时，打开文件成功。



**B**

第一个终端



第二个终端



**C**

终端一

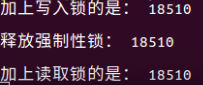


终端二

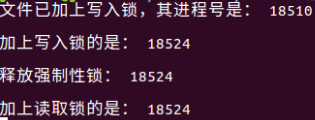


**D**

终端一



终端二



**(2)**

**A**

终端一



终端二

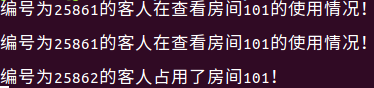


**B**

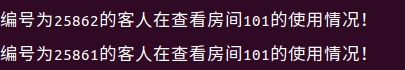
终端一



终端二



**C**



### 实验小结

（1）位置指针和文件指针的区别：

文件位置指针是指文件打开之后，在文件内部进行移动的指针；文件指针是指向一个文件的指针，确切的将是指向引用文件这个结构体所定义的对象的起始地址，文件指针的移动是指在文件之间来移动。

（2）在文件模式字（st\_mode）中，有两个位（bit10和bit11）分别称为设置用户组ID位和设置用户ID位，分别有两个测试常量S\_ISGID和S\_ISUID与其对应。若此文件为可执行文件：

1）当设置用户组ID位为1，则文件执行时，内核将其进程的有效用户组ID设置为文件的所有组ID。

2）当设置用户ID位为1，则文件执行时，内核将其进程的有效用户ID设置为文件所有者的用户ID。

（3）文件读写操作时常用到的函数是read,write和lseek。

read函数用于从指定的文件描述符中读出数据；

write函数用于向打开的文件写入数据；

lseek函数用于在指定的文件描述符中将文件指针定位到相应的位置。