

基础牢，

建筑高，

成就！

LINUX 指令大全

树

阳

1) A 类指令

名称 at

使用权限：所有使用者

使用方式：at -V [-q queue] [-f file] [-mldbv] TIME

说明：at 可以让使用者指定在 TIME 这个特定时刻执行某个程式或指令，TIME 的格式是 HH:MM 其中的 HH 为小时，MM 为分钟，甚至你也可以指定 am, pm, midnight, noon, teatime(就是下午 4 点钟)等口语词。

如果想要指定超过一天内的时间，则可以用 MMDDYY 或者 MM/DD/YY 的格式，其中 MM 是分钟，DD 是第几日，YY 是指年份。另外，使用者甚至也可以使用像是 now + 时间间隔来弹性指定时间，其中的时间间隔可以是 minutes, hours, days, weeks

另外，使用者也可指定 today 或 tomorrow 来表示今天或明天。当指定了时间并按下 enter 之后，at 会进入交谈模式并要求输入指令或程式，当你输入完后按下 ctrl+D 即可完成所有动作，至于执行的结果将会寄回你的帐号中。

把计：

-V：印出版本编号

-q：使用指定的伫列(Queue)来储存，at 的资料是存放在所谓的 queue 中，使用者可以同时使用多个 queue，而 queue 的编号为 a, b, c... z 以及 A, B, ... Z 共 52 个

-m：即使程式/指令执行完成后没有输出结果，也要寄封信给使用者

-f file：读入预先写好的命令档。使用者不一定要使用交谈模式来输入，可以先将所有的指定先写入档案后再一次读入

-l：列出所有的指定（使用者也可以使用 atq 而不用 at -l）

-d：删除指定（使用者也可以使用 atrm 而不用 at -d）

-v：列出所有已经完成但尚未删除的指定

例子：

三天后的下午 5 点钟执行 /bin/ls：

```
at 5pm + 3 days /bin/ls
```

三个星期后的下午 5 点钟执行 /bin/ls：

```
at 5pm + 2 weeks /bin/ls
```

明天的 17:20 执行 /bin/date：

```
at 17:20 tomorrow /bin/date
```

1999 年的最后一天的最后一分钟印出 the end of world！

```
at 23:59 12/31/1999 echo the end of world！
```

名称 alias

用途

定义或显示别名。

语法

```
alias [ -t ] [ -x ] [ AliasName [ =String ] ] ...
```

描述

alias 命令创建或重定义别名定义或将现有别名定义写至标准输出。

如果未提供标志或参数，则将所有现有别名定义写至标准输出。可以通过使用 **AliasName** 参数来显示特定的别名定义。

通过使用 **AliasName=String** 参数对来创建新的别名。当 **shell** 在命令行上或 **shell** 脚本中碰到别名时，它替换此字符串提供的定义。**String** 变量可以包含任何有效的 **shell** 文本。如果此字符串包含空格，请将 **String** 变量的值括在单引号内。如果 **AliasName** 参数不是一个有效的名称，则 **alias** 命令显示一条错误消息。

如果指定 **-t** 标志，则 **shell** 显示被跟踪的别名。一个被跟踪的命令使用该命令的完整路径名称。当重新设置 **PATH** 环境变量的值时，被跟踪的命令可能变成未定义，但用 **-t** 标志创建的别名仍旧被跟踪。

如果指定 **-x** 标志，**shell** 显示导出的别名。导出的别名在所有 **shell** 中都是活动的。

别名定义影响当前 **shell** 环境和任何子 **shell** 的执行环境。别名定义不影响当前 **shell** 的父进程或 **shell** 调用的任何实用程序环境。

标志

- t** 设置或显示所有现有被跟踪的别名。如果此标志和 **AliasName** 参数一起使用，则跟踪新的别名且别名定义包含通过进行路径搜索而获得的完整路径名称。当重新设置 **PATH** 环境变量的值时，别名定义变成未定义但仍旧被跟踪。
- x** 显示所有现有的导出的别名定义。如果此标志和 **AliasName** 参数一起使用，则导出新的别名。导出的别名不在 **shell** 的单独的调用间定义。必须将别名定义置于您的环境文件中以使别名对于单独的 **shell** 调用都是已经定义的。

退出状态

返回以下出口值：

- 0** 成功完成。
- >0** 指定的别名名称之一不具有别名定义或发生错误。

示例

要更改 **ls** 命令使其在列中显示信息并对输出进行注释，请输入：

```
alias aa='ls -l'
```

要创建用来重复命令历史文件中的先前条目的命令，请输入：

```
alias r='fc -s'
```

要为 **du** 命令使用 **1KB** 单元，请输入：

```
alias du=du\ -k
```

要为用户 **Dee** 创建一个命令来显示所有活动的进程，请输入：

```
alias psc='ps -ef | grep Dee'
```

要查看 **ls** 命令的完整路径名称，请输入：

```
alias -t ls
```

屏幕显示 **ls=/usr/bin/ls**。

2) B 类指令

3) C 类指令

名称: cat

使用权限: 所有使用者

使用方式: `cat [-AbeEnstTuv] [--help] [--version] fileName`

说明: 把档案串连接后传到基本输出 (萤幕或加 `> fileName` 到另一个档案)

参数:

`-n` 或 `--number` 由 1 开始对所有输出的行数编号

`-b` 或 `--number-nonblank` 和 `-n` 相似, 只不过对于空白行不编号

`-s` 或 `--squeeze-blank` 当遇到有连续两行以上的空白行, 就代换为一行的空白行

`-v` 或 `--show-nonprinting`

范例:

`cat -n textfile1 > textfile2` 把 `textfile1` 的档案内容加上行号后输入 `textfile2` 这个档案里

`cat -b textfile1 textfile2 >> textfile3` 把 `textfile1` 和 `textfile2` 的档案内容加上行号 (空白行不加) 之后将内容附加到 `textfile3`

名称 cd

使用权限: 所有使用者

使用方式: `cd [dirName]`

说明: 变换工作目录至 `dirName`。其中 `dirName` 表示法可为绝对路径或相对路径。若目录名称省略, 则变换至使用者的 `home directory` (也就是刚 `login` 时所在的目录)。

另外, `"~"` 也表示为 `home directory` 的意思, `"."` 则是表示目前所在的目录, `".."` 则表示目前目录位置的上一层目录。

范例: 跳到 `/usr/bin/`:

`cd /usr/bin`

跳到自己的 `home directory`:

`cd ~`

跳到目前目录的上上两层：

```
cd ../../
```

指令 chmod

使用权限：所有使用者

使用方式：chmod [-cfvR] [--help] [--version] mode file...

说明：Linux/Unix 的档案存取权限分为三级：档案拥有者、群组、其他。利用 **chmod** 可以藉以控制档案如何被他人所存取。

把计：

mode：权限设定字串，格式如下：**[ugoa...][[+ -=][rwxX]...][...]**，其中 **u** 表示该档案的拥有者，**g** 表示与该档案的拥有者属于同一个群体(**group**)者，**o** 表示其他以外的人，**a** 表示这三者皆是。

+ 表示增加权限、**-** 表示取消权限、**=** 表示唯一设定权限。

r 表示可读取，**w** 表示可写入，**x** 表示可执行，**X** 表示只有当该档案是子目录或者该档案已经被设定过为可执行。

-c：若该档案权限确实已经更改，才显示其更改动作

-f：若该档案权限无法被更改也不要显示错误讯息

-v：显示权限变更的详细资料

-R：对目前目录下的所有档案与子目录进行相同的权限变更(即以递归的方式逐个变更)

--help：显示辅助说明

--version：显示版本

范例：将档案 **file1.txt** 设为所有人皆可读取：

```
chmod ugo+r file1.txt
```

将档案 **file1.txt** 设为所有人皆可读取：

```
chmod a+r file1.txt
```

将档案 **file1.txt** 与 **file2.txt** 设为该档案拥有者，与其所属同一个群体者可写入，但其他以外的人则不可写入：

```
chmod ug+w,o-w file1.txt file2.txt
```

将 **ex1.py** 设定为只有该档案拥有者可以执行：

```
chmod u+x ex1.py
```

将目前目录下的所有档案与子目录皆设为任何人可读取：

```
chmod -R a+r *
```

此外 **chmod** 也可以用数字来表示权限如 **chmod 777 file**
语法为：**chmod abc file**

其中 **a,b,c** 各为一个数字，分别表示 **User**、**Group**、及 **Other** 的权限。

r=4, w=2, x=1

若要 **rwX** 属性则 $4+2+1=7$;

若要 **rw-**属性则 $4+2=6$;

若要 **r-x** 属性则 $4+1=5$ 。

范例：

chmod a=rwx file

和

chmod 777 file

效果相同

chmod ug=rwx,o=x file

和

chmod 771 file

效果相同

若用 **chmod 4755 filename** 可使此程式具有 **root** 的权限

指令 **chown**

使用权限：**root**

使用方式：**chown [-cfhvR] [--help] [--version] user[:group] file...**

说明：**Linux/Unix** 是多人多工作业系统，所有的档案皆有拥有者。利用 **chown** 可以将档案的拥有者加以改变。一般来说，这个指令只有是由系统管理者(**root**)所使用，一般使用者没有权限可以改变别人的档案拥有者，也没有权限可以自己的档案拥有者改设为别人。只有系统管理者(**root**)才有这样的权限。

把计：

user：新的档案拥有者的使用者 **IDgroup**：新的档案拥有者的使用者群体(**group**)**-c**：若该档案拥有者确实已经更改，才显示其更改动作**-f**：若该档案拥有者无法被更改也不要显示错误讯息**-h**：只对于连结(**link**)进行变更，而非该 **link** 真正指向的档案**-v**：显示拥有者变更的详细资料**-R**：对目前目录下的所有档案与子目录进行相同的拥有者变更(即以递归的方式逐个变更)**--help**：显示辅助说明**--version**：显示版本

范例：

将档案 file1.txt 的拥有者设为 users 群体的使用者 jessie：

```
chown jessie:users file1.txt
```

将目前目录下的所有档案与子目录的拥有者皆设为 users 群体的使用者 lampport：

```
chmod -R lampport:users *
```

名称：cp

使用权限：所有使用者

使用方式：

```
cp [options] source dest
```

```
cp [options] source... directory
```

说明：将一个档案拷贝至另一档案，或将数个档案拷贝至另一目录。

把计

-a 尽可能将档案状态、权限等资料都照原状予以复制。

-r 若 source 中含有目录名，则将目录下之档案亦皆依序拷贝至目的地。

-f 若目的地已经有相同档名的档案存在，则在复制前先予以删除再行复制。

范例：

将档案 aaa 复制(已存在)，并命名为 bbb：

```
cp aaa bbb
```

将所有的 C 语言程式拷贝至 Finished 子目录中：

```
cp *.c Finished
```

名称：cut

使用权限：所有使用者

用法：cut -cnum1-num2 filename

说明：显示每行从开头算起 num1 到 num2 的文字。

范例：

```
shell>> cat example
test2
this is test1
shell>> cut -c0-6 example ## print 开头算起前 6 个字元
test2
this i
```

名称：cal

使用权限：所有使用者

使用方式：cal [-m] [month [year]]

说明：

显示日历。若只有一个参数，则代表年份(1-9999)，显示该年的年历。年份必须全部写出：`cal 89\` 将不会是显示 1989 年的年历。使用两个参数，则表示月份及年份。若没有参数则显示这个月的月历。

