电子科技大学

University of Electronic Science and Technology of China

Linux操作系统编程

实验三 | 文件与目录操作

主讲老师：杨珊

实验三｜文件与目录实验目的

目的一：掌握Linux目录操作方法

- ✓ 打开目录、关闭目录
- ✓ 读取目录文件

目的二：掌握Linux文件属性获取方法

- ✓ 获取Linux文件属性的函数

目的三：掌握文件属性解析相关的重要数据结构、函数、宏和文件掩码等

实验三｜文件与目录实验内容

仿写ls -l的功能（编写myls程序）,参数通过命令行传入：

1、获取当前工作目录路径并对该目录实现遍历

2、仿ls -l以列表形式出当前工作目录下的所有文件（包括子目录）

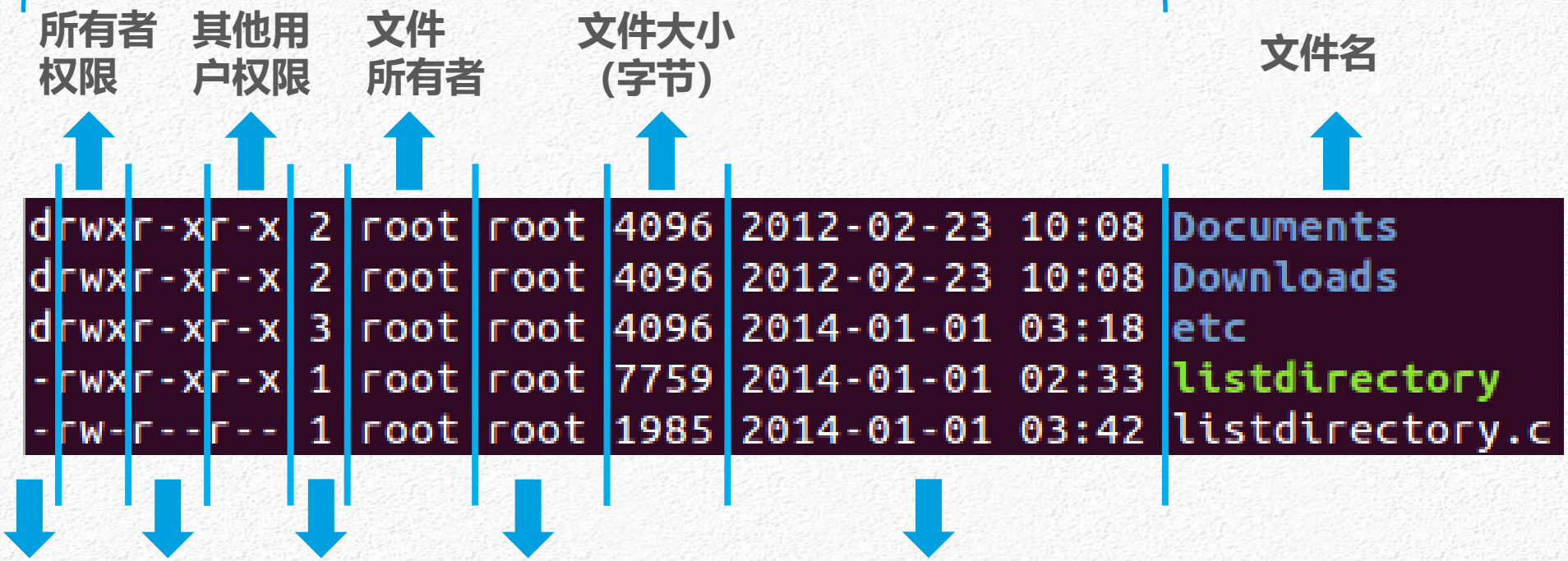
需显示的文件属性有：

文件类型 权限 硬链接数 所有者用户名 所有者所在组用户名 文件大小 最后修改时间

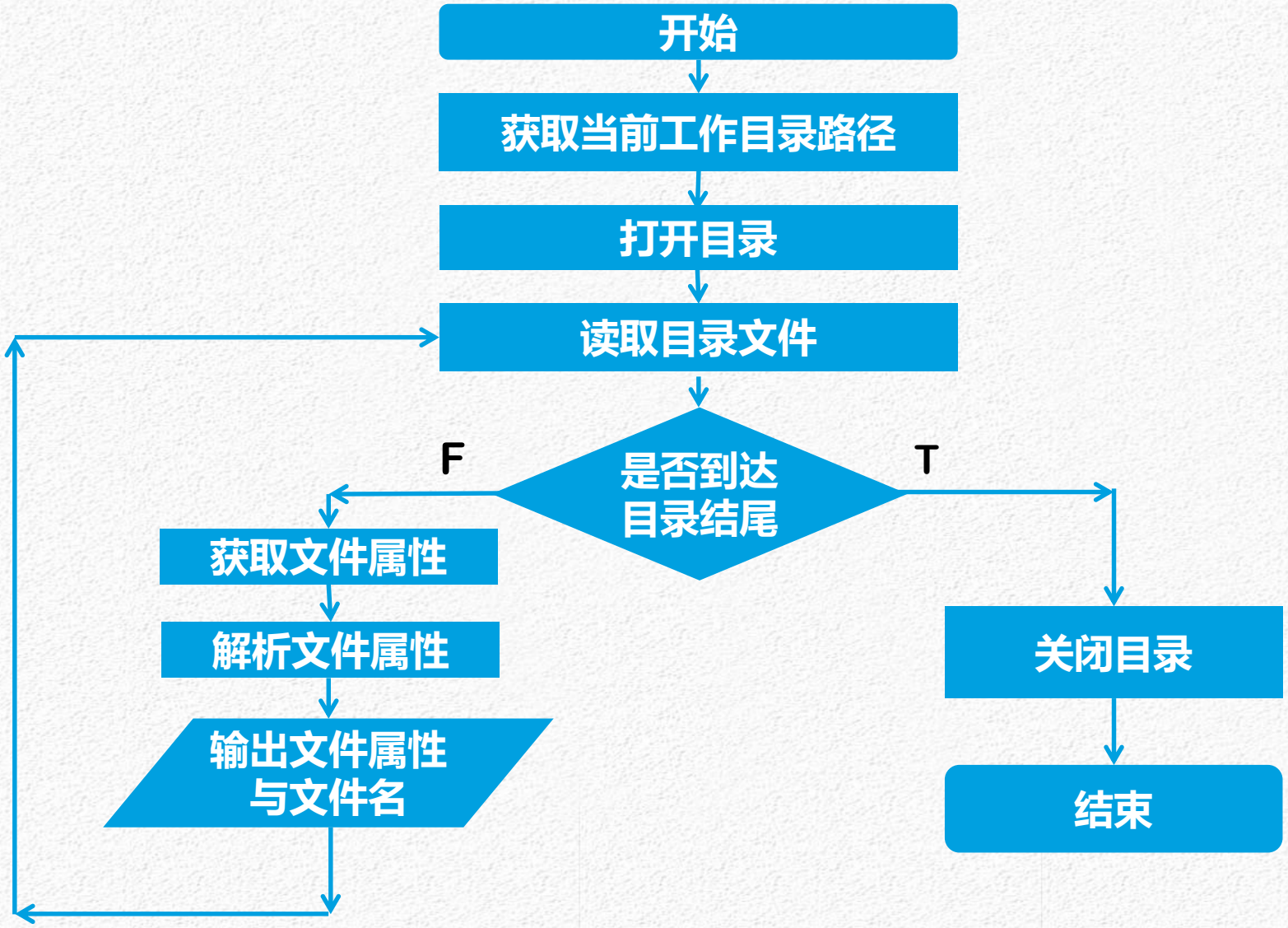
```
drwxr-xr-x 2 root root 4096 2012-02-23 10:08 Documents
drwxr-xr-x 2 root root 4096 2012-02-23 10:08 Downloads
```

实验三 | ls-l功能分析

文件属性



实验三 | ls -l功能分析


开始
获取当前工作目录路径
打开目录
读取目录文件
是否到达
目录结尾
F
T
获取文件属性
解析文件属性
输出文件属性
与文件名
关闭目录
结束

实验三 | 获取当前工作路径

头文件：unistd.h

函数定义：char *getcwd(char *buf, size_t size)

分配一个内存区buf，存储当前工作路径字符串。Size则是buf 大小（该内存区需要手动释放），失败返回NULL

函数定义：char *get_current_dir_name(void)

成功返回路径字符串缓冲区指针（该内存区需要手动释放），失败返回NULL

实验三 |打开关闭目录函数

常用函数：opendir，closedir

头文件：dirent.h

函数定义：DIR *opendir(const char * name);

打开name指定的目录，并关联一个目录流（类似于C库函数中的文件流）

失败返回NULL

函数定义：int closedir(DIR *dir);

