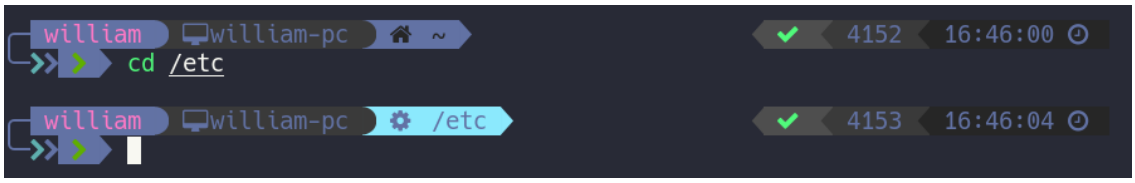
Linux命令

1. cd

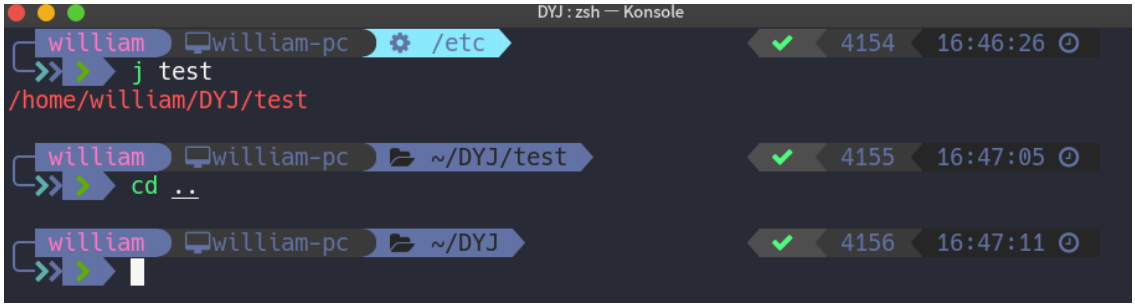
cd（change directory）

1. 进入家目录（cd ~），直接cd也行

2. 进入系统/etc目录（cd /etc）

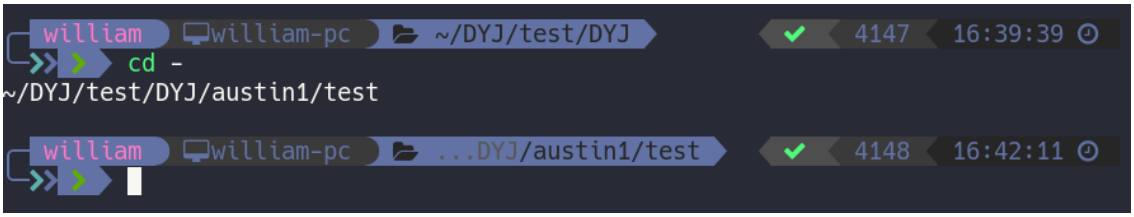


3. 切换到当前目录的上一级目录（cd ..）



4. 进入当前目录的父目录的父目录（cd ../../）

5. 返回当前用户上一次所在的目录（cd -）

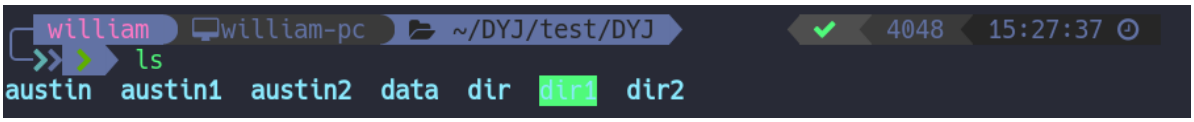


2. pwd

`pwd` （print working directory）显示当前工作目录的绝对路径

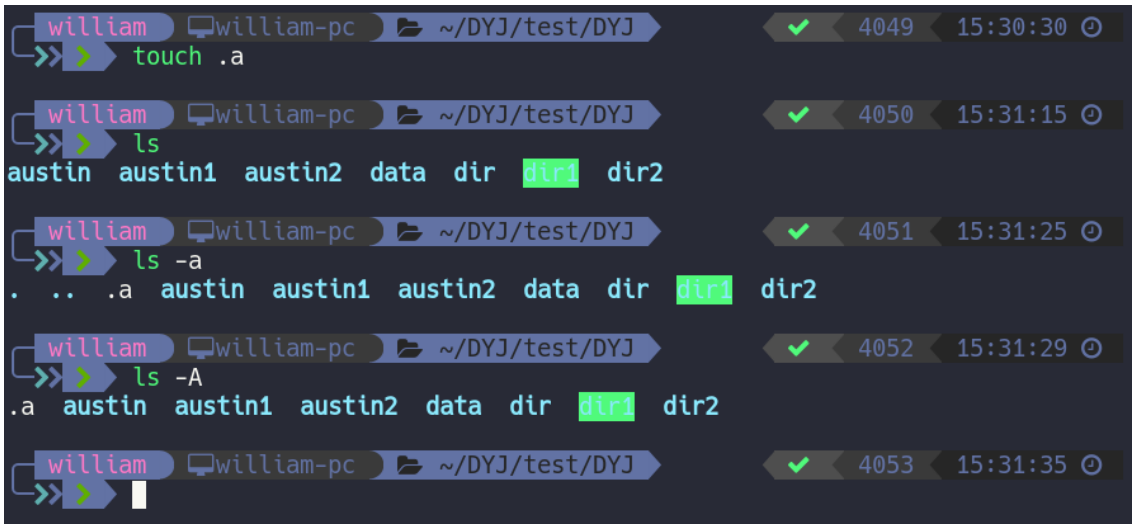
3. ls

ls （list directory contents）列出目录的内容及其内容属性信息

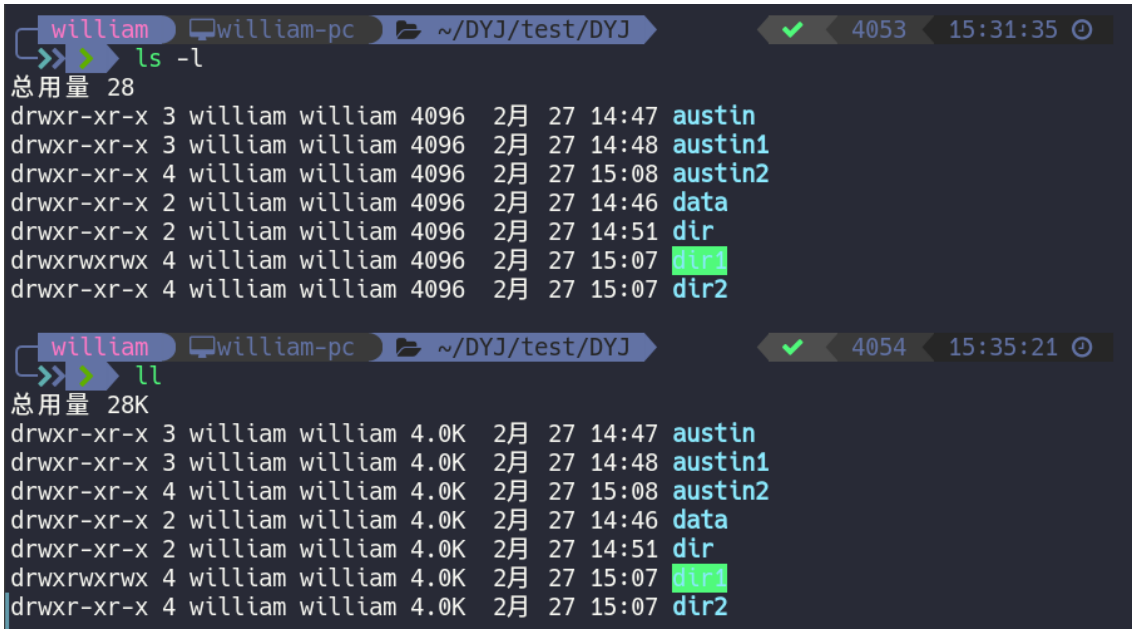



```
ls [-latdhurR] [file]
```

1. ls -a 显示所有文件，ls -A 不显示..和.目录



2. `ls -l` 以长格式列出文件及目录信息，缩写`ll`



`drwxr-xr-x 3 william william 4096  2月 27 14:47 austin`

- 1位：字符文件类型，“d”目录，“-”普通文件，“l”链接文件，“c”设备文件，“p”命令管道文件，“s”sock文件，与shell编程有关文件
- 2-4位：r表是读 (Read) 4、w表示写 (Write) 2、x表示执行 (eXecute) 1
- 5-7位：改文件/目录所属用户组的权限
- 8-10位：其他用户权限
- *3* 硬链接个数。新建一个空目录，这个目录的第二字段就是2，表示该目录下有两个子目录。**每一个目录都有一个指向它本身的子目录**。
- *william*文件的拥有者
- *william*文件拥有者所在组
- *4.0K*文件所占用的空间。**请注意是文件夹本身的大小，而不是文件夹以及它下面的文件的总大小**
- *2月 27 14:47*文件最近访问（修改）的时间
- *austin*文件名