1752 年 9 月第 3 日起改用西洋新历，因这时大部份的国家都采用新历，有 10 天被去除，所以该月份的月历有些不同。在此之前为西洋旧历。

匡兜

-m：以星期一至每周的第一天方式显示。
-j：以凯撒历显示，即以一月一日起的天数显示。
-y：显示今年年历。

范例：

cal：显示本月的月历。

```
[root@mylinux /root]# date
Tue Aug 15 08:00:18 CST 2000
[root@mylinux /root]# cal
August 2000
Su Mo Tu We Th Fr Sa
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
```



```
[root@mylinux /root]#
```

cal 2001 : 显示公元 2001 年年历。

```
[root@mylinux /root]# cal 2001
```

```
2001
```

January February March

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 1 2 3 1 2 3

7 8 9 10 11 12 13 4 5 6 7 8 9 10 4 5 6 7 8 9 10

14 15 16 17 18 19 20 11 12 13 14 15 16 17 11 12 13 14 15 16 17

21 22 23 24 25 26 27 18 19 20 21 22 23 24 18 19 20 21 22 23 24

28 29 30 31 25 26 27 28 25 26 27 28 29 30 31

April May June

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7 1 2 3 4 5 1 2

8 9 10 11 12 13 14 6 7 8 9 10 11 12 3 4 5 6 7 8 9

15 16 17 18 19 20 21 13 14 15 16 17 18 19 10 11 12 13 14 15 16

22 23 24 25 26 27 28 20 21 22 23 24 25 26 17 18 19 20 21 22 23

29 30 27 28 29 30 31 24 25 26 27 28 29 30

July August September

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7 1 2 3 4 1

8 9 10 11 12 13 14 5 6 7 8 9 10 11 2 3 4 5 6 7 8

15 16 17 18 19 20 21 12 13 14 15 16 17 18 9 10 11 12 13 14 15

22 23 24 25 26 27 28 19 20 21 22 23 24 25 16 17 18 19 20 21 22

29 30 31 26 27 28 29 30 31 23 24 25 26 27 28 29

30

October November December

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 1 2 3 1

7 8 9 10 11 12 13 4 5 6 7 8 9 10 2 3 4 5 6 7 8

14 15 16 17 18 19 20 11 12 13 14 15 16 17 9 10 11 12 13 14 15

21 22 23 24 25 26 27 18 19 20 21 22 23 24 16 17 18 19 20 21 22

28 29 30 31 25 26 27 28 29 30 23 24 25 26 27 28 29

30 31

```
[root@mylinux /root]#
```

cal 5 2001 : 显示公元 2001 年 5 月月历。

```
[root@mylinux /root]# cal 5 2001
```

May 2001

Su Mo Tu We Th Fr Sa

1 2 3 4 5

6 7 8 9 10 11 12

13 14 15 16 17 18 19

20 21 22 23 24 25 26

27 28 29 30 31

```
[root@mylinux /root]#
```

cal -m : 以星期一为每周的第一天方式，显示本月的月历。

```
[root@mylinux /root]# cal -m
```

August 2000

Mo Tu We Th Fr Sa Su

1 2 3 4 5 6

7 8 9 10 11 12 13

14 15 16 17 18 19 20

21 22 23 24 25 26 27

28 29 30 31

```
[root@mylinux /root]#
```

cal -jy : 以一月一日起的天数显示今年的年历。

```
[root@mylinux /root]# cal -jy
2000
```

January February

Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	32	33	34	35	36								
2	3	4	5	6	7	8	37	38	39	40	41	42	43
9	10	11	12	13	14	15	44	45	46	47	48	49	50
16	17	18	19	20	21	22	51	52	53	54	55	56	57
23	24	25	26	27	28	29	58	59	60				
30	31												

March April

Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat
61	62	63	64	92									
65	66	67	68	69	70	71	93	94	95	96	97	98	99
72	73	74	75	76	77	78	100	101	102	103	104	105	106
79	80	81	82	83	84	85	107	108	109	110	111	112	113
86	87	88	89	90	91	114	115	116	117	118	119	120	
121													

May June

Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat
122	123	124	125	126	127	153	154	155					
128	129	130	131	132	133	134	156	157	158	159	160	161	162
135	136	137	138	139	140	141	163	164	165	166	167	168	169
142	143	144	145	146	147	148	170	171	172	173	174	175	176
149	150	151	152	177	178	179	180	181	182				

July August

Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat
183	214	215	216	217	218								
184	185	186	187	188	189	190	219	220	221	222	223	224	225
191	192	193	194	195	196	197	226	227	228	229	230	231	232
198	199	200	201	202	203	204	233	234	235	236	237	238	239
205	206	207	208	209	210	211	240	241	242	243	244		
212	213												

September October

Sun	Mon	Tue	Wed	Thu	Fri	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Sat
245	246	275	276	277	278	279	280	281					
247	248	249	250	251	252	253	282	283	284	285	286	287	288
254	255	256	257	258	259	260	289	290	291	292	293	294	295
261	262	263	264	265	266	267	296	297	298	299	300	301	302
268	269	270	271	272	273	274	303	304	305				

November December
Sun Mon Tue Wed Thu Fri Sat Sun Mon Tue Wed Thu Fri Sat
306 307 308 309 336 337
310 311 312 313 314 315 316 338 339 340 341 342 343 344
317 318 319 320 321 322 323 345 346 347 348 349 350 351
324 325 326 327 328 329 330 352 353 354 355 356 357 358
331 332 333 334 335 359 360 361 362 363 364 365
366

[root@mylinux /root]#

名称 crontab

使用权限：所有使用者

使用方式：

`crontab [-u user] filecrontab [-u user] { -l | -r | -e }`

说明：

crontab 是用来让使用者在固定时间或固定间隔执行程式之用，换句话说，也就是类似使用者的时程表。**-u user** 是指设定指定 **user** 的时程表，这个前提是你必须要有其权限(比如说是 **root**)才能够指定他人的时程表。如果不使用 **-u user** 的话，就是表示设定自己的时程表。

参数：

-e：执行文字编辑器来设定时程表，内定的文字编辑器是 **VI**，如果你想用别的文字编辑器，则请先设定 **VISUAL** 环境变量来指定使用那个文字编辑器(比如说 `setenv VISUAL joe`)

-r：删除目前的时程表

-l：列出目前的时程表

时程表的格式如下：

f1 f2 f3 f4 f5 program

其中 **f1** 是表示分钟，**f2** 表示小时，**f3** 表示一个月份中的第几日，**f4** 表示月份，**f5** 表示一个星期中的第几天。**program** 表示要执行的程式。

当 **f1** 为 ***** 时表示每分钟都要执行 **program**，**f2** 为 ***** 时表示每小时都要执行程式，其余类推

当 **f1** 为 **a-b** 时表示从第 **a** 分钟到第 **b** 分钟这段时间内要执行，**f2** 为 **a-b** 时表示从第 **a** 到第 **b** 小时都要执行，其余类推

当 **f1** 为 ***/n** 时表示每 **n** 分钟个时间间隔执行一次，**f2** 为 ***/n** 表示每 **n** 小时个时间间隔执行一次，其余类推

当 **f1** 为 **a, b, c,...** 时表示第 **a, b, c,...** 分钟要执行，**f2** 为 **a, b, c,...** 时表示第 **a, b, c,...** 个小时要执行，其余

类推

使用者也可以将所有的设定先存放在档案 `file` 中，用 `crontab file` 的方式来设定时程表。
例子：

每月每天每小时的第 0 分钟执行一次 `/bin/ls`：

```
0 7 * * * /bin/ls
```

在 12 月内，每天的早上 6 点到 12 点中，每隔 20 分钟执行一次 `/usr/bin/backup`：

```
0 6-12/3 * 12 * /usr/bin/backup
```

周一到周五每天下午 5:00 寄一封信给 alex@domain.name：

```
0 17 * * 1-5 mail -s "hi" alex@domain.name < /tmp/maildata
```

每月每天的午夜 0 点 20 分, 2 点 20 分, 4 点 20 分....执行 `echo "haha"`

```
20 0-23/2 * * * echo "haha"
```

注意：

当程式在你所指定的时间执行后，系统会寄一封信给你，显示该程式执行的内容，若是你不希望收到这样的信，请在每一行空一格之后加上 `> /dev/null 2>&1` 即可。

名称：chfn

使用权限：所有使用者

用法：shell>> chfn

说明：提供使用者更改个人资讯，用于 `finger` and `mail username`

范例：

```
shell>> chfn
```

```
Changing finger information for user
```

```
Password: [del]
```

```
Name[:Johnney Huang ### 提供 finger 时的资料
```

```
Office[:NCCU
```

```
Office Phone[: [del]
```

```
Home Phone[: [del]
```

名称：chgrp

功能说明：变更文件或目录的所属群组。

语 法: chgrp [-cfhRv][--help][--version][所属群组][文件或目录...] 或 chgrp [-cfhRv][--help][--reference=<参考文件或目录>][--version][文件或目录...]

补充说明: 在 UNIX 系统家族里, 文件或目录权限的掌控以拥有者及所属群组来管理。您可以使用 **chgrp** 指令去变更文件与目录的所属群组, 设置方式采用群组名称或群组识别码皆可。

参 数:

- c 或--changes 效果类似"-v"参数, 但仅回报更改的部分。
- f 或--quiet 或--silent 不显示错误信息。
- h 或--no-dereference 只对符号连接的文件作修改, 而不更动其他任何相关文件。
- R 或--recursive 递归处理, 将指定目录下的所有文件及子目录一并处理。
- v 或--verbose 显示指令执行过程。
- help 在线帮助。
- reference=<参考文件或目录> 把指定文件或目录的所属群组全部设成和参考文件或目录的所属群组相同。
- version 显示版本信息

名称: chsh

使用权限: 所有使用者

用法: shell>> chsh

说明: 更改使用者 shell 设定

范例:

```
shell>> chsh
Changing fihanging shell for user1
Password: [del]
New shell [/bin/tcsh]: ### [是目前使用的 shell]
[del]
```

```
shell>> chsh -l ### 展示 /etc/shells 档案内容
/bin/bash
/bin/sh
/bin/ash
/bin/bsh
/bin/tcsh
/bin/csh
```

指令: clear

用途：清除萤幕用。

使用方法：在 `console` 上输入 `clear`。

名称: `reset`, `tset`

使用方法: `tset [-lQqrs] [-] [-e ch] [-i ch] [-k ch] [-m mapping] [terminal]`

使用说明:

`reset` 其实和 `tset` 是一同个命令，它的用途是设定终端机的状态。一般而言，这个命令会自动的从环境变数、命令列或是其它的组态档决定目前终端机的型态。如果指定型态是 `?` 的话，这个程式会要求使用者输入终端机的型别。

由于这个程式会将终端机设回原始的状态，除了在 `login` 时使用外，当系统终端机因为程式不正常执行而进入一些奇怪的状态时，你也可以用它来重设终端机。例如不小心把二进位档用 `cat` 指令进到终端机，常会有终端机不再回应键盘输入，或是回应一些奇怪字元的问题。此时就可以用 `reset` 将终端机回复至原始状态。选项说明：

`-p`

将终端机类别显示在萤幕上，但不做设定的动作。这个命令可以用来取得目前终端机的类别。

`-e ch`

将 `erase` 字元设成 `ch`

`-i ch`

将中断字元设成 `ch`

`-k ch`

将删除一行的字元设成 `ch`

`-l`

不要做设定的动作，如果没有使用选项 `-Q` 的话，`erase`、中断及删除字元的目前值依然会送到萤幕上。

`-Q`

不要显示 `erase`、中断及删除字元的值到萤幕上。

`-r`

将终端机类别印在萤幕上。

`-s`

将设定 `TERM` 用的命令用字串的型式送到终端机中，通常在 `.login` 或 `.profile` 中用

范例：

让使用者输入一个终端机型别并将终端机设到该型别的预设状态。

```
# reset ?
```

将 `erase` 字元设定 `control-h`

```
# reset -e ^B
```

将设定用的字串显示在萤幕上

```
# reset -s
```

Erase is control-B (^B).

Kill is control-U (^U).

Interrupt is control-C (^C).

```
TERM=xterm;
```

名称: **compress**

使用权限: 所有使用者

使用方式: **compress** [-dfvcV] [-b maxbits] [file ...]

说明:

compress 是一个相当古老的 **unix** 档案压缩指令，压缩后的档案会加上一个 **.Z** 延伸档名以区别未压缩的档案，压缩后的档案可以以 **uncompress** 解压。若要将数个档案压成一个压缩档，必须先将档案 **tar** 起来再压缩。由于 **gzip** 可以产生更理想的压缩比例，一般人多已改用 **gzip** 为档案压缩工具。

参数:

c 输出结果至标准输出设备（一般指荧幕）

f 强迫写入档案，若目的档已经存在，则会被覆盖 (**force**)

v 将程式执行的讯息印在荧幕上 (**verbose**)

b 设定共同字串数的上限，以位元计算，可以设定的值为 **9** 至 **16 bits** 。由于值越大，能使用的共同字串就越多，压缩比例就越大，所以一般使用预设值 **16 bits (bits)**

d 将压缩档解压缩

V 列出版本讯息

范例:

将 **source.dat** 压缩成 **source.dat.Z** ，若 **source.dat.Z** 已经存在，内容则会被压缩档覆盖。

```
compress -f source.dat
```


将 `source.dat` 压缩成 `source.dat.Z`，并列印出压缩比例。

`-v` 与 `-f` 可以一起使用

```
compress -vf source.dat
```

将压缩后的资料输出后再导入 `target.dat.Z` 可以改变压缩档名。

```
compress -c source.dat > target.dat.Z
```

`-b` 的值越大，压缩比例就越大，范围是 `9-16`，预设值是 `16`。

```
compress -b 12 source.dat
```

将 `source.dat.Z` 解压成 `source.dat`，若档案已经存在，使用者按 `y` 以确定覆盖档案，若使用 `-df` 程式则会自动覆盖档案。由于系统会自动加入 `.Z` 为延伸档名，所以 `source.dat` 会自动当作 `source.dat.Z` 处理。

```
compress -d source.dat
```

```
compress -d source.dat.Z
```

4) D 类指令

名称 `date`

使用权限：所有使用者

使用方式：

```
date [-u] [-d datestr] [-s datestr] [--utc] [--universal] [--date=datestr] [--set=datestr] [--help] [--version] [+FORMAT]
[MMDDhhmm[[CC]YY][.ss]]
```

说明：

date 可以用来显示或设定系统的日期与时间，在显示方面，使用者可以设定欲显示的格式，格式设定为一个加号后接数个标记，其中可用的标记列表如下：

时间方面：

%：印出 %
%n：下一行
%t：跳格
%H：小时(00..23)
%I：小时(01..12)
%k：小时(0..23)
%l：小时(1..12)
%M：分钟(00..59)
%p：显示本地 AM 或 PM
%r：直接显示时间 (12 小时制，格式为 hh:mm:ss [AP]M)
%s：从 1970 年 1 月 1 日 00:00:00 UTC 到目前为止的秒数
%S：秒(00..61)
%T：直接显示时间 (24 小时制)
%X：相当于 %H:%M:%S
%Z：显示时区

