第4章 UNIX的文件和目录

4.1 文件和目录的层次结构

UNIX系统通过目录管理文件，文件系统组织成树状结构，目录中可以含有多个文件，也可以含有子目录。UNIX系统中，路径名分割符用正斜线/。表4-1所示为一些常见的目录和文件。与系统有关的一些主要目录的取名和在层次结构中的位置，几乎在所有UNIX系统中都相同。

表4-1 UNIX常见的目录和文件

路　　径	说　　明
/unix	程序文件，UNIX 内核
/etc	供系统维护管理用的命令和配置文件。例如： 文件/etc/passwd:存放的是用户相关的配置信息； 文件/etc/issue:登录前在 login 之上的提示信息； 文件/etc/motd:存放登录成功后显示给用户的信息。 文件系统管理的程序有 fsck，mount，shutdown 等。 许多系统维护的命令，在不同的 UNIX 系统之间区别很大
/tmp，/usr/tmp	存放临时文件
/bin	系统常用命令，如 ls，ln，cp，cat 等
/dev	存放设备文件，如终端设备文件，磁带机，打印机等
/usr/include	C 语言头文件存放目录
/usr/bin	存放一些常用命令，如 ftp，make 等
/lib,/usr/lib	存放各种库文件，包括 C 语言的链接库文件，动态链接库，还包括与终端类型相关的 terminfo 终端库，等等。静态链接库文件有.a 后缀，动态链接库文件的后缀不是.DLL，而是.so。UNIX 很早就广泛地使用动态链接库，静态链接库逐渐过时。.a 取名于 archive（存档），.so 取名于 shared objects（共享对象）
/usr/spool	存放与用户有关的一些临时性文件,如：打印队列,已收到但未读的邮件等

4.2 文件和目录的命名

（1）名字长度。现在的UNIX都支持长文件名，文件名长度的最大值都在200以上，早期的UNIX至少可以支持长度为14个字符的文件名。

（2）取名的合法字符。除斜线外的所有字符都是命名的合法字符，空格、星号甚至不可打印字符也可以做文件名。一个字节的取值0~255之中，47是斜线的ASCII码，不可作为文件名，ASCII码0用作C语言的字符串结束标志，其余的254种取值都可以作为文件名。

（3）大小写字母有区别。例如：makefile，Makefile，MAKEFILE是三个不同的文件名。

4.3 shell的文件名通配符

4.3.1 规则

- UNIX的文件名通配符是由shell程序解释的。
- 对几乎所有的shell来说，表4-2列出的有关文件名通配符的规则几乎都一致。

表4-2 常用的shell文件名通配符

符号	含　义
*	匹配任意长度（包括空字符串）。如：try*c 匹配 try1.c，try.c，try.basic。例外的情况是当文件通配符的第一个字符为*时，*不匹配以句点（.）开头的文件。例：*file 匹配文件 file，tempfile，makefile，但不匹配.profile 文件
?	匹配任一单字符。例如：p?.c 可以匹配 p1.c，p2.c，pa.c
[]	匹配括号内任一字符，也可以用减号指定一个范围。例如：[A-Z]*匹配所有名字以大写字母开头的文件，*.[ch]匹配所有含.c 或者.h 后缀的文件，[Mm]akefile 匹配 Makefile 或者 makefile

4.3.2 与DOS文件名通配符的区别

- UNIX的文件名通配符要无二义性。
- 在UNIX中，文件名通配符允许用于任何命令，而DOS中只能用于dir/del/copy等有限的几个命令中。
- 关于文件扩展名。DOS中*.*匹配所有文件，UNIX中*.*要求文件名中必须含有句点，否则不匹配。
- 匹配子目录中的文件。在UNIX中可以使用*/*.[ch]通配符，匹配当前目录下所有一级子目录中文件名后缀为.c和.h的文件，这在DOS中不允许。

