

cut截取命令

语法: cut 参数 文件或输入

参数:

```
1 -d :后面接分隔字符, 根据分隔字符将一段信息划分成为几段
2
3 -f :取出第几段
4
5 -c :以字符为单位取出固定字符区间
```

```
1 例子1: 单个字符切割 cut的默认定界符是空格, 但有些文件的定界符不
   是空格, 所以用-d参数
2 [root@exercise1 ~]# tail -1 /etc/passwd
3 chrony:x:998:996::/var/lib/chrony:/sbin/nologin
4 [root@exercise1 ~]# tail -1 /etc/passwd | cut -d ":" -f
   3
5 998
6
7 或
8
9 [root@exercise1 ~]# tail -1 /etc/passwd | cut -d ":" -
   f3
10 998
11 [root@exercise1 ~]#
12
13 [root@exercise1 ~]# tail -1 /etc/passwd | cut -d ":" -
   f3,6 #只列出想要的第3列与第6列
14 998:/var/lib/chrony
15 [root@exercise1 ~]#
```

例子2: 获取第12个字符以后的所有字符

```
[root@exercise1 ~]# tail -1 /etc/passwd | cut -c 12-
8:996::/var/lib/chrony:/sbin/nologin
[root@exercise1 ~]#
```

例子3: 获取第12-19个字符的内容

```
[root@exercise1 ~]# tail -1 /etc/passwd | cut -c 12-19
```

```
8:996::/
```

```
[root@exercise1 ~]#
```

例子4: 获取ip地址

```
[root@exercise1 ~]# ifconfig ens33 | grep netmask | cut -d " " -f 10
```

#根据空格分割

```
192.168.119.142
```

不足之处:

1. 不够智能切割选择, 切割后需要自己数在第几个区域
2. 用-c则会以字符为单位, 输出正常; 而-b只会傻傻的以字节 (8位二进制位) 来计算, 输出就是乱码。当遇到多字节字符时, 可以使用-n选项, -n用于告诉cut不要将多字节字符拆开。
3. 如果文件里面的某些分隔区间是由若干个空格来间隔的, 那么用cut就有点麻烦了, 因为cut只擅长处理“以一个字符间隔”的文本内容

sort排序命令

语法: sort 参数 文件或输入

选项:

- | | | |
|---|----|----------------------------|
| 1 | -f | : 忽略大小写差异 |
| 2 | -n | : 使用纯数字进行排序 |
| 3 | -r | : 反向排序 |
| 4 | -u | : uniq, 相同的数据中, 仅出现一行 (去重) |
| 5 | -t | : 分隔符号, 默认是用[tab]键来分隔 |
| 6 | -k | : 以哪个区间来排序 |

```
1 例子1:
2
3 [root@exercise1 opt]# cat >> /opt/c.txt << eof    #先创建
   一个文件用于测试
4 a:1
5 995:abc
6 99:abc
7 b:2
8 cd:563
9 c:3
10 d:102
11 e:210
12 eof
```

```
[root@exercise1 opt]# ls
1 1f      1.txt.xz 4 a.zip    back.tar etc      etc.tar.gz grub.tar j
messages
1d 1.txt  2      5 back    boot    etc.tar  etc.tar.xz i.bak  log
test
1e 1.txt.bz2 3      abc back2.tar c.txt  etc.tar.bz2 grub2.tar id
log.bak
```