日期方面：

%a：星期几 (Sun..Sat)
%A：星期几 (Sunday..Saturday)
%b：月份 (Jan..Dec)
%B：月份 (January..December)
%c：直接显示日期与时间
%d：日 (01..31)
%D：直接显示日期 (mm/dd/yy)
%h：同 %b
%j：一年中的第几天 (001..366)
%m：月份 (01..12)
%U：一年中的第几周 (00..53) (以 Sunday 为一周的第一天情形)
%w：一周中的第几天 (0..6)
%W：一年中的第几周 (00..53) (以 Monday 为一周的第一天情形)
%x：直接显示日期 (mm/dd/yy)
%y：年份的最后两位数字 (00..99)

%Y: 完整年份 (0000..9999)

若是不以加号作为开头, 则表示要设定时间, 而时间格式为 MMDDhhmm[[CC]YY][.ss], 其中 MM 为月份, DD 为日, hh 为小时, mm 为分钟, CC 为年份前两位数字, YY 为年份后两位数字, ss 为秒数

把计 :

-d datestr : 显示 datestr 中所设定的时间 (非系统时间)

--help : 显示辅助讯息

-s datestr : 将系统时间设为 datestr 中所设定的时间

-u : 显示目前的格林威治时间

--version : 显示版本编号

例子 :

显示时间后跳行, 再显示目前日期 :

```
date +%T%n%D
```

显示月份与日数 :

```
date +%B %d
```

显示日期与设定时间(12:34:56) :

```
date --date 12:34:56
```

注意 :

当你不希望出现无意义的 0 时(比如说 1999/03/07), 则可以在标记中插入 - 符号, 比如说 date +%H:%M:%S 会把时分秒中无意义的 0 给去掉, 像是原本的 08:09:04 会变为 8:9:4。另外, 只有取得权限者(比如说 root)才能设定系统时间。

当你以 root 身分更改了系统时间之后, 请记得以 clock -w 来将系统时间写入 CMOS 中, 这样下次重新开机时系统时间才会持续抱持最新的正确值。

名称: dd

dd 命令的用法

dd 命令是将从标准输入中读取的数据按参数指定的方法进行变换后送到标准输出。除了数据变换外, 还可以利用一些参数修改一些数据的物理属性。如块的大小等。

dd 命令的选项及含义:

ds=blocksize 指定输入输出块的大小, 该选项使 ibs 和 obs 选项无效。

if=filename 指定将要拷贝输入的文件名

ibs=blocksize 指定输入文件的块大小。

ifskip=numberofeof 指定前跳过的 eof 标志的个数

files=numberofblocks 在拷贝前, 在输入文件上跳过指定数目的块。

of=filename 指定创建的输出文件名

obs=blocaksize 指定输出块大小

seek=recordnumber 拷贝文件时，指定输入文件的开始记录号。

conv=conversionparameter 指定数据转换的类型，描述数据转换类型的参数有 ASCII,EBCDIC,block,unblock.lcase,,ucase.

示例：将 file.dd 输出到磁盘文件。

```
dd if=file.dd of=/dev/rfd0135ds18
```

将 EBCDIC 格式存放的文件转换为 ASCII 文件

```
dd if=file1 of=file2 conv=ascii
```

将磁带上的三个文件拷贝到文件 file1

```
dd if=/dev/mnt0 fskip=2 of=file1
```

名称：dmesg

功能说明：显示开机信息。

语 法：dmesg [-cn][-s <缓冲区大小>]

补充说明：kernel 会将开机信息存储在 ring buffer 中。您若是开机时来不及查看信息，可利用 dmesg 来查看。开机信息亦保存在 /var/log 目录中，名称为 dmesg 的文件里。

参 数：

-c 显示信息后，清除 ring buffer 中的内容。

-s<缓冲区大小> 预设置为 8196，刚好等于 ring buffer 的大小。

-n 设置记录信息的层级。

5) E 类指令

名称：expr

使用权限：所有使用者

字符串长度

```
shell>> expr length "this is a test"
```

```
14
```

数字商数

```
shell>> expr 14 % 9
```

```
5
```

从位置处抓取字符串

```
shell>> expr substr "this is a test" 3 5
```

is is

数字串 only the first character

```
shell>> expr index "testforthe game" e
2
```

字符串真实重现

```
shell>> expr quote thisisatestformela
thisisatestformela
```

6) F 类指令

名称 find

用法 : find

使用说明 :

将档案系统内符合 **expression** 的档案列出来。你可以指要档案的名称、类别、时间、大小、权限等不同资讯的组合，只有完全相符的才会被列出来。

find 根据下列规则判断 **path** 和 **expression**，在命令列上第一个 **- () , !** 之前的部份为 **path**，之后的是 **expression**。如果 **path** 是空字符串则使用目前路径，如果 **expression** 是空字符串则使用 **-print** 为预设 **expression**

expression 中可使用的选项有二三十个之多，在此只介绍最常用的部份。

-mount, -xdev : 只检查和指定目录在同一个档案系统下的档案，避免列出其它档案系统中的档案

-amin n : 在过去 **n** 分钟内被读取过

-anewer file : 比档案 **file** 更晚被读取过的档案

-atime n : 在过去 **n** 天过读取过的档案

-cmin n : 在过去 **n** 分钟内被修改过

-cnewer file : 比档案 **file** 更新的档案

-ctime n : 在过去 **n** 天过修改过的档案

-empty : 空的档案 **-gid n or -group name** : **gid** 是 **n** 或是 **group** 名称是 **name**

-ipath p, -path p : 路径名称符合 **p** 的档案，**ipath** 会忽略大小写

-name name, -iname name : 档案名称符合 **name** 的档案。**iname** 会忽略大小写

-size n : 档案大小 是 **n** 单位，**b** 代表 512 位元组的区块，**c** 表示字数，**k** 表示 kilo bytes，**w** 是二个位元组。**-type c** : 档案类型是 **c** 的档案。

d : 目录

c : 字型装置档案

b: 区块装置档案
p: 具名贮列
f: 一般档案
l: 符号连结
s: socket
-pid n : process id 是 n 的档案

你可以使用 () 将运算式分隔，并使用下列运算。

exp1 -and exp2

! expr

-not expr

exp1 -or exp2

exp1, exp2

范例:

将目前目录及其子目录下所有延伸档名是 c 的档案列出来。

```
# find . -name "*.c"
```

将目前目录其其下子目录中所有一般档案列出

```
# find . -ftype f
```

将目前目录及其子目录下所有最近 20 分钟内更新过的档案列出

```
# find . -ctime -20
```

名称: **finger**

使用权限: 所有使用者

使用方式: **finger** [options] user[@address]

说明: **finger** 可以让使用者查询一些其他使用者的资料。会列出来的资料有:

Login Name

User Name

Home directory

Shell

Login status

mail status

.plan

.project

.forward

其中 .plan , .project 和 .forward 就是使用者在他的 Home Directory 里的 .plan , .project 和 .forward 等

档案里的资料。如果没有就没有。**finger** 指令并不限于在同一伺服器上查询，也可以寻找某一个远端伺服器上的使用者。只要给一个像是 **E-mail address** 一般的地址即可。

把计

-l

多行显示。

-s

单行显示。这个选项只显示登入名称，真实姓名，终端机名称，闲置时间，登入时间，办公室号码及电话号码。如果所查询的使用者是远端伺服器的使用者，这个选项无效。

范例：下列指令可以查询本机管理员的资料：

finger root

其结果如下：

Login: root Name: root

Directory: /root Shell: /bin/bash

Never logged in.

No mail.

No Plan.

名称： **fdformat**

使用权限： 所有使用者

使用方式： **fdformat [-n] device**

使用说明：

对指定的软碟机装置进行低阶格式化。使用这个指令对软碟格式化的时候，最好指定像是下面的装置：

/dev/fd0d360 磁碟机 A: ，磁片为 360KB 磁碟

/dev/fd0h1440 磁碟机 A: ，磁片为 1.4MB 磁碟

/dev/fd1h1200 磁碟机 B: ，磁片为 1.2MB 磁碟

如果使用像是 **/dev/fd0** 之类的装置，如果里面的磁碟不是标准容量，格式化可能会失败。在这种情况下，使用者可以用 **setfdprm** 指令先行指定必要参数。

参数：

-n 关闭确认功能。这个选项会关闭格式化之后的确认步骤。

范例：

```
fdformat -n /dev/fd0h1440
```

将磁碟机 A 的磁片格式化成 1.4MB 的磁片。并且省略确认的步骤。

7) G 类指令

名称：grep

使用权限：所有使用者

使用方法：

◎grep -- print lines matching a pattern (将符合样式的该行列出)

◎语法: grep [options]

PATTERN [FILE...]

grep 用以在 file 内文中比对相对应的部分，或是当没有指定档案时，由标准输入中去比对。在预设的情况下，grep 会将符合样式的那一行列出。

此外，还有两个程式是 grep 的变化型，egrep 及 fgrep。其中 egrep 就等同於 grep -E，fgrep 等同於 grep -F。

◎参数

1.-A NUM, --after-context=NUM

除了列出符合行之外，并且列出後 NUM 行。

ex:\$ grep -A 1 panda file

(从 file 中搜寻有 panda 样式的行，并显示该行的後 1 行)

2. -a 或--text

grep 原本是搜寻文字档，若拿二进位的档案作为搜寻的目标，则会显示如下的讯息: Binary file 二进位档名 matches 然後结束。

若加上-a 参数则可将二进位档案视为文字档案搜寻，相当於--binary-files=text 这个参数。

ex: (从二进位档案 mv 中去搜寻 panda 样式) (错误!!!)

\$ grep panda mv

Binary file mv matches

(这表示此档案有 match 之处，详见--binary-files=TYPE) \$

(正确!!!)

\$ grep -a panda mv

3. -B NUM, --before-context=NUM

与 -A NUM 相对，但这此参数是显示除符合行之外并显示在它之前的 NUM 行。

ex: (从 file 中搜寻有 panda 样式的行, 并显示该行的前 1 行)

```
$ grep -B 1 panda file
```

4. -C [NUM], -NUM, --context[=NUM]

列出符合行之外并列出上下各 NUM 行, 预设值是 2。

ex: (列出 file 中除包含 panda 样式的行外并列出其上下 2 行)

(若要改变预设值, 直接改变 NUM 即可)

```
$ grep -C[NUM] panda file
```

5. -b, --byte-offset

列出样式之前的内文总共有多少 byte ..

ex: \$ grep -b panda file

显示结果类似於: 0:panda

66:pandahuang

123:panda03

6. --binary-files=TYPE

此参数 TYPE 预设为 binary(二进位), 若以普通方式搜寻, 只有 2 种结果:

1.若有符合的地方: 显示 Binary file 二进位档名 matches

2.若没有符合的地方: 什麼都没有显示。

若 TYPE 为 without-match, 遇到此参数,

grep 会认为此二进位档案没有包含任何搜寻样式, 与-I 参数相同。

若 TPYE 为 text, grep 会将此二进位档视为 text 档案, 与-a 参数相同。

Warning: --binary-files=text 若输出为终端机, 可能会产生一些不必要的输出。

7. -c, --count

不显示符合样式行, 只显示符合的总行数。

若再加上-v,--invert-match, 参数显示不符合的总行数。

8. -d ACTION, --directories=ACTION

若输入的档案是一个资料夹, 使用 ACTION 去处理这个资料夹。

预设 ACTION 是 read(读取), 也就是说此资料夹会被视为一般的档案;

若 ACTION 是 skip(略过), 资料夹会被 grep 略过:

若 ACTION 是 recurse(递), grep 会去读取资料夹下所有的档案,

此相当於-r 参数。

9. -E, --extended-regexp

采用规则表示式去解释样式。

10. -e PATTERN, --regexp=PATTERN

把样式做为一个 partern, 通常用在避免 partern 用-开始。

11. -f FILE, --file=FILE

事先将要搜寻的样式写入到一个档案, 一行一个样式。

然後采用档案搜寻。

空的档案表示没有要搜寻的样式, 因此也就不会有任何符合。

ex: (newfile 为搜寻样式档)

```
$grep -f newfile file
```

12. -G, --basic-regexp

将样式视为基本的规则表示式解释。(此为预设)

13. -H, --with-filename

在每个符合样式行前加上符合的档案名称，若有路径会显示路径。

ex: (在 file 与 testfile 中搜寻 panda 样式)

```
$grep -H panda file ./testfile
```

```
file:panda
```

```
./testfile:panda
```

```
$
```