4.3.3 文件名通配符的处理过程

UNIX处理文件名通配符的过程分三步：

（1）在shell提示符下，从键盘输入命令，输入的命令被shell所接受。

（2）shell对所键入的内容作若干种加工处理,其中含有对文件名通配符的扩展工作，生成结果命令。

（3）调用操作系统的系统调用，创建新的进程执行命令，并把参数传递给新进程，执行生成的结果命令。

shell在第二步中，含有文件名生成工作，把用空格分开的每一段作为一个“单词”，扫描每个词，从中寻找 * ？ []。

- 如果其中之一出现，则该词被识别为一个文件名通配符，用与文件名通配符相匹配的文件名表取代该词。
- 可以匹配多个名字时，按字母序排列多个名字。
- 如果没有找到与文件名通配符相匹配的文件名，在**B-shell**中不改变该词，在**C-shell**中产生错误。

【例4-1】 体验shell对文件名通配符的展开处理。

（1）设当前目录下只有try.c，zap.c，arc.c三文件，在shell提示符下，键入命令：

cat *.c

- shell根据当前目录下的所有文件的文件名集合，将*.c扩展为arc.c try.c zap.c，扩展后的多个文件名按照字典序排列。
- 这样，cat *.c被加工成了cat arc.c try.c zap.c，实际执行加工之后的命令。
- 从cat命令的角度来说，都是指定了三个文件作为处理对象，cat程序在执行的时候，已经看不到*.c，它看到的是三个文件名。

（2）设当前目录下有四个文件0131.rpt，0130.rpt，wang.mail，lee.mail。在两个mail文件中查找数字串的命令为：

grep '[0-9][0-9]*' *.mail

（3）使用文件名通配符，可以简化一些命令的输入，尤其是那些较长的名字。

如：vi m*e 替换成 vi makefile

cd *work.d替换成cd configure_network.d。

(cd后面只能有一个参数，此时只能匹配一个名字)

4.3.4 验证文件名匹配的结果

【例4-2】 从程序员的角度理解shell对通配符的处理。

编写一个很简短的C语言程序文件arg.c。

```
int main(int argc, char *argv[])
{
    int i;
    for (i = 0; i < argc; i++)
        printf("%d: [%s]\n", i, argv[i]);
}
```

编译、链接以生成可执行文件。使用命令：

cc arg.c -o arg

或者使用命令：

make arg

在当前目录下生成名为**arg**的可执行文件，没有**Windows**系统中那样的**.EXE**扩展名。执行这个程序的命令，如：

./arg abc ABCDEF

- 在UNIX系统中，如果直接键入arg abc ABCDEF，默认情况下，shell会只在系统规定的目录中搜索指定文件arg，搜索不到也不会到当前目录下搜索。
- 需要用命令./arg显式地指定程序文件存储的路径为点目录(./)，shell就会到./目录下搜寻执行文件arg。
- 用户可以通过设置shell环境变量PATH将当前目录增加到系统的搜索路径中去。一般默认情况下不搜索当前目录。

针对输入的./arg abc ABCDEF，argv数组的布局如图4-1所示。在C语言中，从主函数main的两个参数，可以获得命令行参数的内容。

- 第一个参数argc是命令行参数的个数
- 第二个参数argv是一个指向数组的指针，数组的每个元素是个指针，指向一个字符串，argv[0]字符串是命令自身，其余的是命令行的参数。