3. `ls -d` 只显示目录本身的信息，与l连用

```
william  william-pc  ~/DYJ/test/DYJ
» ls -d dir
dir

william  william-pc  ~/DYJ/test/DYJ                    4061  15:56:10
» ls -ld dir
drwxr-xr-x 2 william william 4096  2月  27 14:51 dir
```

4. `ls -R`递归查看目录，感觉没tree好用

```
william  william-pc  ~/DYJ/test/DYJ                    4062  15:56:30
» ls
austin  austin1  austin2  data  dir  dir1  dir2

william  william-pc  ~/DYJ/test/DYJ                    4063  15:58:19
» ls -R dir2
dir2:
dir3  dir4

dir2/dir3:

dir2/dir4:

william  william-pc  ~/DYJ/test/DYJ                    4064  15:58:24
» tree dir2
dir2
├── dir3
└── dir4
```

5. `ls -t`根据最后修改时间排序

```
william  william-pc  ~/DYJ/test/DYJ                    4066  16:02:03
» ll
总用量 28K
drwxr-xr-x 3 william william 4.0K  2月  27 14:47 austin
drwxr-xr-x 3 william william 4.0K  2月  27 14:48 austin1
drwxr-xr-x 4 william william 4.0K  2月  27 15:08 austin2
drwxr-xr-x 2 william william 4.0K  2月  27 14:46 data
drwxr-xr-x 2 william william 4.0K  2月  27 14:51 dir
drwxrwxrwx 4 william william 4.0K  2月  27 15:07 dir1
drwxr-xr-x 4 william william 4.0K  2月  27 15:07 dir2

william  william-pc  ~/DYJ/test/DYJ                    4067  16:02:05
» ll -t
总用量 28K
drwxr-xr-x 4 william william 4.0K  2月  27 15:08 austin2
drwxrwxrwx 4 william william 4.0K  2月  27 15:07 dir1
drwxr-xr-x 4 william william 4.0K  2月  27 15:07 dir2
drwxr-xr-x 2 william william 4.0K  2月  27 14:51 dir
drwxr-xr-x 3 william william 4.0K  2月  27 14:48 austin1
drwxr-xr-x 3 william william 4.0K  2月  27 14:47 austin
drwxr-xr-x 2 william william 4.0K  2月  27 14:46 data
```

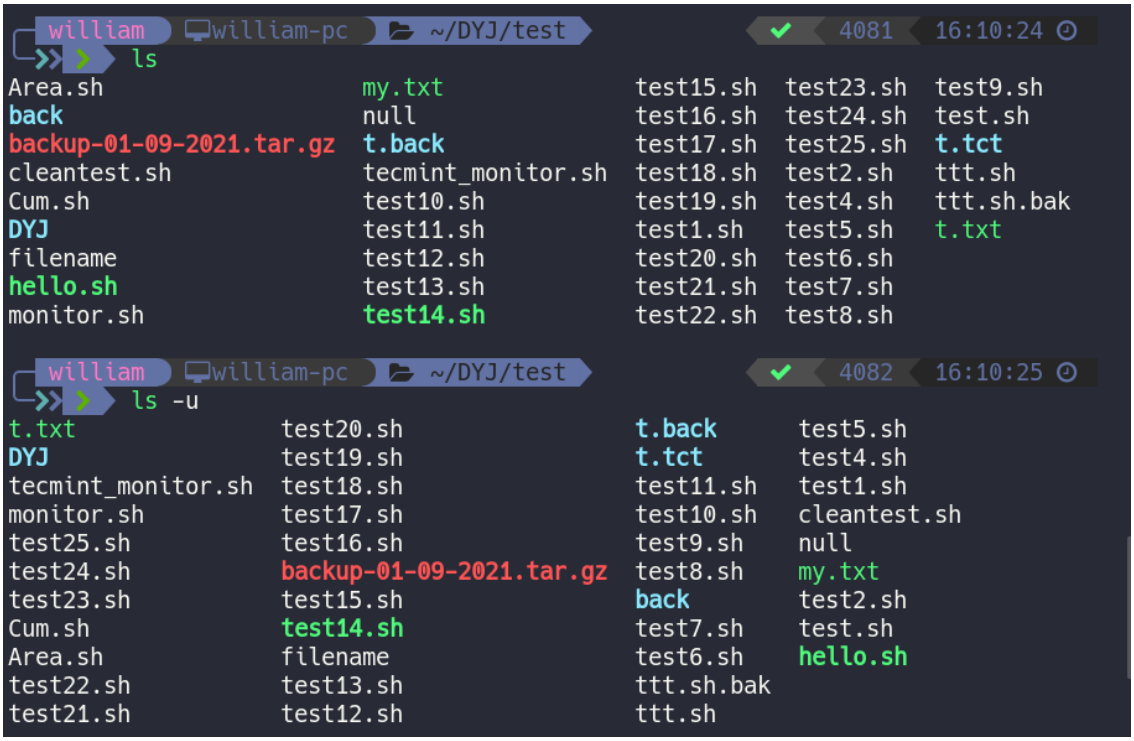
6. `ls -r`反向排序

```
william  william-pc  ~/DYJ/test/DYJ                    4067  16:02:05
>> > ll -t
总用量 28K
drwxr-xr-x 4 william william 4.0K  2月 27 15:08 austin2
drwxrwxrwx 4 william william 4.0K  2月 27 15:07 dir1
drwxr-xr-x 4 william william 4.0K  2月 27 15:07 dir2
drwxr-xr-x 2 william william 4.0K  2月 27 14:51 dir
drwxr-xr-x 3 william william 4.0K  2月 27 14:48 austin1
drwxr-xr-x 3 william william 4.0K  2月 27 14:47 austin
drwxr-xr-x 2 william william 4.0K  2月 27 14:46 data

william  william-pc  ~/DYJ/test/DYJ                    4068  16:02:16
>> > ll -rt
总用量 28K
drwxr-xr-x 2 william william 4.0K  2月 27 14:46 data
drwxr-xr-x 3 william william 4.0K  2月 27 14:47 austin
drwxr-xr-x 3 william william 4.0K  2月 27 14:48 austin1
drwxr-xr-x 2 william william 4.0K  2月 27 14:51 dir
drwxr-xr-x 4 william william 4.0K  2月 27 15:07 dir2
drwxrwxrwx 4 william william 4.0K  2月 27 15:07 dir1
drwxr-xr-x 4 william william 4.0K  2月 27 15:08 austin2
```