14. -h, --no-filename

与-H 参数相类似，但在输出时不显示路径。

15. --help

产生简短的 help 讯息。

16. -I

grep 会强制认为此二进位档案没有包含任何搜寻样式，与--binary-files=without-match 参数相同。

ex: \$ grep -I panda mv

17. -i, --ignore-case

忽略大小写，包含要搜寻的样式及被搜寻的档案。

ex: \$ grep -i panda mv

18. -L, --files-without-match

不显示平常一般的输出结果，反而显示出没有符合的档案名称。

19. -l, --files-with-matches

不显示平常一般的输出结果，只显示符合的档案名称。

20. -mmap

如果可能，使用 mmap 系统呼叫去读取输入，而不是预设的 read 系统呼叫。

在某些状况，--mmap 能产生较好的效能。然而，--mmap

如果运作中档案缩短，或 I/O 错误发生时，

可能造成未定义的行为(包含 core dump)，。

21. -n, --line-number

在显示行前，标上行号。

ex: \$ grep -n panda file

显示结果相似於下行号:符合行的内容

22. -q, --quiet, --silent

不显示任何的一般输出。请参阅-s 或--no-messages

23. -r, --recursive

递地，读取每个资料夹下的所有档案，此相当於 -d recsuse 参数。

24. -s, --no-messages

不显示关于不存在或无法读取的错误讯息。

小: 不像 GNU grep，传统的 grep 不符合 POSIX.2 协定，因为缺乏-q 参数，且他的-s 参数表现像 GNU grep 的 -q 参数。

Shell Script 倾向将传统的 grep 移植，避开-q 及-s 参数，且将输出限制到/dev/null。

POSIX: 定义 UNIX 及 UNIX-like 系统需要提供的功能

25. -V, --version

显示出 `grep` 的版本号到标准错误。

当您在回报有关 `grep` 的 bugs 时，`grep` 版本号是必须要包含在内的。

26. `-v, --invert-match`

显示除搜寻样式行之外的全部。

27. `-w, --word-regexp`

将搜寻样式视为一个字去搜寻，完全符合该"字"的行才会被列出。

28. `-x, --line-regexp`

将搜寻样式视为一行去搜寻，完全符合该"行"的行才会被列出。

名称: `groupadd`

`groupadd` - 建立新群组

语法

`groupadd [-g gid [-o]] [-r] [-f] group`

描述

`groupadd` 可指定群组名称来建立新的群组帐号。需要时可从系统中取得新群组值。`groupadd` 有下列选项可用。

`-g gid group's`

ID 值。除非使用 `-o` 参数不然该值必须是唯一，不可相同。数值不可为负。预设值为最小不得小于 500 而逐次增加。0~499 传统上是保留给系统帐号使用。

`-r`

此参数是用来建立系统帐号。的 UID 会比定义在系统档上 `/etc/login.defs` 的 `UID_MIN` 来的小。注意 `useradd` 此用法所建立的帐号不会建立使用者目录，也不会在乎纪录在 `/etc/login.defs` 的定义值。如果你想要有使用者目录须额外指定 `-m` 参数来建立系统帐号。这是 RED HAT 额外增设的选项。它会自动帮你选定一个小于 499 的 gid 不需要再加上 `-g` 参数。这是 RED HAT 额外增设的选项。

`-f`

This is force flag. 新增一个已经存在的群组帐号，系统会出现错误讯息然后结束 `groupadd`。如果是这样的情况，不会新增这个群组（如果是这个情况下，系统不会再新增一次）也可同时加上 `-g` 选项当你加上一个 gid，此时 gid 就不用是唯一值，可不加 `-o` 参数，建好群组后会显结果 (adding a group as neither `-g` or `-o` options were specified).

这是 RED HAT 额外增设的选项。

名称: `getkey`

函数名:

`char GetKey()`

功能:

等待一个按键。

参数：

无

返回值：

按键的码。对于数字键返回其 ASCII 码，其他功能键的定义如下

宏定义的值	说明
-------	----

VK_ESCAPE	取消
VK_ENTER	回车
VK_BACKSPACE	回退
VK_UP	上
VK_DOWN	下
VK_LEFT	左
VK_RIGHT	右

说明：

一直等待任意按键按下，没有光标显示。

例子：

下面的例子显示每次按键的键值，直到按下 ESC 为止。

```
#include "stdapi.h"

void tmain()
{
    byte key = 0;
    while (key != VK_ESCAPE) {
        key = GetKey();
        TextOut(0, 1, "Key: %02x", key);
    }
}
```

8) H 类指令

功能说明：显示与设定硬件时钟。

语 法：hwclock [--adjust][--debug][--directisa][--hctosys][--show][--systohc][--test]
[--utc][--version][--set --date=<日期与时间>]

补充说明：在 Linux 中有硬件时钟与系统时钟等两种时钟。硬件时钟是指主机板上的时钟设备，也就是通常可在 BIOS 画面设定的时钟。系统时钟则是指 kernel 中的时钟。当 Linux 启动时，系统时钟会去读取硬件时钟的设定，之后系统时钟即独立运作。所有 Linux 相关指令与函数都是读取系统时钟的设定。

参 数:

--adjust hwclock 每次更改硬件时钟时，都会记录在/etc/adjtime 文件中。使用--adjust 参数，可使 hwclock 根据先前的记录来估算硬件时钟的偏差，并用来校正目前的硬件时钟。

--debug 显示 hwclock 执行时详细的信息。

--directisa hwclock 预设从/dev/rtc 设备来存取硬件时钟。若无法存取时，可用此参数直接以 I/O 指令来存取硬件时钟。

--hctosys 将系统时钟调整为与目前的硬件时钟一致。

--set --date=<日期与时间> 设定硬件时钟。

--show 显示硬件时钟的时间与日期。

--systohc 将硬件时钟调整为与目前的系统时钟一致。

--test 仅测试程序，而不会实际更改硬件时钟。

--utc 若要使用格林威治时间，请加入此参数，hwclock 会执行转换的工作。

--version 显示版本信息。

9) I 类指令

10) J 类指令

11) K 类指令

名称: kill

使用权限: 所有使用者

使用方式:

kill [-s signal | -p] [-a] pid ...

kill -l [signal]

说明: kill 送出一个特定的信号 (signal) 给行程 id 为 pid 的行程根据该信号而做特定的动作, 若没有指定, 预设是送出终止 (TERM) 的信号

把计

-s (signal) : 其中可用的讯号有 HUP (1), KILL (9), TERM (15), 分别代表着重跑, 砍掉, 结束; 详细的信号可以用 kill -l

-p : 印出 pid , 并不送出信号

-l (signal) : 列出所有可用的信号名称

范例：

将 pid 为 323 的行程砍掉 (kill)：

```
kill -9 323
```

将 pid 为 456 的行程重跑 (restart)：

```
kill -HUP 456
```

12) L 类指令

名称 less

使用权限：所有使用者

使用方式：

```
less [Option] filename
```

说明：

less 的作用与 more 十分相似，都可以用来浏览文字档案的内容，不同的是 less 允许使用者来回卷动以浏览已经看过的部份，同时因为 less 并未在一开始就读入整个档案，因此在遇上大型档案的开启时，会比一般的文书编辑器(如 vi)来的快速。

名称： lpd

使用权限： 所有使用者

使用方式： **lpd** [-l] [#port]

lpd 是一个常驻的印表机管理程式，它会根据 **/etc/printcap** 的内容来管理本地或远端的印表机。**/etc/printcap** 中定义的每一个印表机必须在 **/var/lpd** 中有一个相对应的目录，目录中以 **cf** 开头的档案表示一个等待送到适当装置的印表工作。这个档案通常是由 **lpr** 所产生。

lpr 和 **lpd** 组成了一个可以离线工作的系统，当你使用 **lpr** 时，印表机不需要能立即可用，甚至不用存在。**lpd** 会自动监视印表机的状况，当印表机上线后，便立即将档案送交处理。这个得所有的应用程式不必等待印表机完成前一工作。

参数：

-l: 将一些除错讯息显示在标准输出上。

#port: 一般而言, lpd 会使用 getservbyname 取得适当的 TCP/IP port, 你可以使用这个参数强迫 lpd 使用指定的 port。

范例:

这个程式通常是由 /etc/rc.d 中的程式在系统启始阶段执行。

名称 lpq

-- 显示列表机贮列中未完成的工作 用法

lpq [l] [P] [user]

说明

lpq 会显示由 lpd 所管理的列表机贮列中未完成的项目。

范例

范例 1. 显示所有在 lp 列表机贮列中的工作

```
# lpq -PlpRank Owner Job Files Total Size1st root 238 (standard input) 1428646 bytes
```

相关函数

lpr, lpc, lpd

名称: lpr

使用权限: 所有使用者

使用方式: lpr [-P printer]

将档案或是由标准输入送进来的资料送到印表机贮列之中, 印表机管理程式 lpd 会在稍后将这个档案送给适当的程式或装置处理。lpr 可以用来将料资送给本地或是远端的主机来处理。

参数:

-p Printer: 将资料送至指定的印表机 Printer, 预设值为 lp。

范例：

将 www.c 和 kkk.c 送到印表机 lp。

```
lpr -Plp www.c kkk.c
```

名称 lprm

-- 将一个工作由印表机贮列中移除 用法

```
/usr/bin/lprm [P] [file...]
```

说明

尚未完成的印表机工作会被放在印表机贮列之中，这个命令可用来将尚未送到印表机的工作取消。由于每一个印表机都有一个独立的贮列，你可以用 **-P** 这个命令设定想要作用的印列机。如果没有设定的话，会使用系统预设的印表机。

这个命令会检查使用者是否有足够的权限删除指定的档案，一般而言，只有档案的拥有者或是系统管理员才有这个权限。

范例

将印表机 **hpprinter** 中的第 **1123** 号工作移除

```
lprm -Phpprinter 1123
```

将第 **1011** 号工作由预设印表机中移除

```
lprm 1011
```

指令名称 ln

使用权限：所有使用者

使用方式：ln [options] source dist，其中 option 的格式为：

```
[-bdfinsvF] [-S backup-suffix] [-V {numbered,existing,simple}]
```

```
[--help] [--version] [--]
```

说明：Linux/Unix 档案系统中，有所谓的连结(link)，我们可以将其视为档案的别名，而连结又可分为两种：硬连结(hard link)与软连结(symbolic link)，**硬连结的意思是一个档案可以有多个名称，而软连结的方式则是产生一个特殊的档案，该档案的内容是指向另一个档案的位置。硬连结是存在同一个档案系统中，而软连结却可以跨越**

不同的档案系统。

`ln source dist` 是产生一个连结(dist)到 `source`，至于使用硬连结或软连结则由参数决定。

不论是硬连结或软连结都不会将原本的档案复制一份，只会占用非常少量的磁碟空间。

-f：连结时先将与 `dist` 同档名的档案删除-d：允许系统管理者硬连结自己的目录-i：在删除与 `dist` 同档名的档案时先进行询问-n：在进行软连结时，将 `dist` 视为一般的档案-s：进行软连结(symbolic link)-v：在连结之前显示其档名-b：将在连结时会被覆写或删除的档案进行备份-S SUFFIX：将备份的档案都加上 SUFFIX 的字尾-V METHOD：指定备份的方式--help：显示辅助说明--version：显示版本

范例：

将档案 `yy` 产生一个 symbolic link：`zz`

```
ln -s yy zz
```

将档案 `yy` 产生一个 hard link：`zz`

```
ln yy xx
```

名称 locate

使用权限：所有使用者

使用方式：`locate [-q] [-d] [--database=]`

`locate [-r] [--regexp=]`

`locate [-qv] [-o] [--output=]`

`locate [-e] [-f] <[-l] [-c]`

`<[-U] [-u]>`

`locate [-Vh] [--version] [--help]`

说明：

`locate` 让使用者可以很快速的搜寻档案系统内是否有指定的档案。其方法是先建立一个包括系统内所有档案名称及路径的资料库，之后当寻找时就只需查询这个资料库，而不必实际深入档案系统之中了。

在一般的 `distribution` 之中，资料库的建立都被放在 `contab` 中自动执行。一般使用者在使用时只要用

```
# locate your_file_name
```

的型式就可以了。 参数：

-u

-U

建立资料库，-u 会由根目录开始，-U 则可以指定开始的位置。

-e

将
排除在寻找的范围之外。

-l
如果 **是 1**，则启动安全模式。在安全模式下，使用者不会看到权限无法看到的档案。这会始速度减慢，因为 **locate** 必须至实际的档案系统中取得档案的权限资料。

-f
将特定的档案系统排除在外，例如我们没有到理要把 **proc** 档案系统中的档案放在资料库中。

-q
安静模式，不会显示任何错误讯息。

-n
至多显示 **个** 输出。

-r
使用正规运算式 **做** 寻找的条件。

-o
指定资料库存的名称。

-d

指定资料库的路径

-h
显示辅助讯息

-v
显示更多的讯息

-V
显示程式的版本讯息 范例：

locate chdrv：寻找所有叫 **chdrv** 的档案

locate -n 100 a.out：寻找所有叫 **a.out** 的档案，但最多只显示 **100** 个

locate -u：建立资料库

名称 **ls**

使用权限：所有使用者

使用方式：`ls [-alrtAFR] [name...]`

说明：显示指定工作目录下之内容（列出目前工作目录所含之档案及子目录）。

-a 显示所有档案及目录（ls 内定将档案名或目录名称开头为"."的视为隐藏档，不会列出）

-l 除档案名称外，亦将档案型态、权限、拥有者、档案大小等资讯详细列出

-r 将档案以相反次序显示(原定依英文字母次序)