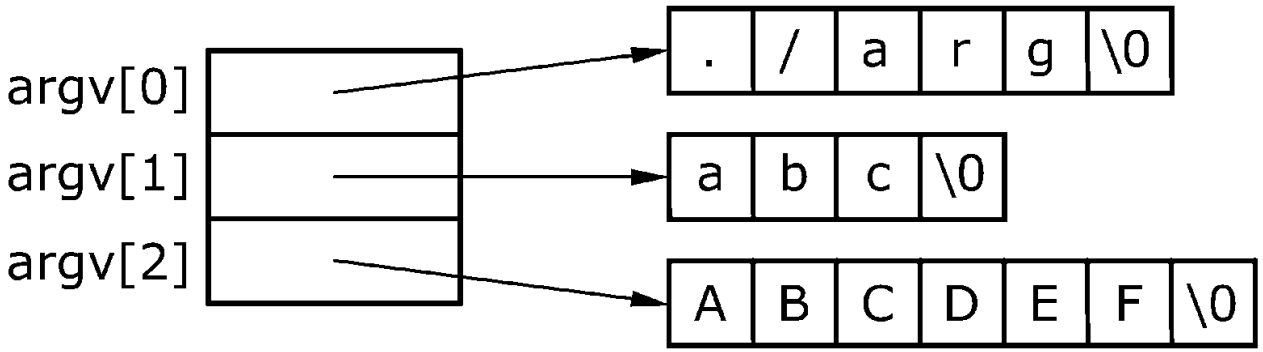

图4-1 main函数的argv参数的存储结构

```
$ cat arg.c
int main(int argc, char *argv[])
 {
    int i;
    for (i = 0; i < argc; i＋＋)
       printf("%d: [%s]\n", i, argv[i]);
 }

$ cc arg.c -o arg
$ ./arg abc ABCDEF
0: [./arg]
1: [abc]
2: [ABCDEF]
```

$ ./arg *.[ch]

0: [./arg]

1: [arg.c]

2: [auth.c]

3: [ccp.c]

4: [chap.c]

…

$ ./arg '*.[ch]'

0: [./arg]

1: [*.[ch]]

将执行结果与同样的arg.c在DOS下执行结果相比较，在DOS系统中，.\arg　*.c执行时，不会进行文件名展开，执行结果为：

0: [.\arg]

1: [*.c]

4.4 文件管理

4.4.1 ls:文件名列表

列出目录中的文件名。ls之后可以跟0个到多个名字。

- 不给出任何名字时，列出当前目录下所有文件和子目录。
- 名字为文件时，列出文件名。
- 名字为目录时，不列出目录名，而是列出目录下的所有文件和子目录。
- 在同一命令行中可以指定多个名字，这点可以配合shell文件名通配符工作。

例如：

$ ls

（ls未指定名字，列出当前目录下的所有文件和子目录名）

```
arg       bak.d     document   pipe1      xsh2.c
arg.c     config.ap  inc        xsh2
```

$ ls arg.c

（为ls指定一个实参，是普通文件）

```
arg.c
```

$ ls /

（为ls指定一个实参，是目录，列出根目录下的所有文件和目录）

```
TT_DB            ftphome1         lpp            tftpboot
WebSM.pref       ftphome2         mbox           tmp
audit            ftphome3         mnt            u
bin              home             nsmail         unix
bosinst.data     image.data       opt            usr
cdrom            inst.log         proc           var
dead.letter      lab              sbin           websm.log
dev              lib              smit.log       websm.script
etc              lost+found       smit.script    wsmmon.dat
```

$ ls arg*

（为ls指定多个实参，均为普通文件）

arg arg.c

$ ls /usr/include/*

（为ls指定多个实参，既有文件也有目录）

/usr/include/60x_regs.h /usr/include/lvm.h
/usr/include/NLchar.h /usr/include/lvmrec.h
/usr/include/NLctype.h /usr/include/macros.h
/usr/include/NLregexp.h /usr/include/malloc.h
/usr/include/NLxio.h /usr/include/math.h
/usr/include/a.out.h /usr/include/mbstr.h
/usr/include/acl.h /usr/include/memory.h
/usr/include/aio.h /usr/include/mesg.h
⋮

/usr/include/Motif2.1:
Dt Mrm Xm uil

/usr/include/Mrm:
MrmAppl.h MrmDecls.h MrmPublic.h MrmWidget.h MrmosI.h

ls有几十个选项，控制每个文件的列表格式，以及列表的范围包括哪些文件。

（1）-a：列出所有（all）项，包括以句点打头的文件，默认情况下，名字以句点打头的文件不被列出。

```
$ ls -a
.        .cshrc    arg       bak.d     document  pipe1     xsh2.c
..       .profile  arg.c     config.ap inc       xsh2
```