7. `ls -h` 以人类可读的信息展示文件大小

```
[william@william-pc blog]$ ls -l
总用量 2592
-rw-r--r--   1 william william    4096  2月   1 22:54 _config.yml
-rw-r--r--   1 william william 2114583  2月   5 13:30 db.json
drwxr-xr-x 764 william william   28672  1月  24 16:16 node_modules
-rw-r--r--   1 william william    1230  1月  24 16:16 package.json
-rw-r--r--   1 william william  479170  1月  24 16:16 package-lock.json
drwxr-xr-x  23 william william    4096  1月  16 17:25 public
-rw-r--r--   1 william william      14 12月  24 19:53 README.md
drwxr-xr-x   2 william william    4096 12月  24 17:30 scaffolds
drwxr-xr-x   9 william william    4096  1月  16 17:20 source
drwxr-xr-x   4 william william    4096  1月  26 13:13 themes
[william@william-pc blog]$ ls -lh
总用量 2.6M
-rw-r--r--   1 william william 4.0K  2月   1 22:54 _config.yml
-rw-r--r--   1 william william 2.1M  2月   5 13:30 db.json
drwxr-xr-x 764 william william  28K  1月  24 16:16 node_modules
-rw-r--r--   1 william william 1.3K  1月  24 16:16 package.json
-rw-r--r--   1 william william 468K  1月  24 16:16 package-lock.json
drwxr-xr-x  23 william william 4.0K  1月  16 17:25 public
-rw-r--r--   1 william william   14 12月  24 19:53 README.md
drwxr-xr-x   2 william william 4.0K 12月  24 17:30 scaffolds
drwxr-xr-x   9 william william 4.0K  1月  16 17:20 source
drwxr-xr-x   4 william william 4.0K  1月  26 13:13 themes
[william@william-pc blog]$
```

8. `ls -u` 最后访问时间排序

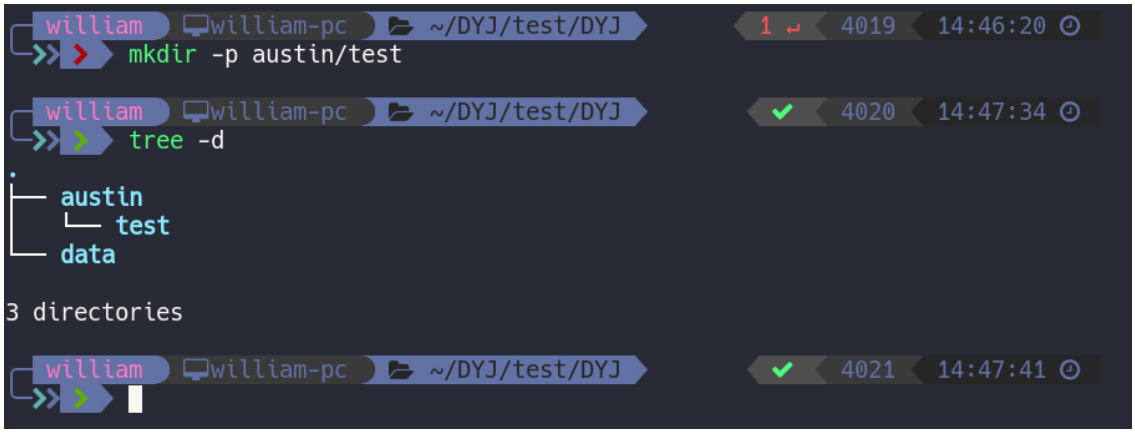


4. mkdir

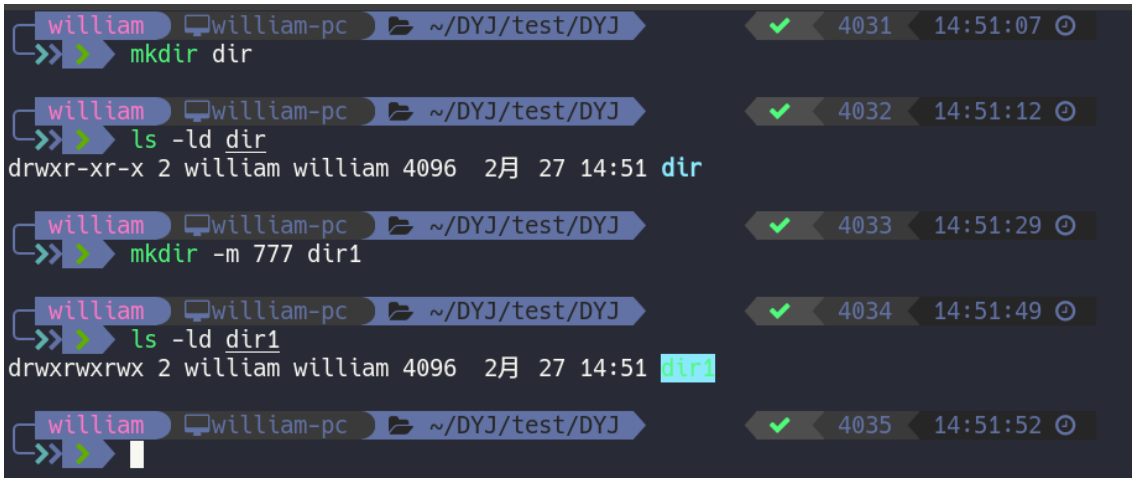
mkdir （make directories）创建目录，如果要创建的目录已存在，则会提示此文件已存在；而不会继续创建目录


```
mkdir [-pmv] directory
```

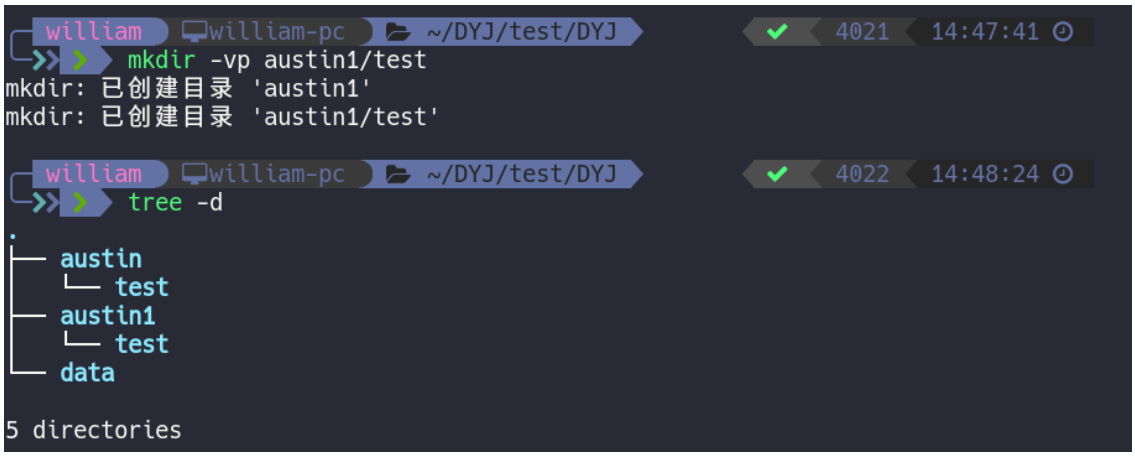
1. mkdir -p
    - 递归创建目录
    - 即使创建的目录已经存在也不会报错



2. mkdir -m
    - 设置新创建目录的默认目录对应的权限



3. mkdir -v
    - 显示创建目录的过程



4. 同时创建多个目录及多级子目录

```
[william@william-pc DYJ]$ mkdir -pv austin2/{dir1,dir2}/{dir3,dir4}
mkdir: 已创建目录 'austin2'
mkdir: 已创建目录 'austin2/dir1'
mkdir: 已创建目录 'austin2/dir1/dir3'
mkdir: 已创建目录 'austin2/dir1/dir4'
mkdir: 已创建目录 'austin2/dir2'
mkdir: 已创建目录 'austin2/dir2/dir3'
mkdir: 已创建目录 'austin2/dir2/dir4'
[william@william-pc DYJ]$ tree -d austin2
austin2
├── dir1
│   ├── dir3
│   └── dir4
└── dir2
    ├── dir3
    └── dir4

6 directories
[william@william-pc DYJ]$
```

5. rmdir

`rmdir （remove empty directories）删除空目录，当目录不为空，命令不起作用`

```
rmdir [-pv] directory
```