-t 将档案依建立时间之先后次序列出

-A 同 -a，但不列出 "."(目前目录) 及 ".."(父目录)

-F 在列出的档案名称后加一符号；例如可执行档则加 "*", 目录则加 "/"

-R 若目录下有档案，则以下之档案亦皆依序列出

范例：

列出目前工作目录下所有名称是 s 开头的档案，愈新的排愈后面：

```
ls -ltr s*
```

将 /bin 目录以下所有目录及档案详细资料列出：

```
ls -lR /bin
```

列出目前工作目录下所有档案及目录；目录于名称后加 "/"，可执行档于名称后加 "*"：

```
ls -AF
```

M 类指令

名称 last

使用权限：所有使用者

使用方式：`shell>> last [options]`

说明：显示系统开机以来获是从每月初登入者的讯息

把计

-R 省略 hostname 的栏位

-num 展示前 num 个

username 展示 username 的登入讯息

tty 限制登入讯息包含终端机代号

范例：

```
shell>> last -R -2
```

johnney pts/1 Mon Aug 14 20:42 still logged in

johnney pts/0 Mon Aug 14 19:59 still logged in

wtmp begins Tue Aug 1 09:01:10 2000 ### /var/log/wtmp

shell>> last -2 minery

minery pts/0 140.119.217.115 Mon Aug 14 18:37 - 18:40 (00:03)

minery pts/0 140.119.217.115 Mon Aug 14 17:22 - 17:24 (00:02)

wtmp begins Tue Aug 1 09:01:10 2000

名称 login

用途

启动用户会话。

语法

login [-h HostName] [-p] [-f User | -k] [User [Environment]]

描述

login 命令（**tsm** 命令的一部分）在系统中为指定在 **User** 参数中的用户启动会话。也可以指定添加到用户环境中的环境变量。这些是

Variable=Value 格式的字符串。**login** 命令通常不在命令行上输入。

注意：

PATH、**IFS**、**HOME** 和 **SHELL** 环境变量不能在命令行初始化。

login 命令支持多字节用户名。建议系统管理员将用户名限制为可移植字符集中的字符，以消除任何不明确地方。

如果 **/etc/nologin** 文件存在，那么系统会禁止用户登录，并显示 **/etc/nologin** 文件的内容。如果此文件存在，系统仍允许 **root**

用户登录。当重新引导系统的时候，**/etc/nologin** 文件被删除。

login 命令能够处理分布式计算环境（DCE）下长达 1024 字符的用户名。DCE 用户名存放在 **LOGIN** 环境变量中。由于 DCE

用户名不遵从标准操作系统的要求，DCE 用户名的前 8 个字符存放在所有标准操作系统文件和环境。

login 命令执行以下的功能：

检查帐户 **login** 命令为用于登录的端口验证用户帐户、确保认证、正确启用登录以及更正容量。

认证用户 **login**

命令通过为每个用户使用系统定义的认证方法来验证用户的身份。如果密码失效，那么用户必须提供新的密码。如果定义了辅助认证方法，为了登录进系统，这些方法会被调用，但是不需成功返回。

建立凭证 **login** 命令从用户数据库中为用户创建初始化凭证。这些凭证定义了用户在系统中的访问权限和义务。

启动会话 **login** 命令从用户数据库、从命令行以及从 **/etc/environment**

配置文件初始化用户环境；将当前目录更改为用户的主目录（正常情况下）；并运行用户初始化程序。

这些函数以给定的顺序执行；如果其中之一失败，那么它以后的函数就不会执行。

当用户成功登录的时候，**login** 命令在跟踪用户登录的 **/etc/utmp** 文件以及用于记帐用途的 **/var/adm/wtmp** 文件中构造项。**login**

命令还设置 **LOGIN** 和 **LOGNAME** 环境变量。

关于每次失败登录的信息记录在 **/etc/security/failedlogin** 文件中。除了未被承认的用户名被记录为 **UNKNOWN_USER**

之外，存储的信息与 **/etc/utmp** 文件中的相同。例如，这确保了不允许作为用户名输入的密码未经加密的意外地进入系统。

成功登录后，**login** 命令显示了本帐户上次成功以及失败登录试图的日期、日期和时间，以及本帐户上次成功登录后总共失败登录试图次数的信息。如果在主目录中有

.hushlogin 文件，那么不会发布这些信息。

login 命令还更改了用户登录端口的所有权。这包含了任何在 **/etc/security/login.cfg** 文件中注释同义的端口。

为了保证系统的完整性，一次仅允许一个会话登录进一个端口。这意味着从 **shell** 提示输入的 **login**

命令无法成功，因为原始会话和新建的登录会话在同一个端口上。然而，**exec login** 命令会执行成功，因为新的 **shell** 会代替了当前的

shell。**login** 命令是一个典型的内置 **shell** 命令，促使 **shell** 替换其自身。

注意：除非终端显示器只能显示大写字母，否则用户名不应该只由大写字母组成。

为了使用多字节用户名登录，必须首先打开一个日语视窗（**aixterm**），然后从日语视窗启动新的登录。

标志

-f User 标志已经认证过的用户。如果登录进程的真实标识为根（0），那么，该用户没有认证。

-h HostName 将登录标识为远程登录，使用 **HostName** 变量来指定请求登录的机器名。这种格式的登录
仅由 **telnetd** 和

rlogind 守护进程使用。

-k 将登录标识为使用 **Kerberos** 认证，并使登录将控制权传递给 **/usr/bin/k5dclogin** 来处理认证。只有
krshd

守护进程才使用这种登录形式。

-p 通过将当前终端类型设置为 **\$TERM** 环境变量的值来取代包含在 **CuAt/PdAt** 对象类数据库中的类型
来保护它。

安全性

访问控制：此命令为 **root** 用户设置 **setuid** 许可，将可执行（x）权限授权给任何用户，它位于可信计算库中。

示例

为了作为用户 **jamesd** 登录进系统，请在登录提示符下输入以下命令：

login: jamesd 如果定义了密码，那么会出现密码提示符。在该提示符下输入密码。

13) M 类指令

名称：**more**

使用权限：所有使用者

使用方式：**more [-dlfpcsu] [-num] [+ /pattern] [+linenum] [fileNames..]**

说明：类似 **cat**，不过会以一页一页的显示方便使用者逐页阅读，而最基本的指令就是按空白键（**space**）就往

下一页显示，按 **b** 键就会往回 (**back**) 一页显示，而且还有搜寻字串的功能 (与 **vi** 相似)，使用中的说明文件，请按 **h** 。

参数: **-num** 一次显示的行数

-d 提示使用者，在画面下方显示 **[Press space to continue, q to quit.]**，如果使用者按错键，则会显示 **[Press h for instructions.]** 而不是 哔 声

-l 取消遇见特殊字元 **^L** (送纸字元) 时会暂停的功能

-f 计算行数时，以实际上的行数，而非自动换行过后的行数 (有些单行字数太长的会被扩展为两行或两行以上)

-p 不以卷动的方式显示每一页，而是先清除萤幕后再显示内容

-c 跟 **-p** 相似，不同的是先显示内容再清除其他旧资料

-s 当遇到有连续两行以上的空白行，就代换为一行的空白行

-u 不显示下引号 (根据环境变数 **TERM** 指定的 **terminal** 而有所不同)

+/ 在每个档案显示前搜寻该字串 (**pattern**)，然后从该字串之后开始显示

+num 从第 **num** 行开始显示

fileNames 欲显示内容的档案，可为复数个数

范例:

more -s testfile 逐页显示 **testfile** 之档案内容，如有连续两行以上空白行则以一行空白行显示。

more +20 testfile 从第 **20** 行开始显示 **testfile** 之档案内容。

名称: mv

使用权限: 所有使用者

使用方式:

mv [options] source dest

mv [options] source... directory

说明: 将一个档案移至另一档案，或将数个档案移至另一目录。

参数: **-i** 若目的地已有同名档案，则先询问是否覆盖旧档。

范例:

将档案 **aaa** 更名为 **bbb** :

mv aaa bbb

将所有的 C 语言程式移至 **Finished** 子目录中 :

mv -i *.c

名称: mail

使用权限: 所有使用者

使用方式：mail [-iInv] [-s subject] [-c cc-addr] [-b bcc-addr] user1 [user 2 ...]

说明：

mail 不仅只是一个指令，mail 还是一个电子邮件程式，不过利用 mail 来读信的人应该很少吧！对于系统管理者来说 mail 就很有用，因为管理者可以用 mail 写成 script，定期寄一些备忘录提醒系统的使用者。

参数：

i 忽略 tty 的中断讯号。(interrupt)

I 强迫设成互动模式。(Interactive)

v 列印出讯息，例如送信的地点、状态等等。(verbose)

n 不读入 mail.rc 设定档。

s 邮件标题。

c cc 邮件地址。

b bcc 邮件地址。

范例：

将信件送给一个或以上的电子邮件地址，由于没有加入其他的选项，使用者必须输入标题与信件的内容等。而 user2 没有主机位置，就会送给邮件伺服器的 user2 使用者。

```
mail user1@email.address
```

```
mail user1@email.address user2
```

将 mail.txt 的内容寄给 user2 同时 cc 给 user1。如果将这一行指令设成 cronjob 就可以定时将备忘录寄给系统使用者。

```
mail -s 标题 -c user1 user2 < mail.txt
```

指令：mesg

使用权限：所有使用者

使用方式：mesg [y|n]

说明：决定是否允许其他人传讯息到自己的终端机介面

把计

y：允许讯息传到终端机介面上。

n：不允许讯息传到终端机介面上。

如果没有设定，则讯息传递与否则由终端机界面目前状态而定。

例子：

改变目前讯息设定，改成不允许讯息传到终端机介面上：

`mesg n`

与 `mesg` 相关的指令有：`talk`，`write`，`wall`。

名称：**mkdosfs**

使用权限： 所有使用者

使用方式：`mkdosfs [-c | -l filename]`

`[-f number_of_FATs]`

`[-F FAT_size]`

`[-i volume_id]`

`[-m message_file]`

`[-n volume_name]`

`[-r root_dir_entry]`

`[-s sector_per_cluster]`

`[-v]`

`device`

`[block_count]`

说明： 建立 DOS 档案系统。`device` 指你想要建立 DOS 档案系统的装置代号。像是 `/dev/hda1` 等等。

`block_count` 则是你希望配置的区块数。如果 `block_count` 没有指定则系统会自动替你计算符合该装置大小的区块数。

参数:

-c 建立档案系统之前先检查是否有坏轨。

-l 从得定的档案中读取坏轨记录。

-f 指定档案配置表 (FAT, File Allocation Table) 的数量。预设值为 2。目前 Linux 的 FAT 档案系统不支援超过 2 个 FAT 表。通常这个不需要改。

-F 指定 FAT 表的大小, 通常是 12 或是 16 个位元组。12 位元组通常用于磁碟片, 16 位元组用于一般硬碟的分割区, 也就是所谓的 FAT16 格式。这个值通常系统会自己选定适当的值。在磁碟片上用 FAT16 通常不会发生作用, 反之在硬碟上用 FAT12 亦然。

-i 指定 Volume ID。一般是一个 4 个位元组的数字, 像是 2e203a47。如果不给系统会自己产生。

-m 当使用者试图用这片磁片或是分割区开机, 而上面没有作业系统时, 系统会给使用者一段警告讯息。这个参数就是用来变更这个讯息的。你可以先用档案编辑好, 然后用这个参数指定, 或是用

-m -

这样系统会要求你直接输入这段文字。要特别注意的是, 档案里的字串长度不要超过 418 个字, 包括展开的跳栏符号 (TAB) 和换行符号 (换行符号在 DOS 底下算两个字元!)