关闭指定目录流，释放相关数据结构

成功返回0；失败返回-1

实验三｜读取目录文件函数

头文件：sys/types.h; dirent.h

函数定义：struct dirent * readdir(DIR *dir) 读取目录流dir标识的目录下文件

到达文件结尾或者错误发生，返回NULL

```
If(currentdir = opendir(buf))==NULL)
{    printf("open directory fail \n");
      return 0;     }
else {      printf("file in directory include: \n")
            while((currentdp = readdir(currentdir)!=NULL)
            printf("%s\n",currentdp->d_name);  }
```

实验三｜读取目录文件函数｜dirent结构

重要数据结构

```
struct dirent
{
    ino_t d_ino;          i节点号
    off_t d_off;          在目录文件中的偏移
    usigned short d_reclen; 文件名长度
    unsigned char d_type;   文件类型
★   char d_name[256];       文件名
};
```

```
printf( “%s\n” ,currentdp->d_name);
```

```
10:05 .
11:25 ..
15:44 .android
17:01 Applications
10:01 Applications (Parallels)
17:48 .bash_history
16:55 .bash_profile
16:02 .bash_profile-anaconda3.bak
15:54 .bash_sessions
23:21 .CFUserTextEncoding
21:29 CLionProjects
14:22 .config
10:13 Creative Cloud Files
11:08 .cups
```

实验三 | 获取文件属性

常用函数：stat，lstat

头文件： sys/stat.h

函数定义：int stat(const char *path, struct stat *buf);

int lstat(const char *path, struct stat *buf);

函数作用：读取path参数**指定文件**的**文件属性**，填充到buf参数指向的结构体

区别：lstat返回符号链接的文件属性

stat返回符号链接引用文件的文件属性

实验三｜存储文件属性的stat结构

```
struct stat {
    mode_t     st_mode;     文件类型与访问权限
    ino_t      st_ino;      i节点号
    dev_t      st_dev;      文件使用的设备号
    dev_t      st_rdev;     设备文件的设备号
    nlink_t    st_nlink;    文件的硬链接数
    uid_t      st_uid;      文件所有者用户ID
    gid_t      st_gid;      文件所有者组ID
    off_t      st_size;     文件大小（以字节为单位）
    time_t     st_atime;    最后一次访问该文件的时间
    time_t     st_mtime;    最后一次修改该文件的时间
    time_t     st_ctime;    最后一次改变该文件状态的时间
    blksize_t st_blksize;   包含该文件的磁盘块的大小
    blkcnt_t   st_blocks;   该文件所占的磁盘块数
};
```

★ st_mode　★ st_nlink　★ st_uid　★ st_gid　★ st_size　★ st_mtime

重要数据结构

实验三｜解析st_mode的内容

mode_t st_mode;

无符号整数，其低16位定义如下

```
-rwxr-xr-x 1 root root
drwxr-xr-x 2 root root
drwxr-xr-x 2 root root
```

Bit	15	14	13	12	4	3

文件类型域　文件特殊属　权限域 其

标识	文件类型
-	普通文件
d	目录文件
c	字符设备文件
b	块设备文件
p	管道或FIFO
l	符号链接
s	套接字

标识	文件访问权限
r	读权限
w	写权限
x	执行权限

实验三｜解析st_mode的宏

宏: S_XXXX(file)

- ✓ 是否为普通文件: S_ISREG(st_mode)
- ✓ 是否为目录文件 S_ISDIR(st_mode)
- ✓ 是否为字符设备 S_ISCHR(st_mode)
- ✓ 是否为块设备 S_ISBLK(st_mode)
- ✓ 是否为FIFO S_ISFIFO(st_mode)
- ✓ 是否为套接字 S_ISSOCK(st_mode)
- ✓ 是否为符号连接 S_ISLINK(st_mode)

例:

```
if(S_ISREG(buf.st_mode))
        ptr = "regular";
```

从st_mode中

如何得到文件类型?

实验三｜解析st_mode代码示例

```
int main(int argc, char *argv[])
{
        int   i;
        struct stat  buf;
        char   *ptr;
        for (i = 1; i < argc; i++) {
                printf("%s: ", argv[i]);
                if (lstat(argv[i], &buf) < 0) {
                        err_ret("lstat error");
                        continue;
                }
```

```
if(S_ISREG(buf.st_mode))  ptr = "regular";
else if (S_ISDIR(buf.st_mode))  ptr = "directory";
else if (S_ISCHR(buf.st_mode))  ptr = "character special";
else if (S_ISBLK(buf.st_mode))  ptr = "block special";
else if (S_ISFIFO(buf.st_mode)) ptr = "fifo";
else if (S_ISLNK(buf.st_mode))  ptr = "symbolic link";
else if (S_ISSOCK(buf.st_mode)) ptr = "socket";
else ptr = "** unknown mode **";
 printf("%s\n", ptr); }
        exit(0);
}
// end main
```