（2）-R：递归地列出碰到的子目录（Recursion）。如：ls -R / 将列出系统中所有文件，这一命令执行时间会很长，列出的内容也很多。

```
$ ls -R .
arg       bak.d     document   pipe1     xsh2.c
arg.c     config.ap  inc        xsh2
./bak.d:
1s.c      cld.c     cld2.c     client.c  clock.c

./document:
manual.pdf  paper.pdf
```

（3）-F：标记（Flag）每个文件。

- 若列出的是目录，就在名字后面缀以/；
- 若列出的是可执行文件，就在名字后面缀以*；
- 若列出的是符号连接文件，就在名字后面缀以@；
- 若列出的是管道文件，则名字后面缀以|；
- 若列出的是普通文件，则名字后面无任何标记。

ls命令允许同时指定多个选项，ls -aF命令就是同时使用两个选项a和F。

```
$ ls -F
arg*       bak.d/     document/   pipe1|     xsh2.c
arg.c      config.ap* inc@        xsh2*
$ ls -aF
./         .profile   bak.d/      inc@       xsh2.c
../        arg*       config.ap*  pipe1|
.cshrc     arg.c      document/   xsh2*
```

（4）-i：列出文件的i节点号。例如：

```
$ ls -i
184323 arg      184393 config.ap 184326 pipe1
184321 arg.c    206873 document  184391 xsh2
184327 bak.d    184325 inc       184392 xsh2.c
```

（5）-d：若实参是目录，则只列其名字（不列内容）。

例如：设当前目录结构如图4-2所示。

```
$ ls *
abc      abc.rpt

abc.dir:
f1  f2
$ ls -d abc*
abc      abc.dir  abc.rpt
```

无参数的ls命令，与ls *执行结果并不同，与ls -d *功能相同。

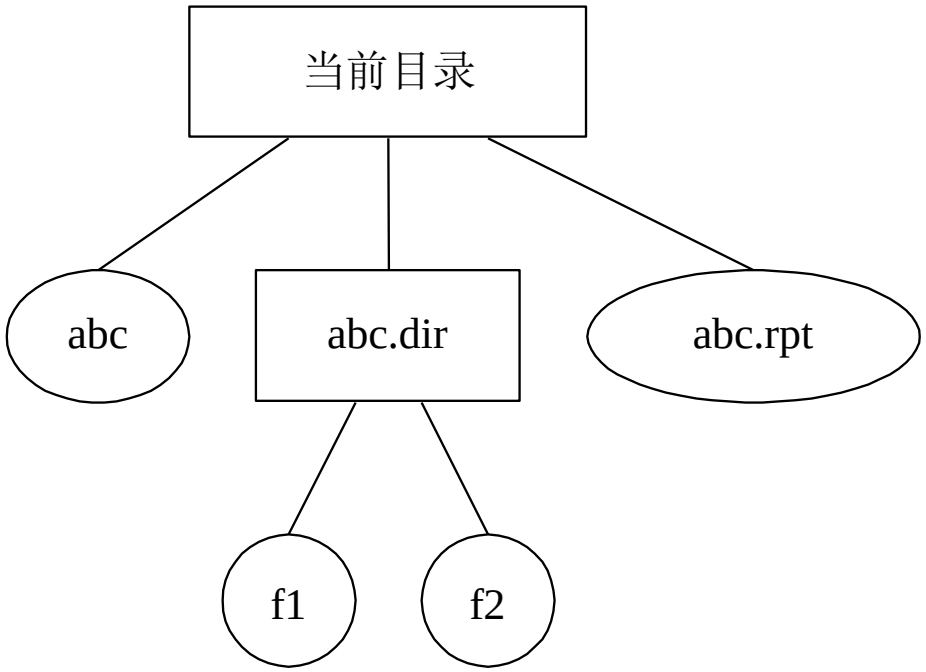

图4-2 文件和目录结构举例，展示ls的-d选项的功能

（6）-l：长格式（long）列表。例如：

$ ls -l

```
-rwxr-xr-x    1 jiang   usr    4423   Aug26  13:31  arg
-rw-r--r--    1 jiang   usr     116   Aug26  13:31  arg.c
drwxr-x---    2 jiang   usr    1536   Aug26  14:00  bak.d
-rwxr-xr-x    1 jiang   usr     128   Aug26  13:38  config.ap
drwxr-x---    2 jiang   usr     512   Aug26  13:59  document
lrwxrwxrwx    1 jiang   usr      12   Aug26  13:33  inc -> /usr/include
prw-r--r--    1 jiang   usr       0   Aug26  13:33  pipe1
-rwxr-xr-x    1 jiang   usr    8661   Aug26  13:36  xsh2
-rw-r--r--    1 jiang   usr    1076   Aug26  13:36  xsh2.c
```