-n 指定 Volume Name, 就是磁碟标签。如同在 DOS 底下的 format 指令一样, 给不给都可以。没有预设值。

-r 指定根目录底下的最大档案数。这里所谓的档案数包括目录。预设值是在软碟上是 112 或是 224, 在硬碟上是 512。没事不要改这个数字。

-s 每一个磁丛 (cluster) 的磁区数。必须是 2 的次方数。不过除非你知道你在作什么, 这个值不要乱给。

-v 提供额外的讯息

范例:

mkdosfs -n Tester /dev/fd0 将 A 槽里的磁碟片格式化为 DOS 格式, 并将标签设为 Tester

名称: mknod

用途

创建特殊文件。

语法

只能由 root 用户或系统组成员运行

mknod Name { b | c } Major Minor

创建 FIFO (已命名的管道)

`mknod Name { p }`

描述

`mknod` 命令建立一个目录项和一个特殊文件的对应索引节点。第一个参数是 `Name` 项设备的名称。选择一个描述性的设备名称。`mknod` 命令有两种形式，它们有不同的标志。

`mknod` 命令的第一种形式只能由 `root` 用户或系统组成员执行。在第一种形式中，使用了 `b` 或 `c` 标志。`b` 标志表示这个特殊文件是面向块的设备（磁盘、软盘或磁带）。`c` 标志表示这个特殊文件是面向字符的设备（其它设备）。

第一种形式的最后两个参数是指定主设备的数目，它帮助操作系统查找设备驱动程序代码，和指定次设备的数目，也就是单元驱动器或行号，它们是十进制或八进制的。一个设备的主要和次要编号由该设备的配置方法分配，它们保存在 `ODM` 中的 `CuDvDr` 类里。在这个对象类中定义了主要和次要编号以确保整个系统设备定义的一致性，这是很重要的。

在 `mknod` 命令的第二种形式中，使用了 `p` 标志来创建 `FIFO`（已命名的管道）。

标志

`b` 表示特殊文件是面向块的设备（磁盘、软盘或磁带）。

`c` 表示特殊文件是面向字符的设备（其它设备）。

`p` 创建 `FIFO`（已命名的管道）。

示例

要创建一个新软盘驱动器的特殊文件，输入：

```
mknod /dev/fd2 b 1 2
```

这创建了 `/dev/fd2` 特殊文件，它是一个特殊块文件，主设备号为 1，次设备号为 2。

文件

`/usr/sbin/mknod` 包含 `mknod` 命令。

名称： **mformat**

使用权限： 所有使用者

使用方式：

```
mformat [-t cylinders] [-h heads] [-s sectors] [-l volume_label] [-F] [-l fsVer-sion] [-S sizecode] [-2 sectors_on_track_0] [-M software_sector_size] [-a] [-X] [-C] [-H hidden_sectors] [-r root_sectors] [-B boot_sector] [-O rate_on_track_0] [-A rate_on_other_tracks] [-1] [-k] drive:
```

在已经做过低阶格式化的磁片上建立 DOS 档案系统。如果在编译 `mttools` 的时候把 `USE_2M` 的参数打开，部分与 2M 格式相关的参数就会发生作用。否则这些参数（像是 `S,2,1,M`）不会发生作用。

参数：

- t 磁柱（`synlinder`）数
- h 磁头（`head`）数
- s 每一磁轨的磁区数
- l 标签
- F 将磁碟格式化为 `FAT32` 格式，不过这个参数还在实验中。
- l 设定 `FAT32` 中的版本号。这当然也还在实验中。
- S 磁区大小代码，计算方式为 $\text{sector} = 2^{(\text{大小代码} + 7)}$
- c 磁丛（`cluster`）的磁区数。如果所给定的数字会导致磁丛数超过 `FAT` 表的限制，`mformat` 会自动放大磁区数。
- s
- M 软体磁区大小。这个数字就是系统回报的磁区大小。通常是和实际的大小相同。
- a 如果加上这个参数，`mformat` 会产生一组 `Atari` 系统的序号给这块软碟。
- X 将软碟格式化成 `XDF` 格式。使用前必须先先用 `xdfcopy` 指令对软碟作低阶格式化的动作。
- C 产生一个可以安装 `MS-DOS` 档案系统的磁碟影像档（`disk image`）。当然对一个实体磁碟机下这个参数是没有意义的。
- H 隐藏磁区的数目。这通常适用在格式化硬碟的分割区时，因为通常一个分割区的前面还有分割表。这个参数未经测试，能不用就不用。
- n 磁碟序号
- r 根目录的大小，单位是磁区数。这个参数只对 `FAT12` 和 `FAT16` 有效。
- B 使用所指定的档案或是设备的开机磁区做为这片磁片或分割区的开机磁区。当然当中的硬体参数会随之更动。
- k 尽量保持原有的开机磁区。
- 0 第 0 轨的资料传输率
- A 第 0 轨以外的资料传输率
- 2 使用 2m 格式
- 1 不使用 2m 格式

范例：

`mformat a:`

这样会用预设值把 `a:`（就是 `/dev/fd0`）里的磁碟片格式化。

14) N 类指令

名称： `nice`

使用权限： 所有使用者

使用方式：nice [-n adjustment] [-adjustment] [--adjustment=adjustment] [--help] [--version] [command [arg...]]

说明：以更改过的优先序来执行程式，如果未指定程式，则会印出目前的排程优先序，内定的 **adjustment** 为 10，范围为 -20 (最高优先序) 到 19 (最低优先序)

把计

-n adjustment, -adjustment, --adjustment=adjustment 皆为将该原有优先序的增加 adjustment

--help 显示求助讯息

--version 显示版本资讯

范例：

将 ls 的优先序加 1 并执行：

nice -n 1 ls

将 ls 的优先序加 10 并执行：

nice ls 将 ls 的优先序加 10 并执行

注意：优先序 (priority) 为作业系统用来决定 CPU 分配的参数，Linux 使用『回合制(round-robin)』的演算法来做 CPU 排程，优先序越高，所可能获得的 CPU 时间就越多。

15) 0 类指令

16) P 类指令

名称：passwd

使用权限：所有使用者

使用方式：**passwd** [-k] [-l] [-u [-f]] [-d] [-S] [username]

说明：用来更改使用者的密码

参数：

-k

-l

-u

-f

-d 关闭使用者的密码认证功能，使用者在登入时将可以不用输入密码，只有具备 **root** 权限的使用者方可使用。

-S 显示指定使用者的密码认证种类，只有具备 **root** 权限的使用者方可使用。

[username] 指定帐号名称。

名称：ps

使用权限：所有使用者

使用方式：ps [options] [--help]

说明：显示瞬间行程 (process) 的动态

参数：

ps 的参数非常多，在此仅列出几个常用的参数并大略介绍含义

-A 列出所有的行程

-w 显示加宽可以显示较多的资讯

-au 显示较详细的资讯

-aux 显示所有包含其他使用者的行程

au(x) 输出格式：

USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND

USER: 行程拥有者

PID: pid

%CPU: 占用的 CPU 使用率

%MEM: 占用的记忆体使用率

VSZ: 占用的虚拟记忆体大小

RSS: 占用的记忆体大小

TTY: 终端的次要装置号码 (minor device number of tty)

STAT: 该行程的状态:

D: 不可中断的静止 (通常等待 b 进行 I/O 动作)

R: 正在执行中

S: 静止状态

T: 暂停执行

Z: 不存在但暂时无法消除

W: 没有足够的记忆体分页可分配

<: 高优先序的行程

N: 低优先序的行程

L: 有记忆体分页分配并锁在记忆体内 (即时系统或握 A I/O)

START: 行程开始时间
TIME: 执行的时间
COMMAND: 所执行的指令

范例:

```
ps
PID TTY TIME CMD
2791 tty0 00:00:00 tcsh
3092 tty0 00:00:00 ps
% ps -A
PID TTY TIME CMD
1 ? 00:00:03 init
2 ? 00:00:00 kflushd
3 ? 00:00:00 kpiod
4 ? 00:00:00 kswapd
5 ? 00:00:00 mdrecoveryd
.....
% ps -aux
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.0 0.7 1096 472 ? S Sep10 0:03 init [3]
root 2 0.0 0.0 0 0 ? SW Sep10 0:00 [kflushd]
root 3 0.0 0.0 0 0 ? SW Sep10 0:00 [kpiod]
root 4 0.0 0.0 0 0 ? SW Sep10 0:00 [kswapd]
.....
```

名称: pstree

使用权限: 所有使用者

使用方式:

pstree [-a] [-c] [-h|-Hpid] [-l] [-n] [-p] [-u] [-G|-U] [pid|user]

pstree -V

说明: 将所有行程以树状图显示, 树状图将会以 **pid** (如果有指定) 或是以 **init** 这个基本行程为根 (**root**) , 如果有指定使用者 **id** , 则树状图会只显示该使用者所拥有的行程

参数:

-a 显示该行程的完整指令及参数, 如果是被记忆体置换出去的行程则会加上括号

-c 如果有重覆的行程名, 则分开列出 (预设值是会在前面加上 *)

范例:

```
pstree
```

```
init-+-amd
```

```
| -apmd
|-atd
|-httpd---10*[httpd]
%pstree -p
init(1)-+-amd(447)
|-apmd(105)
|-atd(339)
%pstree -c
init-+-amd
|-apmd
|-atd
|-httpd-+-httpd
| |-httpd
| |-httpd
| |-httpd
....
```

17) Q 类指令

18) R 类指令

名称：rm

使用权限：所有使用者

使用方式：rm [options] name...

说明：删除档案及目录。

把计

-i 删除前逐一询问确认。

-f 即使原档案属性设为唯读，亦直接删除，无需逐一确认。

-r 将目录及以下之档案亦逐一删除。

范例：

删除所有 C 语言程式档；删除前逐一询问确认：

```
rm -i *.c
```

将 **Finished** 子目录及子目录中所有档案删除：

```
rm -r Finished
```

名称：rmdir

使用权限：于目前目录有适当权限的所有使用者

使用方式： **rmdir [-p] dirName**

说明： 删除空的目录。

参数： **-p** 是当子目录被删除后使它也成为空目录的话，则顺便一并删除。

范例：

将工作目录下，名为 **AAA** 的子目录删除：

```
rmdir AAA
```

在工作目录下的 **BBB** 目录中，删除名为 **Test** 的子目录。若 **Test** 删除后，**BBB** 目录成为空目录，则 **BBB** 亦予删除。

```
rmdir -p BBB/Test
```

名称：renice

使用权限：所有使用者

使用方式： **renice priority [[-p] pid ...] [[-g] pgrp ...] [[-u] user ...]**

说明：重新指定一个或多个行程(Process)的优先序(一个或多个将根据所下的参数而定)

把计

-p pid 重新指定行程的 **id** 为 **pid** 的行程的优先序

-g pgrp 重新指定行程群组(process group)的 **id** 为 **pgrp** 的行程 (一个或多个) 的优先序

-u user 重新指定行程拥有者为 **user** 的行程的优先序

范例：

将行程 **id** 为 **987** 及 **32** 的行程与行程拥有者为 **daemon** 及 **root** 的优先序号码加 **1**：

```
renice +1 987 -u daemon root -p 32
```


注意：每一个行程(Process)都有一个唯一的 (unique) id

名称：rename

用途：

修改文件的文件名

具体的用法：

例如现在已经有 file1 file2 file3.....file100

Rename file file00 file?

第一个 file 表示是你在文件名中修改的部分

第二个 file 表示是你要修改的目的文件是什么

第三个 file? 表示你当前拥有的 file 名称

名称：rpcinfo

用途

报告远程过程调用（RPC）服务器的状态。

语法

显示统计信息列表

/usr/bin/rpcinfo [[-m](#) | [-s](#)] [Host]

显示已注册 RPC 程序列表

/usr/bin/rpcinfo [-p](#) [Host]

报告传输

/usr/bin/rpcinfo [-T](#) transport Host Prognum [Versnum]

显示条目列表

/usr/bin/rpcinfo [-l](#) [[-T](#) transport] Host Prognum Versnum

报告程序状态

/usr/bin/rpcinfo [[-n](#) PortNum] [-u](#) Host Prognum [Versnum]

报告响应状态

/usr/bin/rpcinfo [[-n](#) PortNum] [-t](#) Host Prognum [Versnum]

显示所有运行指定程序版本的主机

/usr/bin/rpcinfo [[-b](#)] [[-T](#) transport] Prognum Versnum

删除服务的注册

/usr/bin/rpcinfo [[-d](#)] [[-T](#) transport] Prognum Versnum

描述

rpcinfo 命令造成 RPC 调用到 RPC 服务器并报告服务器状态。例如，本命令报告服务器是否就绪并处于等待中或不可用。

程序参数可以为名称或号码。如果指定版本，rpcinfo 命令试图调用指定程序的那个版本。否则，rpcinfo 命令试图查找所有通过调用版本 0（零）来指定的程序的已注册版本号，然后试图调用所有已注册的版本。（版本 0 假定为不存在。如果确实存在，rpcinfo 命令则试图通过调用非常高的版本号来获得此信息。）

标志

-b 将 RPC 广播到指定的程序号和版本号的过程 0 并报告响应的所有主机。如果指定传输，它仅广播它的指定传输上的请求。如果任何传输不支持广播则打印错误消

	息。由于对别的系统产生可能的不利影响，应该限制使用广播（-b 标志）。
-d	删除指定程序号和版本号的 RPC 服务的注册。如果使用传输，只注销那个传输上的服务，否则注销所有注册的传输上的服务。本选项仅可由 root 用户执行。
-l	显示在指定主机上的指定程序号和版本号的条目的列表。返回所有使用与用于联系远程 portmap 守护程序的协议在同一个协议系列的传输的条目。本标志只适用于 AIX 4.2.1 或更高。
-m	显示指定主机上的端口映射操作统计信息表。此表包含对每个版本的端口映射（版本 2、3 和 4）的统计信息、每个过程请求和成功服务的次数、所做的远程调用请求的类型和数目和关于已处理的 RPC 地址查询的信息。本信息用来监视在主机上的 RPC 活动。本标志只适用于 AIX 4.2.1 或更高。
-n Portnum	使用 Portnum 参数作为 -t 和 -u 选项的端口号来替换端口映射给出的端口号。使用 -n 选项来避免到远程端口映射查找服务地址的调用。-a 选项废弃本选项。
-p	探测使用端口映射协议版本 2 的主机上的 portmap 服务并显示所有已注册的 RPC 程序的列表。如果不指定主机，缺省为本地主机。
-s	显示主机上所有已注册的 RPC 程序的简明列表。如果不指定主机，缺省为本地主机。本标志只适用于 AIX 4.2.1 或更高。
-t	执行一个到使用 TCP 的指定主机上的程序号的过程 0 的 RPC 调用，并报告是否收到响应。当使用在第三条语法中显示的 -T 选项时废弃本选项。
-T	指定需要服务的传输。本标志只适用于 AIX 4.2.1 或更高。
-u	执行一个到使用 UDP 的指定主机上的程序号的过程 0 的 RPC 调用，并报告是否收到响应。当使用在第三条语法中显示的 -T 选项时废弃本选项。