[root@exercise1 opt]# cat c.txt | sort -t ":" -k2 #以:为分隔符, 会发现以冒号的数字进行排序并且以左边第一

#个数字进行排序, 然后再按照左边同数字的情况下按大小排序, 看截图

```
a:1
d:102
b:2
e:210
c:3
cd:563
ab:99
```

或

```
[root@exercise1 opt]# cat c.txt | sort -t ":" -k 2
a:1
d:102
```

b:2
e:210
c:3
cd:563
ab:99
[root@exercise1 opt]#

```
[root@exercise1 opt]# cat c.txt
a:1
ab:99
b:2
cd:563
c:3
d:102
e:210
ff:209
gg:20
[root@exercise1 opt]# cat c.txt | sort -t ":" -k 2
a:1
d:102
b:2
gg:20
ff:209
e:210
c:3
cd:563
ab:99
```

```
1 例子2：反向排序
2
3 [root@exercise1 ~]# cat /opt/c.txt | sort -r -t ":" -k2
4 ab:99
5 cd:563
6 c:3
7 e:210
8 ff:209
9 gg:20
10 b:2
11 d:102
12 a:1
```

```
1 例子3: 纯数字排序
2 [root@exercise1 ~]# cat /opt/c.txt | sort -n -t ":" -k2
3 a:1
4 b:2
5 c:3
6 gg:20
7 ab:99
8 d:102
9 ff:209
10 e:210
11 cd:563
12
13
```

```
1 例4: 以字母比较, 要字母后的内容完整一致; 以数字比较, 数字一致即可
2 [root@home opt]# cat c.txt | sort -t: -k2 -u
3 a:1
4 d:102
5 b:2
6 e:210
7 c:3
8 cd:563
9 995:abc
```

uniq去重命令

语法: `uniq 参数 文件或输入`

选项:

```
1 -i    :忽略大小写
2
3 -c    :进行计数 (常用)
4
5 -d    :只显示重复的行
```

```
1 例子1: 查找谁登陆过系统,且统计登陆次数
2
3 [root@exercise1 ~]# last | cut -d " " -f1 | sort | uniq
   -c    #查出记录并且按空格作为分隔符然后排序再去重
4
5 reboot
6 root
7 wtmp
8 [root@exercise1 ~]#
```

```
1 备注:
2 备注一】关于last命令的几点说明:
3 1. wtmp,btmp,utmp均为二进制文件,不能用cat查看,可用last打开
4 2. echo > /var/log/wtmp 可清空wtmp记录
5 【备注二】Linux系统的三个主要日志子系统:
6 1. 进程日志(acct/pacct: 记录用户命令)
7 2. 错误日志(/var/log/messages:系统级信息; access-log:记录
   HTTP/WEB的信息)
8 3. 连接日志(/var/log/wtmp,/var/log/btmp,/var/run/utmp)
9 >>>有关当前登录用户的信息记录在文件utmp中;
10 >>>登录进入和退出纪录在文件wtmp中;
11 >>>最后一次登录文件可以用lastlog命令察看;
12 >>>数据交换、关机和重起也记录在wtmp文件中;
```

例子2: 查找谁登陆过系统统计登录次数

```
[root@exercise1 ~]# last | cut -d " " -f1 | sort | uniq -c
1
9 reboot
21 root
1 wtmp
```

例子3: 查找谁重复登陆过系统

```
[root@exercise1 ~]# last | cut -d " " -f1 | sort | uniq -d
reboot
```

```
root
[root@exercise1 ~]#
```

wc统计命令

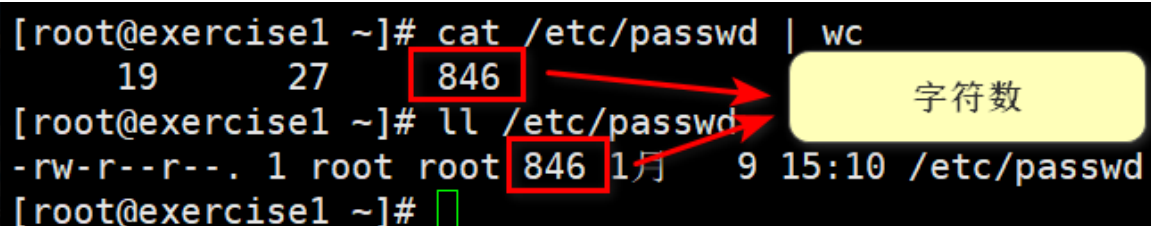
语法: wc 参数 文件或输入

选项:

```
1 -l      : 仅列出行 (常用)
2
3 -L      : 获取文件中最长一行的 (可用来限制密码)
4
5 -w      : 仅列出多少字
6
7 -m      : 多少字符
```