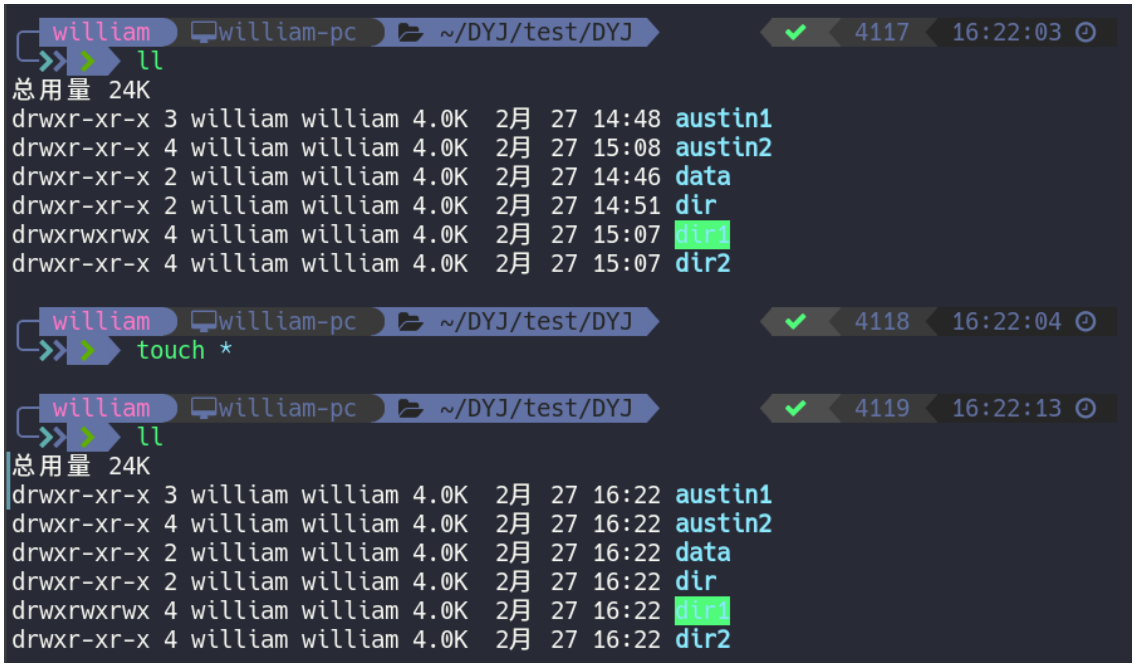
1. rmdir -p 递归删除目录，当子目录删除后空父目录为空，一并删除。
2. rmdir -v 显示执行过程

```
william  william-pc  ~/DYJ/test/DYJ   1 ↵  4114  16:20:48
» tree austin
austin
└── test

1 directory, 0 files

william  william-pc  ~/DYJ/test/DYJ   ✔  4115  16:21:04
» rmdir -p -v austin/test
rmdir: 正在删除目录，'austin/test'
rmdir: 正在删除目录，'austin'

william  william-pc  ~/DYJ/test/DYJ   ✔  4116  16:21:10
»
```

6. touch

```
touch [-acm] [-t time] file
```

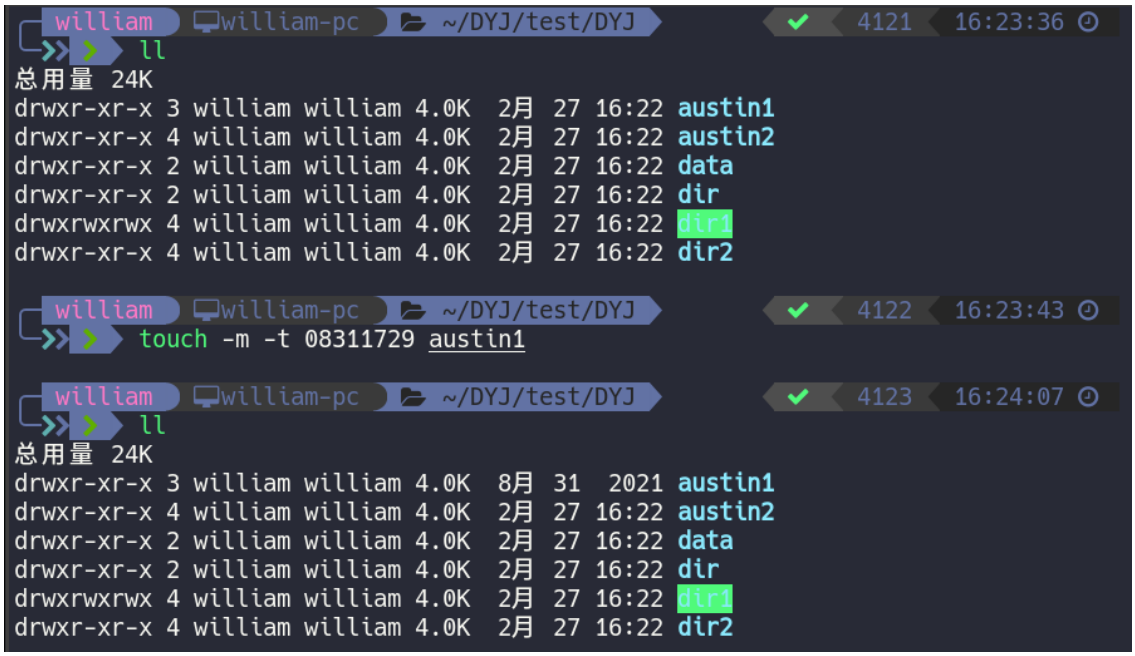
1. `touch *`



将目录中所有文件修改成相同时间和日期

2. `touch -m`

   只修改时间`touch -m -t 08311729 file1`(-t 具体时间日期)