示例

显示在本地机器上注册的所有 RPC 服务，输入：

```
rpcinfo -p
```

显示在指定机器上注册的所有 RPC 服务，输入：

```
rpcinfo -p zelda
```

在本示例中，rpcinfo 显示在名为 zelda 的机器上注册的所有 RPC 服务。

显示本地网络上运行指定服务器的某个版本的所有机器，输入：

```
rpcinfo -b ypserv 2
```

在本示例中，rpcinfo 命令显示运行 ypserv 守护程序的版本 2 的所有机器的列表。

删除服务注册，输入：

```
rpcinfo -d sprayd 1
```

在本示例中，rpcinfo 命令删除 sprayd 守护程序的版本 1。

19) S 类指令

名称：sed

用途

提供流编辑器。

语法

`sed [-n] Script [File ...]`

`sed [-n] [-e Script] ... [-f ScriptFile] ... [File ...]`

描述

`sed` 命令根据编辑脚本修改来自指定的 `File` 参数的行，并将其写到标准输出。`sed` 命令包含很多功能，用于选择要修改的行，并只对选择的行作更改。

`sed` 命令使用两个工作空间来保留修改的行：保留选定行的模式空间和暂时存储行的保留空间。

编辑脚本由单独的子命令构成，每个单独的行上有一个子命令。`sed` 子命令的一般格式如下：

`[address-range] function[modifiers]`

`sed` 命令通过将一个输入行读入方式空间，依次应用所有的 `sed` 子命令（这些子命令的地址选择了该行），并将方式空间写到标准输出来处理每个输入的 `File`

参数。然后清除方式空间，并对在输入的 `File` 参数中指定的每行重复该过程。一些 `sed` 子命令使用保留空间来保存后继检索的所有或部分模式空间。

当命令包含地址（行号或搜索模式）时，该命令只会对被寻址的行起作用。否则，该命令适用于所有的行。

地址是十进制行号、`$`（美元符号），它表示输入的最后行的地址或上下文地址。上下文地址是与 `ed` 命令中所使用的表达式相似的正则表达式，除了具有以下的差别之外：

您可以为模式选择字符定界符。表达式的一般格式是：

`\?pattern?`其中 `?`（问号）是可选的字符定界符。除了空格或换行符，您可以从当前的语言环境中选择任何字符。仅在第一次出现 `?` 时要求 `\`（反斜杠）字符

模式的缺省格式如下：

`/pattern\`（反斜杠）字符不是必需的。

`\n` 序列与模式空间中的换行符（除了终止的换行符）相匹配。

`.`（句点）与除终止的换行符之外的任何字符匹配。也就是说，与 `ed` 命令不同（`ed` 命令不能匹配行中间的换行符），`sed` 命令能够匹配模式空间中的换行符。

某些叫做 `addressed` 的命令允许您指定该命令适用的一行或多行。下列的规则适用于寻址命令：

没有地址的命令行选择每一行。

具有以上下文格式表示的一个地址的命令行选择与该地址匹配的每一行。

具有以逗号分离的两个地址的命令行选择从与第一个地址匹配的第一行到与第二个地址匹配的下一行之间的整个范围。（如果第二个地址小于或等于第一个选择的行号，那么只会选择一行。）此后就会重复这个过程以再次寻找第一个地址。

标志

`-e Script` 使用 `Script` 变量作为编辑脚本。如果您只使用一个 `-e` 标志并且不使用 `-f` 标志，则可以省略 `-e` 标志。

`-f ScriptFile` 使用 `ScriptFile` 变量作为编辑脚本源文件。`ScriptFile` 变量是适用于 `File` 参数的一组准备好的编辑命令。

`-n` 禁止所有正常地写到标准输出的信息。

例如：

`Sed -n `2p` file1` 显示 `file1` 的第二行

名称：split

使用权限：所有使用者

使用方式: `split [OPTION] [INPUT [PREFIX]]`

说明:

将一个档案分割成数个。而从 `INPUT` 分割输出成固定大小的档案, 其档名依序为 `PREFIXaa`, `PREFIXab...`; `PREFIX` 预设值为 ``x`。若没有 `INPUT` 档或为 ``-`, 则从标准输入读进资料。

匡兜

`-b, --bytes=SIZE`

`SIZE` 值为每一输出档案的大小, 单位为 `byte`。

`-C, --line-bytes=SIZE`

每一输出档中, 单行的最大 `byte` 数。

`-l, --lines=NUMBER`

`NUMBER` 值为每一输出档的列数大小。

`-NUMBER`

与 `-l NUMBER` 相同。

`--verbose`

于每个输出档被开启前, 列印出侦错资讯到标准错误输出。

`--help`

显示辅助资讯然后离开。

`--version`

列出版本资讯然后离开。

`SIZE` 可加入单位: `b` 代表 512, `k` 代表 1K, `m` 代表 1 Meg。

范例:

PostgreSQL 大型资料库备份与回存:

因 PostgreSQL 允许表格大过你系统档案的最大容量, 所以要将表格 `dump` 到单一的档案可能会有问题, 使用 `split` 进行档案分割。

```
% pg_dump dbname | split -b 1m - filename.dump.
```

重新载入

```
% createdb dbname
```

```
% cat filename.dump.* | pgsqldb dbname
```

名称: **skill**

使用权限: 所有使用者

使用方式: **skill** [signal to send] [options] 选择程序的规则

说明:

送个讯号给正在执行的程序,预设的讯息为 **TERM** (中断), 较常使用的讯息为 **HUP**, **INT**, **KILL**, **STOP**, **CONT**, 和 **0**

讯息有三种写法:分别为 **-9**, **-SIGKILL**, **-KILL**, 可以使用 **-I** 或 **-L** 已列出可使用的讯息。

一般参数:

-f 快速模式/尚未完成

-i 互动模式/ 每个动作将要被确认

-v 详细输出/ 列出所选择程序的资讯

-w 智能警告讯息/ 尚未完成

-n 没有动作/ 显示程序代号

参数: 选择程序的规则可以是, 终端机代号,使用者名称,程序代号,命令名称。

-t 终端机代号 (**tty** 或 **pty**)

-u 使用者名称

-p 程序代号 (**pid**)

-c 命令名称 可使用的讯号:

以下列出已知的讯号名称,讯号代号,功能。

名称 (代号) 功能/ 描述

ALRM 14 离开

HUP 1 离开

INT 2 离开

KILL 9 离开/ 强迫关闭

PIPE 13 离开

POLL 离开

PROF 离开

TERM 15 离开

USR1 离开

USR2 离开

VTALRM 离开

STKFLT 离开/ 只适用于 i386, m68k, arm 和 ppc 硬体

UNUSED 离开/ 只适用于 i386, m68k, arm 和 ppc 硬体

TSTP 停止 /产生与内容相关的行为

TTIN 停止 /产生与内容相关的行为

TTOU 停止 /产生与内容相关的行为

STOP 停止 /强迫关闭

CONT 从新启动 /如果在停止状态则从新启动,否则忽略

PWR 忽略 /在某些系统中会离开

WINCH 忽略

CHLD 忽略

ABRT 6 核心

FPE 8 核心

ILL 4 核心

QUIT 3 核心

SEGV 11 核心

TRAP 5 核心

SYS 核心 /或许尚未实作

EMT 核心 /或许尚未实作

BUS 核心 /核心失败

XCPU 核心 /核心失败

XFSZ 核心 /核心失败

范例：

停止所有在 PTY 装置上的程序

```
skill -KILL -v pts/*
```

停止三个使用者 user1 , user2 , user3

```
skill -STOP user1 user2 user3
```

其他相关的命令: kill

名称 showmount

用法: showmount -e 主机名或 IP 地址

名称 sort

说明: sort 命令对指定文件中所有的行进行排序, 并将结果显示在标准输出上。如不指定输入文件或使用“-”, 则表示排序内容来自标准输入。

sort 排序是根据从输入行抽取的一个或多个关键字进行比较来完成的。排序关键字定义了用来排序的最小的字符序列。缺省情况下以整行为关键字按 ASCII 字符顺序进行排序。

改变缺省设置的选项主要有：

- m 若给定文件已排好序，合并文件。
- c 检查给定文件是否已排好序，如果它们没有都排好序，则打印一个出错信息，并以状态值 1 退出。
- u 对排序后认为相同的行只留其中一行。（这是一个常用参数，可能去除文件中的重复行）
- o 输出文件 将排序输出写到输出文件中而不是标准输出，如果输出文件是输入文件之一，sort 先将该文件的内容写入一个临时文件，然后再排序和写输出结果。

改变缺省排序规则的选项主要有：

- d 按字典顺序排序，比较时仅字母、数字、空格和制表符有意义。
- f 将小写字母与大写字母同等对待。
- I 忽略非打印字符。
- M 作为月份比较：“JAN”<“FEB”
- r 按逆序输出排序结果。
- +pos1 - pos2 指定一个或几个字段作为排序关键字，字段位置从 pos1 开始，到 pos2 为止（包括 pos1，不包括 pos2）。如不指定 pos2，则关键字为从 pos1 到行尾。字段和字符的位置从 0 开始。
- b 在每行中寻找排序关键字时忽略前导的空白（空格和制表符）。
- t separator 指定字符 separator 作为字段分隔符。

例如：

Sort -g file1 排列 file1 从小到大显示里面的内容，

Sort -f file1 逐行比较 file1 内部的内容，然后排列从小到大输出。

名称 Sleep, usleep

Sleep 60 单位是秒

Usleep 60 单位是微妙

20) T 类指令

名称：touch

使用权限：所有使用者

使用方式：

touch [-acfm]

[-r reference-file] [--file=reference-file]

[-t MMDDhhmm[[CC]YY][.ss]]

[-d time] [--date=time] [--time={atime,access,use,mtime,modify}]

[--no-create] [--help] [--version]

file1 [file2 ...]

说明：

touch 指令改变档案的时间记录。ls -l 可以显示档案的时间记录。

参数：

a 改变档案的读取时间记录。

m 改变档案的修改时间记录。
c 假如目的档案不存在，不会建立新的档案。与 `--no-create` 的效果一样。
f 不使用，是为了与其他 `unix` 系统的相容性而保留。
r 使用参考档的时间记录，与 `--file` 的效果一样。
d 设定时间与日期，可以使用各种不同的格式。
t 设定档案的时间记录，格式与 `date` 指令相同。
`--no-create` 不会建立新档案。
`--help` 列出指令格式。
`--version` 列出版本讯息。

范例：

最简单的使用方式，将档案的时间记录改为现在的时间。若档案不存在，系统会建立一个新的档案。

`touch file`

`touch file1 file2`

将 `file` 的时间记录改为 5 月 6 日 18 点 3 分，公元两千年。时间的格式可以参考 `date` 指令，至少需输入 `MMDDHHmm`，就是月日時与分。

`touch -c -t 05061803 file`

`touch -c -t 050618032000 file`

将 `file` 的时间记录改变成与 `referencefile` 一样。

`touch -r referencefile file`

将 `file` 的时间记录改成 5 月 6 日 18 点 3 分，公元两千年。时间可以使用 `am`, `pm` 或是 24 小时的格式，日期可以使用其他格式如 `6 May 2000`。

`touch -d "6:03pm" file`

`touch -d "05/06/2000" file`

`touch -d "6:03pm 05/06/2000" file`

名称： `time`

使用权限： 所有使用者

使用方式： `time [options] COMMAND [arguments]`

说明： `time` 指令的用途，在于量测特定指令执行时所需消耗的时间及系统资源等资讯。例如 `CPU` 时间、记忆体、输入输出等等。需要特别注意的是，部分资讯在 `Linux` 上显示不出来。这是因为在 `Linux` 上部分资源的分配函数与 `time` 指令所预设的方式并不相同，以致于 `time` 指令无法取得这些资料。

把计

`-o` or `--output=FILE`

设定结果输出档。这个选项会将 `time` 的输出写入 所指定的档案中。如果档案已经存在，系统将覆写其内容。

`-a` or `--append`

配合 `-o` 使用，会将结果写到档案的末端，而不会覆盖掉原来的内容。

`-f FORMAT` or `--format=FORMAT`

以 `FORMAT` 字串设定显示方式。当这个选项没有被设定的时候，会用系统预设的格式。不过你可以用环境变量 `time` 来设定这个格式，如此一来就不必每次登入系统都要设定一次。