- 第1列为文件属性。

第1列的第1字符描述文件类型。常见的文件类型有：- 普通文件，d目录文件，l 符号连接文件，b块设备文件，c 字符设备文件，p（pipe）命名管道文件。

第1列的第2~10个字符，描述文件的访问权限。其中第2~4个字符描述文件所有者对文件的访问权限；第5~7个字符描述文件所有者的同组用户对文件的访问权限；第8~10个字符描述其他用户对文件的访问权限。权限描述方式为rwx，分别对应读权限,写权限,可执行权限,相应位置显示的是字母，表明有相应权限，显示减号，表明没有相应的权限。

- 第2列是文件的link数，涉及此文件的目录项数。

- 第3列，第4列分别是文件主的名字和组名。
- 第5列是文件的大小。对于普通磁盘文件，列出文件内容大小；对于目录，列出目录表大小（而不是目录下各文件长度之和）；对于符号连接文件，列出符号连接文件自身的长度；对于字符设备和块设备文件，列出主设备号和次设备号；对于管道文件，列出当前管道内的数据长度。
- 第6列是文件最后一次被修改的日期和时间。
- 第7列是文件名。对于符号连接文件，还附带列出符号连接文件的内容。

再如：

ls -l 以长格式列出当前目录下所有文件

ls -l ·列出当前目录下所有文件，可查知各文件权限

ls -ld ·列出当前目录自身，可查知当前目录自身的权限

ls -l * 列出当前目录下所有文件（但不包含目录）和一级子目录中所有文件名

ls -Flad rpt* 可以在同一命令中指定多个选项

ls -l | grep '^d' | wc -l 统计当前目录有多少子目录

4.4.2 cp:复制文件

复制文件的命令格式如下。

格式1: cp *file1* *file2*

格式2: cp *file1* *file2* ... *filen* *dir*

其中，*file1*，*file2*，...，*filen*为文件名（*n*>0），*dir*为已有目录的名字。

- 命令的第一种格式把文件*file1*复制到*file2*。若*file2*存在，则覆盖，否则，创建*file2*。
- cp命令的第二种格式，将*file1*~*filen*一个或多个文件复制到目录*dir*下。当*n*为1时，第二种格式的命令看起来和第一种格式相同，但是完成的操作不同。