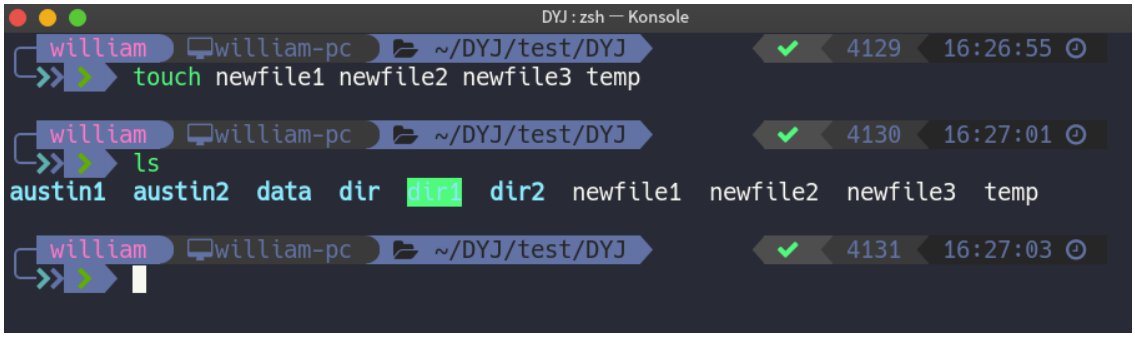


3. `touch -a`

   只修改访问时间`touch -a -t 200812211030 file2`

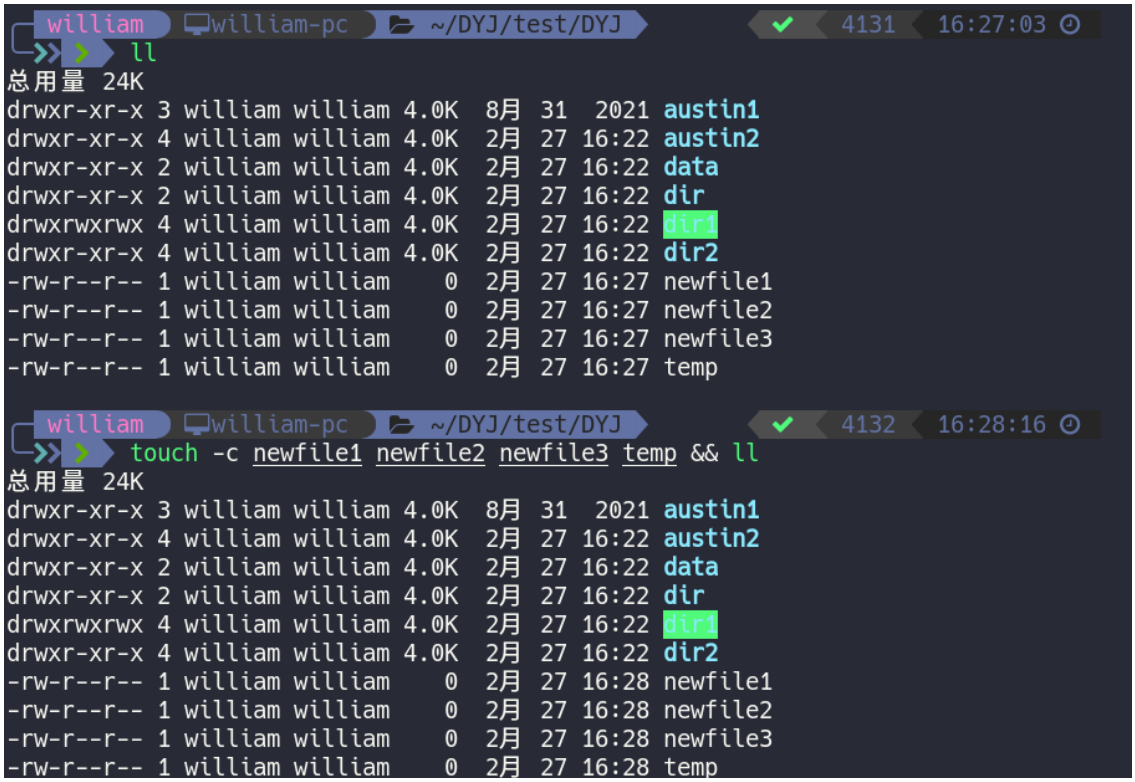
4. `touch newfile`

   创建多文件`touch newfile1 newfile2 newfile3 temp`



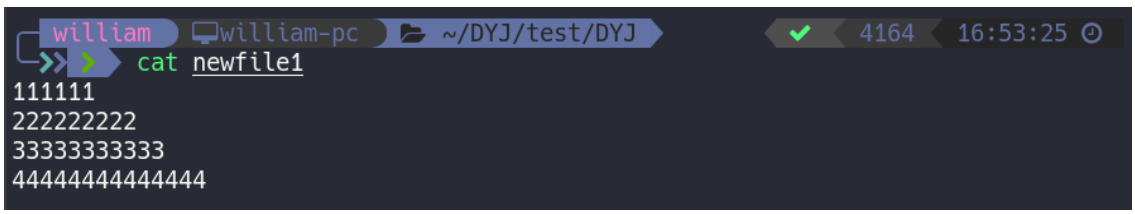
5. `touch -c(not create) file`

   更新访问时间不创建新文件`touch -c newfile1 newfile2 newfile3 temp`



7. cat

1. 查看文件



2. 把多个文件合并成一个

```
william william-pc ~/DYJ/test/DYJ 4166 16:53:54
>> cat newfile2
hello

william william-pc ~/DYJ/test/DYJ 4167 16:54:04
>> cat newfile1 newfile2 > newfile3

william william-pc ~/DYJ/test/DYJ 4168 16:55:30
>> cat newfile3
111111
222222222
33333333333
44444444444444

hello
```

3. 追加内容到文件尾

```
william william-pc ~/DYJ/test/DYJ 4169 16:55:37
>> cat >> newfile3 <<EOF
heredoc> 好累...
heredoc> EOF

william william-pc ~/DYJ/test/DYJ 4170 16:58:18
>> cat newfile3
111111
222222222
33333333333
44444444444444

hello
好累...
```

4. 文件内容清空

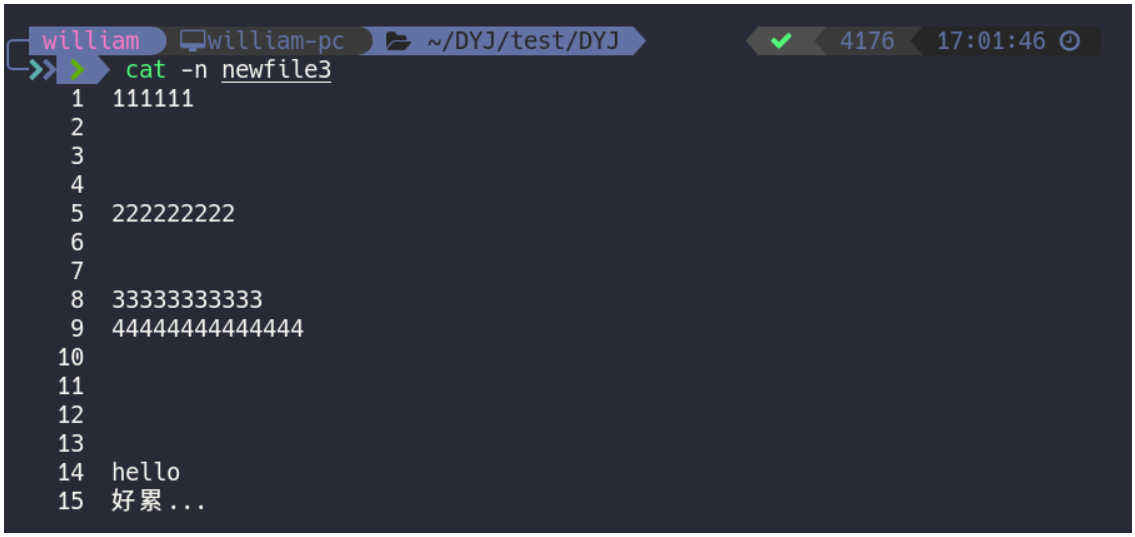
```
william william-pc ~/DYJ/test/DYJ 4184 17:08:15
>> touch null

william william-pc ~/DYJ/test/DYJ 4185 17:09:36
>> cat null > newfile3 && cat newfile3

william william-pc ~/DYJ/test/DYJ 4186 17:10:24
>>
```

5. 编辑新文件，旧文件覆盖哦

```
william william-pc ~/DYJ/test/DYJ 4176 17:01:46
>> cat -n newfile3
     1  111111
     2
     3
     4
     5  222222222
     6
     7
     8  33333333333
     9  44444444444444
    10
    11
    12
    13
    14  hello
    15  好累...
```

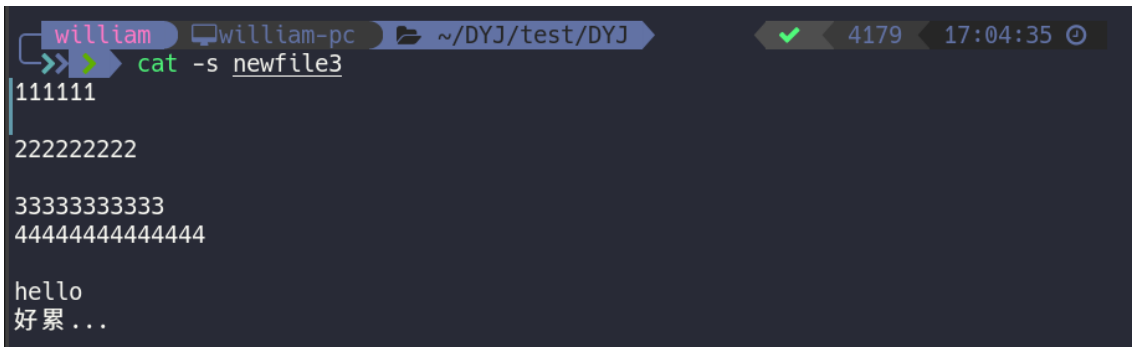
cat [-nbs] file

- `cat -n` 显示行号



- `cat -b` 不对空行编号

- `cat -s` 把两个以上的空行变成一个空行



8. cp

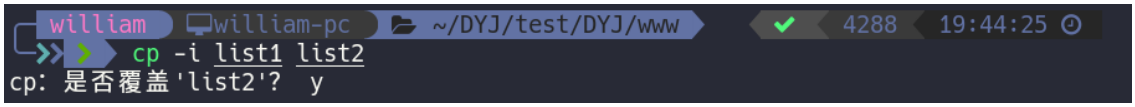
cp命令可以理解为英文单词copy的缩写，其功能为复制文件或目录

```
cp [-pdrai] [source] [dest]
```