例子1: 把/etc/passwd的行数、字数、字符数都显示

```
[root@exercise1 ~]# cat /etc/passwd | wc
  19   27  846          #行   字数   字符数
[root@exercise1 ~]#
```



```
[root@exercise1 ~]# cat /etc/passwd | wc
  19   27  846
[root@exercise1 ~]# ll /etc/passwd
-rw-r--r--. 1 root root 846 1月  9 15:10 /etc/passwd
[root@exercise1 ~]#
```

```
1 [root@exercise1 ~]# cat /etc/passwd | wc -l      #行数
2 19
3 [root@exercise1 ~]# cat /etc/passwd | wc -w      #多少字
4 27
5 [root@exercise1 ~]# cat /etc/passwd | wc -m      #多少字符
6 846
7
8 统计变量长度:
9 [root@exercise1 ~]# name="I am headsome boy"
10 [root@exercise1 ~]# echo $name
11 I am headsome boy
```

```
12 方法1:
13 [root@exercise1 opt]# echo $name|wc -L
14 17
15 方法2:
16 [root@exercise1 opt]# echo ${#name}
17 17
18 方法3:
19 [root@exercise1 opt]# expr length "$name"
20 1
```

```
1 实战：如何以一串命令取得登录人数的总人次？
2 注：“ \ ”使用
3 [root@exercise1 ~]# last | cut -d " " -f1 | sort | grep
  -v "^$\\|wtmp\\|reboot" | wc -l
4 21
```

tr命令

作用：可以用来删除一段信息当中的文字，或是进行文字信息的替换(任何参数都不加)

语法：tr 参数 文件或输入

```
1 tr [OPTION]... SET1 SET2
2
3 字符集1(SET1)：指定要转换或删除的原字符集。当执行转换操作时，必须使用参数“字符集2”指定转换的目标字符集。但执行删除操作时，不需要参数“字符集2”；
4 字符集2(SET2)：指定要转换成的目标字符集。
```

-d :删除文字信息

-s :替换掉重复的字符

-c :反选设定字符。也就是符合 SET1 的部份不做处理，不符合的剩余部份才进行转换

```
1 例子1: 生成随机字符
2
3 [root@exercise1 ~]# cat /dev/urandom |tr -dc [:a1num:]
   |head -c 8
4 ddTRYFKn
```

例子2: 能删除指定字符外的补集 -c

[root@exercise1 ~]# echo "hello 123 world " | tr -d -c '0-9 \n' #除数字、空格字符和换行符之外的所有字符都会被删除

123

[root@exercise1 ~]#

例子3: 将/opt/c.txt输出的信息中, 将": "删除

[root@exercise1 ~]# cat /opt/c.txt | tr -d ":",

a1

ab99

b2

cd563

c3

d102

e210

ff209

gg20

[root@exercise1 ~]#

例子4: 将last输出的信息中, 所有的小写变成大写字符

[root@exercise1 ~]# last | tr "[a-z]" "[A-Z]"

ROOT PTS/0 192.168.119.1 THU JAN 13 18:33 STILL

LOGGED IN

ROOT PTS/1 192.168.119.1 THU JAN 13 12:47 - 16:05 (03:18)

ROOT PTS/0 192.168.119.1 THU JAN 13 08:20 - 14:49 (06:29)

REBOOT SYSTEM BOOT 3.10.0-693.EL7.X THU JAN 13 08:18 - 19:58

(11:39)

ROOT PTS/0 192.168.119.1 WED JAN 12 23:42 - DOWN

(00:46)

.....

WTMP BEGINS SUN JAN 9 09:43:20 2022

[root@exercise1 ~]#

例子5：将文件中重复的数字替换掉

[root@exercise1 ~]# cat /opt/c.txt | tr -s "[0-9]"

a:1

ab:9

b:2

cd:563

c:3

d:102

e:210

ff:209

gg:20

实战：将在window系统里创建的脚本放在linux里运行？

```
1 [root@exercise1 opt]# ./f.sh
2 -bash: ./f.sh: /bin/bash^M: 坏的解释器：没有那个文件或目录
3 [root@exercise1 opt]# cat f.sh |tr -d '\r' >test1.sh
4 #\r: 脚本文件在windows下编辑过，windows下每一行的结尾是\n\r，而
   在linux下文件的结尾是，
5
6 那么你在windows下编辑过的文件在linux下打开看的时候每一行的结尾就
   会多出来一个字符\r
7 [root@exercise1 opt]# chmod +x test1.sh
8 [root@exercise1 opt]# ./test1.sh
9 this is a good day
```