例如：

cp *.c backup.d

其中 **backup.d** 为一个子目录,符合第二种格式。

cp fa.c backup.d命令也符合上述的格式2，把文件**fa.c**复制到现有的一个目录**backup.d**中。

【例4-3】 设文件目录结构如图4-3所示，将backup.d下的两个文件p1.c和p2.c复制到当前目录。

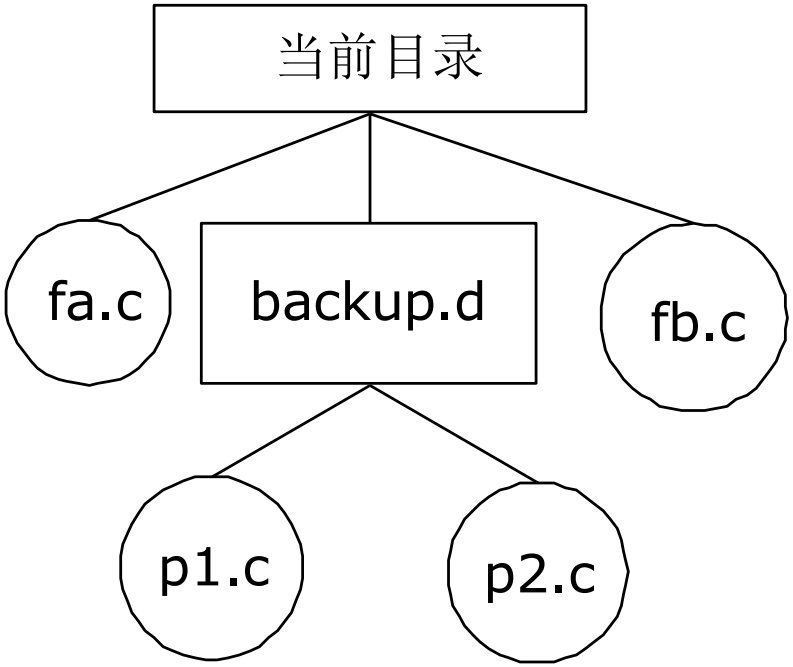

图4-3 文件和目录结构举例，展示cp命令的处理过程

- 在UNIX中执行cp backup.d/p?c命令，经过shell完成文件名生成之后，实际执行：

  cp backup.d/pl.c backup.d/p2.c

  执行结果是p1.c和p2.c没有复制到当前目录,而是文件p1.c覆盖掉文件p2.c。

- 如果运气好的话，backup.d目录下还有p3.c，展开后cp有三个参数，既不符合格式1，也不符合格式2，命令无效，拒绝做任何操作。

- 将上述两文件复制到当前目录的方法，使用cp命令的格式2，最后一个参数句点是当前目录:

  cp backup.d/p?.c .

  再如：

  cp /usr/include/*.h .

4.4.3 mv:移动文件

格式1：mv *existing-file-or-dir new-name*

格式2：mv *file1 file2 ... filen directory*

- mv命令的第一种格式可以将文件和目录改名，或移动到另一个目录，并使用新名字。
- 第二种格式，*directory*是一个已存在的目录，可以把一个或多个已有的文件或目录移动到目录*directory*中，并保持原先的名字。其中的*file1~filen*可以是已有文件，也可以是已有的目录。

在同一个文件系统内的mv操作比先复制成新文件再删除旧文件的方法更高效，因为mv操作只需要修改目录表，不需要访问文件内容。

4.4.4 rm:删除文件

格式：**rm *file1 file2 ... filen***

例如：**rm core a.out**

rm *.o *.tmp

（1）选项-r：递归删除（recursively）。允许***file1~filen***是已有的文件名或者目录名。当它是一个目录时，递归地删除子目录中的所有文件和目录。经常使用这一命令删除一棵已有的目录树。

（2）选项-i：交互方式（interactive）。每次删除前，经过操作员确认。

（3）选项-f：强迫删除（force）。只读文件也可以被删除。

【例4-4】 rm命令选项的功能。

（1）rm -r backap.d

删除当前目录下的整个子目录backup.d。

（2）rm -rf xx*

清除所有以xx打头的文件。程序员应当为一段时间内临时使用的一些文件的取名符合某一特定规律。

（3）rm -i *.test

有选择地删除若干文件。

（4）rm * .bak

误操作，星号之后多出了一个空格，那么，经shell文件名展开之后，rm会忠实地删除所有文件，并且可能会通知用户，企图删除的文件.bak不存在。

【例4-5】处理以减号（-）打头的文件。

设当前目录下只有a，b，c这3个文件。

- 执行命令rm -i：rm会将-i理解为命令选项，没有指定文件名，不能删除任何文件。

执行命令who > -i，将生成文件-i，因为这个名字符合文件命令规则。

- rm –i： 不能删除文件-i。
- 如果使用命令rm *： 文件名通配符展开后的命令变成rm -i a b c，那么，rm将提示删除文件a，b或c，惟独不提示删除文件-i。
- 如果使用命令ls -l *：文件名通配符展开后的命令变成ls -l -i a b c，那么，ls命令以长格式列出a，b，c这3个文件并附带其i节点号，但不列出-i文件。