- `cp -p` 复制文件时保持源文件的所有者、权限信息以及时间属性
- `cp -d`如果复制的源文件是符号文件，那么仅复制符号连接本身，而且保留符号链接所指向的目标文件或目录
- `cp -r` 递归复制目录，即复制目录下所有层级的子目录及文件

- `cp -a` 等同于`p d r`三条功能命令总和

- `cp -i` 覆盖已有文件前提示用户确认



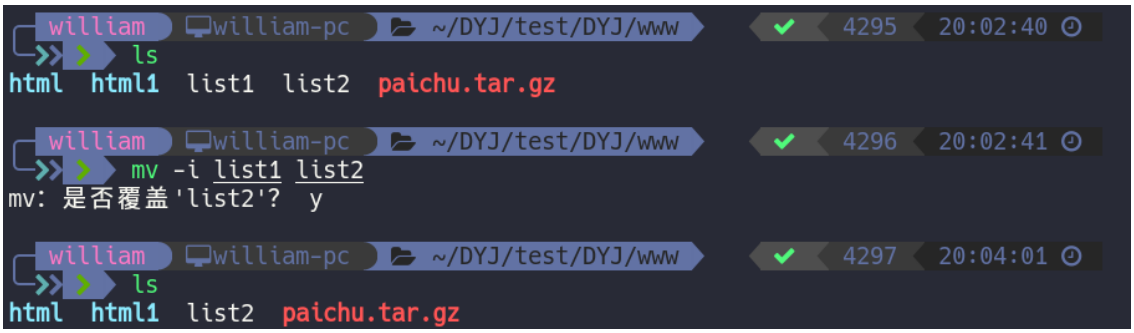
9. mv

mv命令可以理解为英文单词move的缩写，其功能是移动或重命名文件（move/rename files）

mv [-fin] [source] [dest]

- `mv -f` 若目标文件已经存在，则不会询问而是直接覆盖

- `mv -i` 若目标文件已经存在，则会询问是否覆盖



- `mv -n` 不覆盖已存在的文件

10. tar

在Linux系统里，tar是将多个文件打包在一起，并且可以实现解压打包的文件的命令。是系统管理员最常用的命令之一，tar命令不但可以实现对多个文件进行打包，还可以对多个文件打包后进行压缩。打包是指将一大堆文件或目录变成一个总的文件，压缩则是将一个大的文件通过一些压缩算法变成一个小文件。

tar [option] [file]

参数选项	解释说明（带※的为重点）
z	通过 gzip 压缩或解压 ※
c	创建新的 tar 包 ※
v	显示详细的 tar 命令执行过程 ※
f	指定压缩文件的名字 ※
t	不解压查看 tar 包的内容 ※
p	保持文件的原有属性
P（大写）	以绝对路径打包，危险参数
j	通过 bzip2 命令压缩或解压
x	解开 tar 包 ※
C	指定解压的目录路径 ※
--exclude=PATTERN	打包时排除不需要处理的文件或目录 ※
-X 文件名	从指定文件读取不需要处理的文件或目录列表
-N 日期	仅打包比指定日期新的文件，可用于增量打包备份
-h	打包软链接文件指向的真实源文件 ※
--hard-dereference	打包硬链接文件

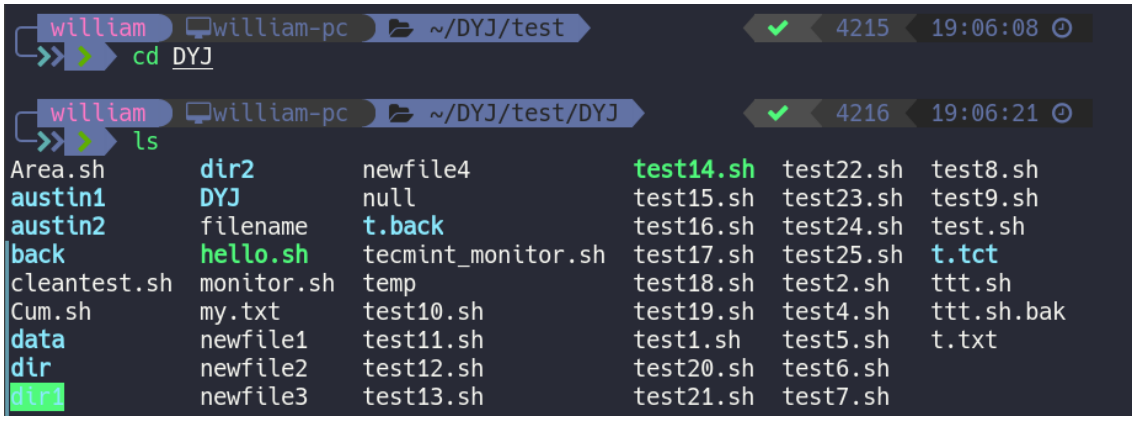
- `tar -zcvf` 打包文件

```
william  william-pc  ~/DYJ/test                    4208  18:59:57
>> > tar -zcvf backup.tar.gz ./*
./Area.sh
./back/
./back/test.sh
./cleantest.sh
./Cum.sh
./DYJ/
./DYJ/dir2/
./DYJ/dir2/dir3/
./DYJ/dir2/dir4/
./DYJ/austin1/
./DYJ/austin1/test/
./DYJ/data/
./DYJ/dir1/
./DYJ/dir1/dir3/
./DYJ/dir1/dir4/
```

- `tar -ztvf` 查看压缩包内的内容**(不解压)**，v显示文件属性

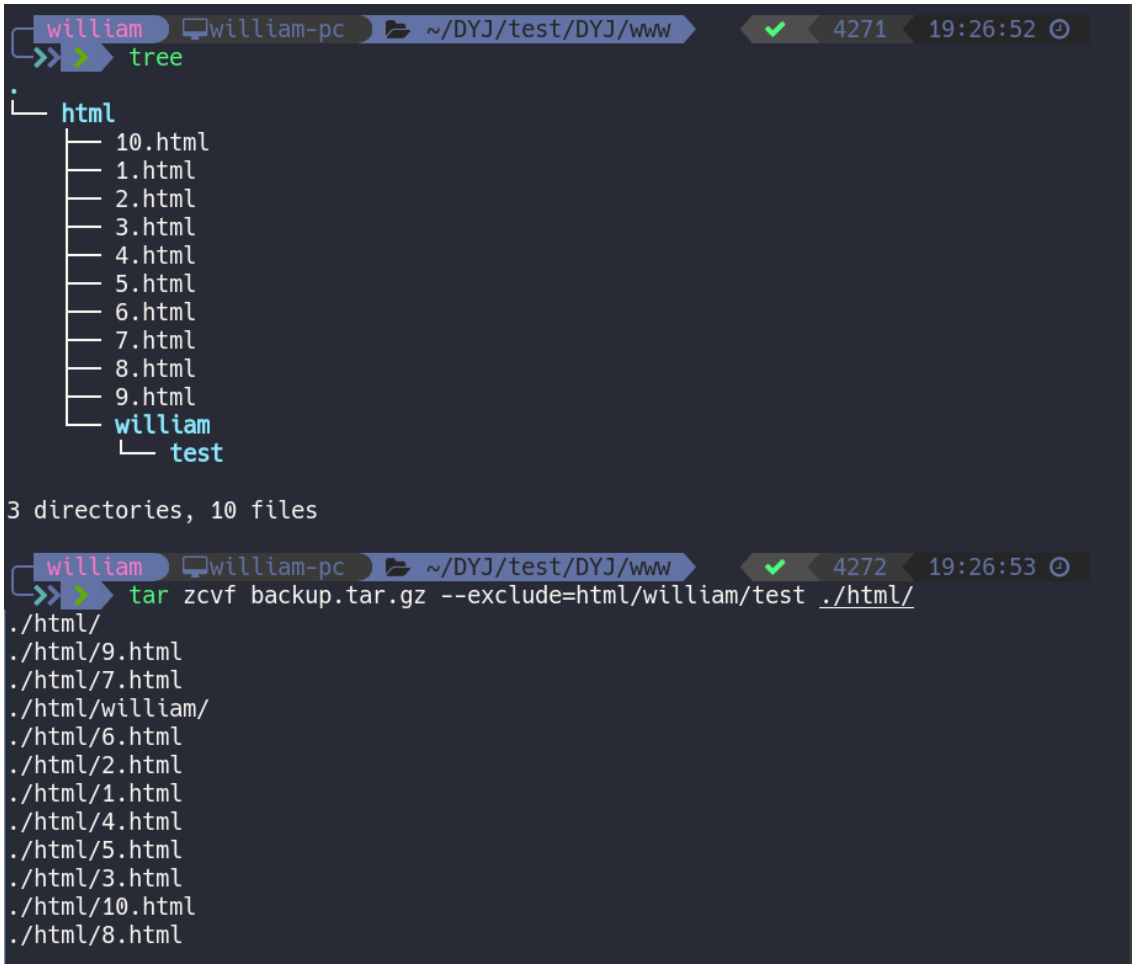
```
william  william-pc  ~/DYJ/test                    4211  19:02:03
>> > tar ztvf backup.tar.gz
-rw-r--r-- william/william  53 2021-01-10 16:08 ./Area.sh
drwxr-xr-x william/william   0 2021-01-09 22:47 ./back/
-rw-r--r-- william/william   0 2021-01-09 22:47 ./back/test.sh
-rw-r--r-- william/william  44 2021-01-09 20:50 ./cleantest.sh
-rw-r--r-- william/william  60 2021-01-10 16:10 ./Cum.sh
drwxr-xr-x william/william   0 2021-02-27 17:09 ./DYJ/
drwxr-xr-x william/william   0 2021-02-27 16:22 ./DYJ/dir2/
drwxr-xr-x william/william   0 2021-02-27 15:07 ./DYJ/dir2/dir3/
drwxr-xr-x william/william   0 2021-02-27 15:07 ./DYJ/dir2/dir4/
drwxr-xr-x william/william   0 2021-08-31 17:29 ./DYJ/austin1/
drwxr-xr-x william/william   0 2021-02-27 14:48 ./DYJ/austin1/test/
drwxr-xr-x william/william   0 2021-02-27 16:22 ./DYJ/data/
drwxrwxrwx william/william   0 2021-02-27 16:22 ./DYJ/dir1/
drwxr-xr-x william/william   0 2021-02-27 15:07 ./DYJ/dir1/dir3/
drwxr-xr-x william/william   0 2021-02-27 15:07 ./DYJ/dir1/dir4/
```