一般设定上，你可以用

\t

表示跳栏，或者用

\n

表示换行。每一项资料要用 % 做为前导。如果要在字串中使用百分比符号，就用 %。（学过 C 语言的人大概会觉得很有趣）

time 指令可以显示的资源有四大项，分别是：

Time resources

Memory resources

IO resources

Command info

详细的内容如下：

Time Resources

E 执行指令所花费的时间，格式是：[hour]:minute:second。请注意这个数字并不代表实际的 CPU 时间。

e 执行指令所花费的时间，单位是秒。请注意这个数字并不代表实际的 CPU 时间。

S 指令执行时在核心模式（kernel mode）所花费的时间，单位是秒。

U 指令执行时在使用者模式（user mode）所花费的时间，单位是秒。

P 执行指令时 CPU 的占用比例。其实这个数字就是核心模式加上使用者模式的 CPU 时间除以总时间。

Memory Resources

M 执行时所占用的实体记忆体的最大值。单位是 KB

t 执行时所占用的实体记忆体的平均值，单位是 KB

K 执行程序所占用的记忆体总量（stack+data+text）的平均大小，单位是 KB

D 执行程序的自有资料区（unshared data area）的平均大小，单位是 KB

p 执行程序的自有堆叠（unshared stack）的平均大小，单位是 KB

X 执行程序间共享内容（shared text）的平均值，单位是 KB

Z 系统记忆体页的大小，单位是 byte。对同一个系统来说这是个常数

IO Resources

F 此程序的主要记忆体页错误发生次数。所谓的主要记忆体页错误是指某一记忆体页已经置换到置换档（swap file）中，而且已经分配给其他程序。此时该页的内容必须从置换档里再读出来。

R 此程序的次要记忆体页错误发生次数。所谓的次要记忆体页错误是指某一记忆体页虽然已经置换到置换档中，但尚未分配给其他程序。此时该页的内容并未被破坏，不必从置换档里读出来

W 此程序被交换到置换档的次数

c 此程序被强迫中断（像是分配到的 CPU 时间耗尽）的次数

w 此程序自愿中断（像是在等待某一个 I/O 执行完毕，像是磁碟读取等等）的次数

I 此程序所输入的档案数

O 此程序所输出的档案数

r 此程序所收到的 Socket Message
s 此程序所送出的 Socket Message
k 此程序所收到的信号 (Signal)数量

Command Info

C 执行时的参数以及指令名称
x 指令的结束代码 (Exit Status)

-p or --portability

这个选项会自动把显示格式设定成为：

real %e

user %U

sys %S

这么做的目的是为了与 POSIX 规格相容。

-v or --verbose

这个选项会把所有程式中用到的资源通通列出来，不但如一般英文语句，还有说明。对不想花时间去熟习格式设定或是刚刚开始接触这个指令的人相当有用。

范例：

利用下面的指令

time -v ps -aux

我们可以获得执行 **ps -aux** 的结果和所花费的系统资源。如下面所列的资料：

USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND

root 1 0.0 0.4 1096 472 ? S Apr19 0:04 init

root 2 0.0 0.0 0 0 ? SW Apr19 0:00 [kflushd]

root 3 0.0 0.0 0 0 ? SW Apr19 0:00 [kpiod]

.....

root 24269 0.0 1.0 2692 996 pts/3 R 12:16 0:00 ps -aux

Command being timed: "ps -aux"

User time (seconds): 0.05

System time (seconds): 0.06

Percent of CPU this job got: 68%

Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.16

Average shared text size (kbytes): 0

Average unshared data size (kbytes): 0

Average stack size (kbytes): 0

Average total size (kbytes): 0

Maximum resident set size (kbytes): 0

Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 238
Minor (reclaiming a frame) page faults: 46
Voluntary context switches: 0
Involuntary context switches: 0
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0

名称 talk

使用权限：所有使用者

使用方式：

talk person [ttyname]

说明：与其他使用者对谈

把计：

person：预备对谈的使用者帐号，如果该使用者在其他机器上，则可输入 **person@machine.name**

ttyname：如果使用者同时有两个以上的 **tty** 连线，可以自行选择合适的 **tty** 传讯息

例子.1：

与现在机器上的使用者 **Rollaend** 对谈，此时 **Rollaend** 只有一个连线：

talk Rollaend

接下来就是等 **Rollaend** 回应，若 **Rollaend** 接受，则 **Rollaend** 输入 ``talk jzlee`` 即可开始对谈，结束请按 **ctrl+c**

例子.2：与 **linuxfab.cx** 上的使用者 **Rollaend** 对谈，使用 **pts/2** 来对谈：

talk Rollaend@linuxfab.cx pts/2

接下来就是等 **Rollaend** 回应，若 **Rollaend** 接受，则 **Rollaend** 输入 ``talk jzlee@jzlee.home`` 即可开始对谈，结束请按 **ctrl+c**

注意：若萤幕的字会出现不正常的字元，试着按 **ctrl+l** 更新萤幕画面。

名称 top

使用权限：所有使用者

使用方式：**top [-] [d delay] [q] [c] [S] [s] [i] [n] [b]**

说明：即时显示 **process** 的动态

把计

d：改变显示的更新速度，或是在交谈式指令列(**interactive command**)按 **s**

q：没有任何延迟的显示速度，如果使用者是有 **superuser** 的权限，则 **top** 将会以最高的优先序执行

c：切换显示模式，共有两种模式，一是只显示执行档的名称，另一种是显示完整的路径与名称 **S**：累积模式，会将已完成或消失的子行程(**dead child process**) 的 **CPU time** 累积起来

s：安全模式，将交谈式指令取消，避免潜在的危机

i：不显示任何闲置(**idle**) 或无用(**zombie**) 的行程

n：更新的次数，完成后将会退出 **top**

b：批次档模式，搭配 **"n"** 参数一起使用，可以用来将 **top** 的结果输出到档案内

范例：

显示更新十次后退出；

```
top -n 10
```

使用者将不能利用交谈式指令来对行程下命令：

```
top -s
```

将更新显示二次的结果输入到名称为 **top.log** 的档案里：

```
top -n 2 -b < top.log
```

名称 tr

1.比方说要把目录下所有的大写档名换为小写档名？

似乎有很多方式，**"tr"**是其中一种：

```
#!/bin/sh
```

```
dir="/tmp/testdir";
```

```
files=`find $dir -type f`;
```

```
for i in $files
```

```
do
dir_name=`dirname $i` ;
ori_filename=`basename $i`
new_filename=`echo $ori_filename | tr [:upper:] [:lower:]` > /dev/null;
#echo $new_filename;
mv $dir_name/$ori_filename $dir_name/$new_filename
done
```

2.自己试验中...lowercase to uppercase

```
tr abcdef...[del] ABCDE...[del]
tr a-z A-Z
tr [:lower:] [:upper:]
```

```
shell>> echo "this is a test" | tr a-z A-Z > www
shell>> cat www
THIS IS A TEST
```

3.去掉不想要的字串

```
shell>> tr -d this ### 去掉有关 t.e.s.t
this
```

```
man
man
test
e
```

4.取代字串

```
shell>> tr -s "this" "TEST"
this
TEST
th
TE
```

名称 true

用途

返回零（true）或非零（false）的退出值。

语法

true

false

描述

`true` 命令返回零退出值。`false` 命令返回非零退出值。这些命令常用作 `shell` 脚本的一部分。

示例

要构造每秒显示一次日期和时间的循环，请在 `shell` 脚本中使用以下代码：

```
while true
do
    date
    sleep 60
done
```

21) U 类指令

名称： `uptime`

使用权限： 所有使用者

使用方式： `uptime [-V]`

说明： `uptime` 提供使用者下面的资讯，不需其他参数：

现在的时间

系统开机运转到现在经过的时间

连线的使用者数量

最近一分钟，五分钟和十五分钟的系统负载

参数： `-V` 显示版本资讯。

范例： `uptime`

其结果为：

10:41am up 5 days, 10 min, 1 users, load average: 0.00, 0.00, 1.99

名称： `uname`

显示当前操作系统名称。

语法

```
uname [ -a | -x | -S Name ] [ -F ] [ -f ] [ -l ] [ -L ] [ -m ] [ -M ] [ -n ] [ -p ] [ -r ] [ -s ] [ -T Name ] [ -u ] [ -v ]
```

描述

`uname` 命令将正在使用的操作系统名写到标准输出中。

机器 ID 号码包括以下数字格式的 12 个字符 `xyyyyyyyymmss`。`xx` 位置指示系统且始终为 `00`。`yyyyyy` 位置包含整个系统的唯一标识号。`mm` 位置代表型号标识。`ss` 位置为子型号并且始终为 `00`。型号标识描述 CPU 平板标识，而不是描述整个系统的型号。

有时可使用 `uname -m` 命令决定使用什么型号。以下列表并非全面。参考硬件供应商提供的文档以获取 `E0-FF` 范围内的值。并且注意并非所有的机器类型都有机器 ID。许多新机器都共享 `4C` 的公共机器 ID。系统型号十六进制代码（`mm`）为：

当安装了新的操作系统软件级别时，`uname` 命令返回的机器标识符值可能改变。这一改变影响使

用该值访问许可程序的应用程序。需要查看标识符，输入 `uname -m` 命令。

如果应用程序受到影响，联系合适的支持组织。

标志

- a 显示 `-m`、`-n`、`-r`、`-s` 和 `-v` 标志指定的所有信息。不能与 `-x` 或 `-SName` 标志连用。如果 `-x` 标志和 `-a` 标志一起指定，`-x` 标志会覆盖它。
- F 显示由十六进制字符构成的系统标识字符串。此标识字符串对特定系统上的所有分区都是相同的。
- f 除分区号还用于此字符串的计算之外，与 `F` 标志类似。产生的标识字符串对特定系统上的每个分区都是唯一的。
- l 显示 LAN 网络号码。
- L 显示 LPAR 号码和 LPAR 名称。如果 LPAR 不存在，“-1”显示为 LPAR 号码，NULL 为 LPAR 名称
- m 显示硬件运行系统的机器 ID 号。
注：
-m 标志不能为 LPAR 环境中的分区生成唯一的机器标识。
- M 显示系统型号名称。如果型号名称属性不存在，显示空字符串
- n 显示节点名称。可能是用以标识系统且为 UUCP 通信网络所知的名称。
- p 显示系统处理器的体系结构。
- r 显示操作系统的发行版号。
- s 显示系统名。标志缺省为开。
- S Name 设置节点名。可以是系统 UUCP 通信网络名。
- T Name 设置系统名。可以是系统 UUCP 通信网络名。
- u 显示系统 ID 号码。如果这一属性未被定义，输出与 `uname -m` 显示的输出一致。
- v 显示操作系统版本。
- x 显示 [-a](#) 标志指定的信息和 `-l` 标志指定的 LAN 网络号。

如果输入标志无效，`uname` 命令退出，生成错误消息、错误返回状态，并无输出。

注意：经过系统重新引导后，“`uname`”命令不保存新系统名和节点名称值。

名称 unalias

用途

除去别名定义。

语法

`unalias` [-a](#)

`unalias` AliasName ...

描述

`unalias` 命令除去每个别名的指定的定义，或除去全部别名定义（如果使用了 `-a` 标志）。别名定义从当前 `shell` 环境中除去。

由于 `unalias` 命令影响当前的 `shell` 执行环境，所以它将作为 Korn shell 或 POSIX shell 常规内置命令提供。

标志

-a 从当前 shell 环境中除去全部别名定义。

22) V 类指令

23) W 类指令

名称 who

使用权线：所有使用者都可使用

使用方式：who - [husfV] [user]

说明：显示系统中有那些使用者正在上面，显示的资料包含了使用者 ID，使用的终端机，从那边连上来的，上线时间，呆滞时间，CPU 使用量，动作等等。

把计：

- h：不要显示标题列
- u：不要显示使用者的动作/工作
- s：使用简短的格式来显示
- f：不要显示使用者的上线位置
- V：显示程式版本

名称 wall

使用权限：所有使用者

使用方式：

wall [message]

使用说明：

wall 会将讯息传给每一个 mesg 设定为 yes 的上线使用者。当使用终端机介面做为标准传入时，讯息结束时需加上 EOF (通常用 Ctrl+D)

例子：

传讯息"hi" 给每一个使用者：

wall hi

名称 write

使用权限：所有使用者

使用方式：

`write user [ttyname]`

说明：传讯息给其他使用者

把计：

user：预备传讯息的使用者帐号

ttyname：如果使用者同时有两个以上的 **tty** 连线，可以自行选择合适的 **tty** 传讯息

例子.1：

传讯息给 **Rollaend**，此时 **Rollaend** 只有一个连线：

`write Rollaend`

接下来就是将讯息打上去，结束请按 **ctrl+c**

例子.2：传讯息给 **Rollaend**，**Rollaend** 的连线有 **pts/2**，**pts/3**：

`write Rollaend pts/2`

接下来就是将讯息打上去，结束请按 **ctrl+c**

注意：若对方设定 **mesg n**，则此时讯席将无法传给对方

24) X 类指令

25) Y 类指令

26) Z 类指令