- **UNIX的许多命令，都允许使用选项，选项是可选择的，并不是每次使用该命令时都需要所有选项。按照惯例，选项都以减号开头，**
- **用一个独立的命令行参数--显式地标志命令行选项的结束，从--参数之后的任何命令行参数，都不再解释为选项。这样命令就可以识别以“-”开头的文件名。**

命令 1、 rm -- -i 删除文件-i。

2、rm ./-i删除当前目录下的文件-i。

4.4.5 find:查找文件

- find命令在一个指定的范围内查找符合条件的文件或者目录，然后执行相应的动作。
- find从指定的路径开始,递归地查找其下属的所有子目录,凡满足条件的文件或目录,执行规定的动作。
- find功能很强，描述“条件”和“动作”选项较多。

例如：

find verl.d ver2.d -name '*.c' -print

对于在规定范围内找到的符合条件的文件，执行的动作是把查找到的文件的路径名打印出来。

1. 关于“条件”的选项

（1）-name文件名的匹配，允许使用文件名通配符*、?和[]，应当把这个文件名通配符描述串传递到find程序中，因此，应当用引号括起来，以免被shell展开。

（2）-type 类型f：普通文件，d：目录，l：符号连接文件，c：字符设备文件，b：块设备文件，p：管道文件，如：-type d。

（3）-size ±n[c]文件大小，正号表示大于，负号表示小于，无符号表示等于，其他与数量有关的选项，也采用这样的方式。例如：

-size +100　　表示长度大于100块

-size -100　　长度小于100块

-size +100000c 表示长度大于100000字节

-size -100000c 表示长度小于100000字节

（4）-mtime ±n　文件最近修改（modify）时间，单位为天。例如：

-mtime -10，表示10天之内曾经修改过

（5）-atime ±n　文件最近访问（access）时间，单位为天。文件“访问”指读取了文件的内容，或者文件作为一个程序被执行。仅仅写文件，文件的“访问”时间不变。

- **find还有许多其他的条件选项，可以指定文件主（-user），组（-group），文件的link数（-links），i节点号（-inum）等。**
- **find中可以指定多个条件，罗列出的多个条件默认为多条件的“与”。**
- **find可以用!和-o表示条件“非”和两条件的“或”，可以使用括号表示更复杂的复合条件。**

2．关于“动作”的选项

（1）-print　打印查找到的符合条件的文件的路径名。

（2）-exec，-ok　对查找到的符合条件的文件执行某个命令。

find命令由于可以用-exec和-ok选项引入其他的命令，允许对搜索到的文件执行所需要的操作。多命令的这种组合，使得find命令可以配合其他命令提供灵活又强大的功能。

【例4-6】几个使用find命令的例子。

（1）find . -type d -print

从当前目录开始查找，仅查找目录，找到后，打印路径名。这种方法可以按层次列出当前的目录结构。

（2）find / -name 'stud*' -type d -print

指定了两个条件：名字与stud*匹配，类型为目录。这是两个条件的“逻辑与”，同时符合这两个条件的项目，打印路径名。

（3）find / -type f -mtime -10 -print

从根目录开始检索最近10天之内曾经修改过的普通磁盘文件。

（4）find . -atime +30 -mtime +30 -print

从当前目录开始检索最近30天之内既没有读过，也没有写过，而且也没有被当作命令执行过的文件。这种方法可以筛选出一个时间周期内不活跃的文件。

（5）find . ! -type d -links +2 -print

从当前目录开始检索link数大于2的非目录文件。

条件的“非”用！。注意，!号与-type之间必须保留一空格。

（6）find / -size +100000c \(-name core -o -name '*.tmp' \) -print

从根目录开始检索那些文件尺寸大于100KB，并且文件名叫core或者文件名有.tmp后缀的文件。在这个命令中，使用了两条件的“或”（-o）及组合（括号）。注意，不要遗漏了所必需的引号、反斜线和空格，尤其是括号前和括号后。上述命令也可以写作下面的形式。

find / -size +100000c '(' -name core -o -name *.tmp ')' -print

find / -size +100000c \(-name core –o -name *.tmp ')' –print