- `tar -zxvf filename.tar.gz -C path` 选项C指定解压路径，若不加C则解压到当前路径



- `tar zcvf filename.tar.gz path` 压缩路径下的文件

- `tar zcvf filename.tar.gz --exclude=PATTERN path` **PATTERN后不要加/** 排除打包



- `tar zcvfX filename.tar.gz excludefile path` 排除多个文件打包

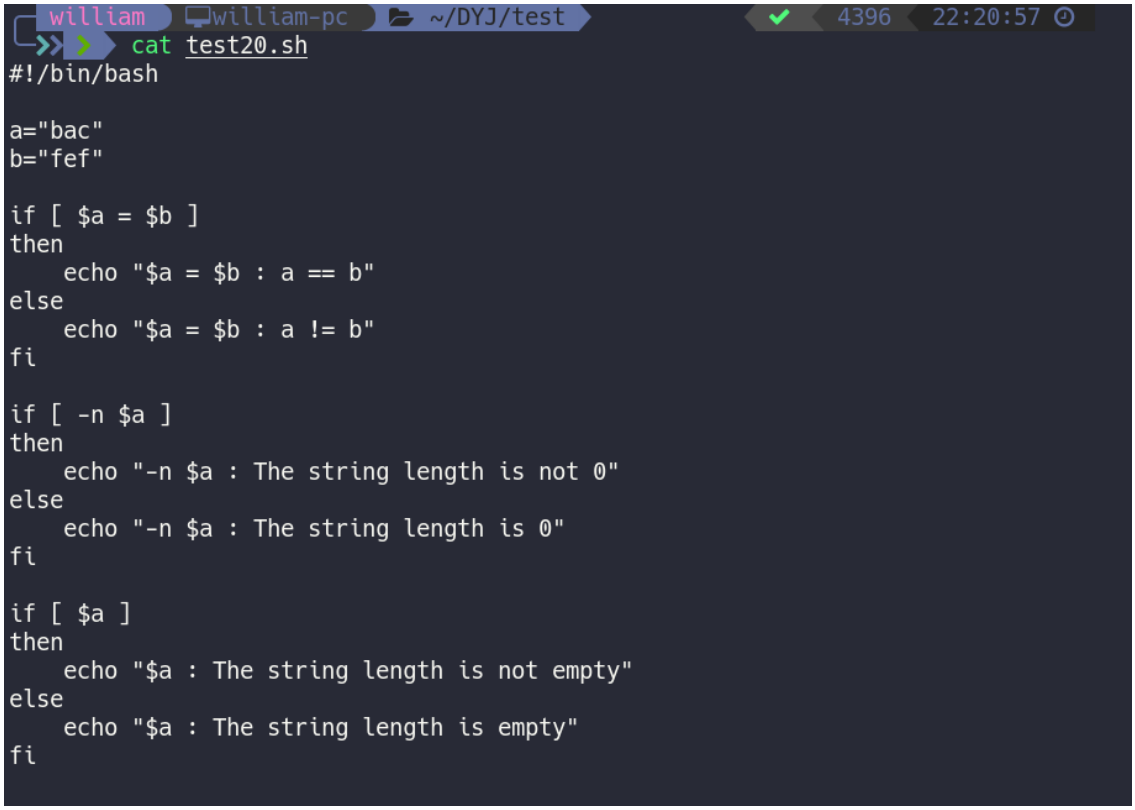
11. grep

grep命令是Linux系统中最重要的命令之一，其功能是从文本文件或管道数据流中筛选匹配的行及数据，如果再配和正则表达式的技术一起使用，则功能更加强大，它是Linux运维人员必须要掌握的命令之一！

grep [-vnicEwo] [PATTERN] file

参数选项	解释说明
-v	显示不匹配的行，或者说排除某些行，显示不包含匹配文本的所有行
-n	显示匹配行及行号
-i	不区分大小写（只适用于单字符），默认是区分大小写的
-c	只统计匹配的行数，注意不是匹配的次数
-E	使用扩展的 egrep 命令
--color=auto	为 grep 过滤的匹配字符串添加颜色
-w	只匹配过滤的单词
-o	只输出匹配的内容