实验三｜解析st_mode的宏

例：

```
if ((S_IRUSR & st_mode) == S_IRUSR)
    {
        printf("r");
    }
```

✓ S_IRWXU	00700	
✓ S_IRUSR	00400	
✓ S_IWUSR	00200	
✓ S_IXUSR	00100	
✓ S_IRWXG	00070	用户组的权限值组合
✓ S_IRGRP	00040	用户组具可读取权限
✓ S_IWGRP	00020	用户组具可写入权限
✓ S_IXGRP	00010	用户组具可执行权限
✓ S_IRWXO	00007	其他用户的权限值组合
✓ S_IROTH	00004	其他用户具可读取权限
✓ S_IWOTH	00002	其他用户具可写入权限
✓ S_IXOTH	00001	其他用户具可执行权限

从st_mode中

如何获取读写权限

实验三｜获取用户属性函数

常用函数：getpwuid

头文件： sys/types.h，pwd.h

函数定义：

struct passwd *

输入用户ID，返回用户属性信息（pa

```
struct passwd{
    char *pw_name;       /* 用户名*/
    char *pw_passwd;     /* 密码.*/
    __uid_t pw_uid;      /* 用户ID.*/
    __gid_t pw_gid;      /*组ID.*/
    char *pw_gecos;      /*真实名*/
    char *pw_dir;        /* 主目录.*/
    char *pw_shell;      /*使用的shell*/};
```

实验三｜获取用户组属性函数

常用函数：getgrgid

头文件： sys/types.h，grp.h

函数定义：

struct group *g

输入用户组ID，返回用户组属性

```
struct group{
  char *gr_name; /*组名称*/
  char *gr_passwd; /* 组密码*/
  gid_t gr_gid; /*组ID*/
  char **gr_mem; /*组成员账号*/ }
```

实验三｜获取文件时间信息

常用函数：localtime，ctime

头文件： time.h

函数定义：struct tm* localtime(const time_t *lock)

localtime返回tm结构体指针（存储时间的年月日各个量）

char* ctime(const time_t *timep)

ctime直接解析为当地时间格式例如“wed jun 30 21:49:08 1993\n”

实验三｜linux编程技巧｜缓冲区分配

```
char *buffer=NULL;
buffer=(char *)malloc(100*sizeof(char));
...
free(buffer);
buffer=NULL;
```

申明全局指针变量

malloc分配空间，free释放空间

堆

有函数

进程生命周期

malloc之后，free之前

```
uffer[100]
nt buffer[100];
uffer为缓冲区
```

注： 局部数组变量和指针变量的比较大同小异，只是作用范围及生命周期都缩减为申明函数之内

实验三 | 实验框架代码

```
56
57  if(1. 获取当前目录路径!= NULL)
58    printf("%s\n",buf);
59  if(    2. 打开当前目录，并判断是否成功    )
60  {
61    printf("open directory fail\n");
62    return 0;
63  }
64  while(    3. 读取当前目录下的目录项，并判断是否成功    )
65  {
66    if(currentdp->d_name[0] != '.')
67     {
68        if( 4. 读取当前文件的信息，并判断是否成功          )
69        {
70            printf("get stat error\n");
71            continue;
72        }
73        print_type(currentstat.st_mode);
74        print_perm(currentstat.st_mode);
75        print_link(currentstat.st_nlink);
76        print_usrname(currentstat.st_uid);
77        print_grname(currentstat.st_gid);
78        print_time(currentstat.st_mtime);
79        print_filename(currentdp);
80     }
81  }
82  6. 关闭当前目录
83  return 0;
```

5. 补充完整这7个打印文件信息的函数

实验三｜文件与目录实验扩展

基本功能实现的基础上，针对各种**特殊情况**和**边界条件**等进行流程的完善与优化，包括：

- ✓ 程序接收命令行参数，决定是否显示当前**目录本身“.”**和**上级目录“..”**
- ✓ 程序接收命令行参数，决定是否显示隐藏文件（文件名以“.”作为开始的文件）
- ✓ 程序接收命令行参数，决定是否显示**符号链接文件本身的属性**还是**符号链接指向文件的文件属性**

电子科技大学

University of Electronic Science and Technology of China

Linux操作系统编程

感谢观看

主讲老师：杨珊