- `grep -v [PATTERN] file`显示不匹配的行，显示不包含匹配文本的所有行



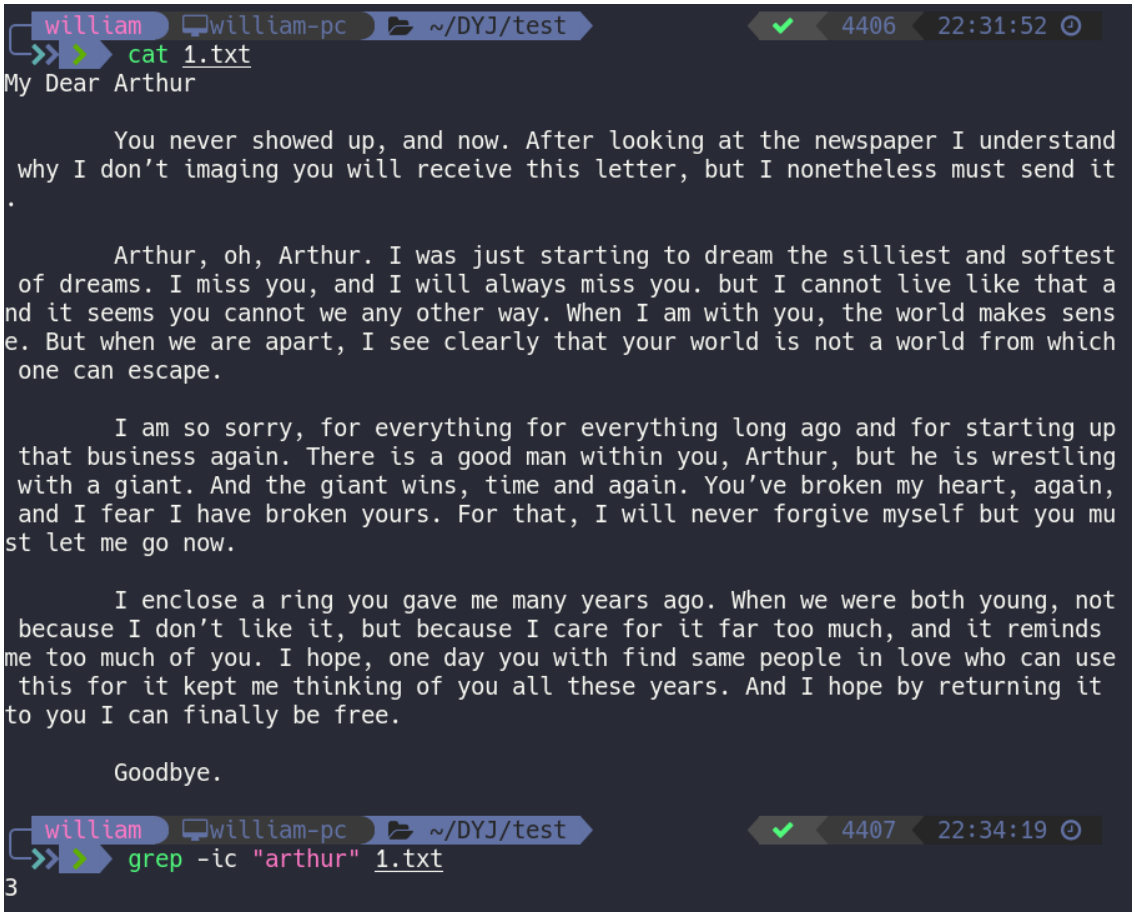
```
william  william-pc  ~/DYJ/test                    4397  22:21:02
>> > grep -v "string" test20.sh
#!/bin/bash

a="bac"
b="fef"

if [ $a = $b ]
then
    echo "$a = $b : a == b"
else
    echo "$a = $b : a != b"
fi

if [ -n $a ]
then
else
fi

if [ $a ]
then
else
fi
```

- `grep -n [PATTERN] file` 使用grep命令显示过滤后的内容的行号

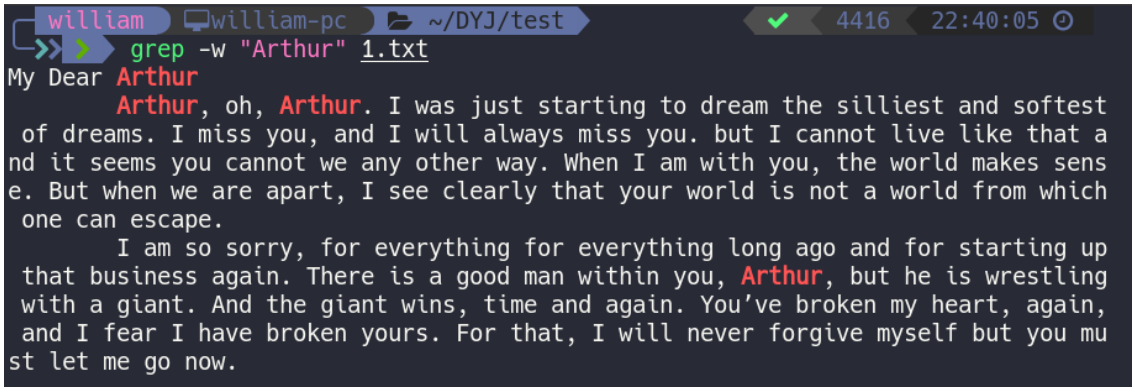
```
william  william-pc  ~/DYJ/test                    4400  22:27:08
>> > cat test5.sh
#!/bin/bash

echo hello; echo there

filename=ttt.sh

if [ -e "$filename" ]; then #if和then需要分隔，-e用于判断文件是否存在
    echo "File $filename exists."; cp $filename $filename.bak
else
    echo "File $filename not found."; touch $filename
fi; echo "File test complete."

william  william-pc  ~/DYJ/test                    4401  22:27:14
>> > cat test5.sh |grep -n "filename"
5:filename=ttt.sh
7:if [ -e "$filename" ]; then #if和then需要分隔，-e用于判断文件是否存在
8:    echo "File $filename exists."; cp $filename $filename.bak
10:    echo "File $filename not found."; touch $filename
```

- `grep -i [PATTERN] file` 不区分大小写

- `grep -c [PATTERN] file` 计算匹配的字符串的数量（**按行字符串匹配第一个**）



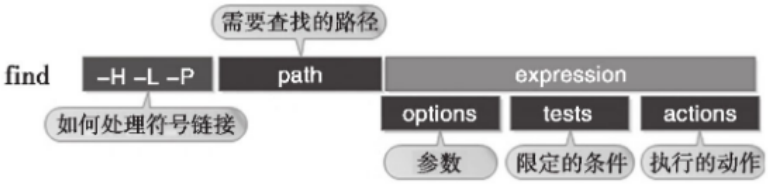
- `grep -w [PATTERN] file` 搜索关键字的字符串



12. find

`find`命令用于查找目录下的文件，同时也可以调用其他命令执行相应的操作。

find [-H] [-L] [-P] [-D debugopts] [-Olevel] [pathname] [expression]



命令参数

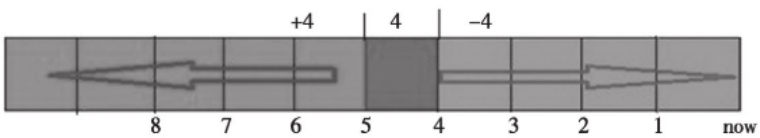
参数选项	解释说明（带※的为重点）
pathname	命令所查找的目录路径，例如用“.”来表示当前目录，用“/”来表示系统根目录※
Options 模块	
-depth	从指定目录下最深层的子目录开始查找
-maxdepth levels	查找的最大目录级数，levels 为自然数※
-regextype type	改变正则表达式的模式。默认为 emacs，还有 posix-awk、posix-basic、posix-egrep、posix-extended
Tests 模块	
-mtime [-n|n|+n]	按照文件的修改时间来查找文件（这个参数最常用），具体如下※ ❑ -n 表示文件更改时间距现在 n 天以内 ❑ +n 表示文件更改时间距现在 n 天以前 ❑ n 是距现在第 n 天
-atime [-n|n|+n]	按照文件的访问时间来查找文件，单位为天
-ctime [-n|n|+n]	按照文件的状态改变时间来查找文件，单位为天
-amin	按照文件的访问时间来查找文件，单位为分钟
-cmin	按照文件的状态改变时间来查找文件，单位为分钟
-mmin	按照文件的修改时间来查找文件，单位为分钟
-group	按照文件所属的组来查找文件
-name	按照文件名查找文件，只支持 *、?、[] 等特殊通配符※
-newer	查找更改时间比指定文件新的文件
-nogroup	查找没有有效用户组的文件，即该文件所属的组在 /etc/groups 中不存在
-nouser	查找没有有效属主的文件，即该文件的属主在 /etc/passwd 中不存在
-path pattern	指定路径样式，配合 -prune 参数排除指定目录
-perm	按照文件权限来查找文件
-regex	接正则表达式
-iregex	接正则表达式，不区分大小写
-size n[cwbkMG]	查找文件长度为 n 块的文件，带有 cwbkMG 时表示文件长度以字节计
-user	按照文件属主来查找文件

参数选项	解释说明（带※的为重点）
-type	查找某一类型的文件：※ b（块设备文件） c（字符设备文件） d（目录） p（管道文件） l（符号链接文件） f（普通文件） s（socket 文件） D（door）
Actions 模块	
-delete	将查找出的文件删除
-exec	对匹配的文件执行该参数所给出的 Shell 命令 ※
-ok	和 -exec 作用相同，但在执行每个命令之前，都会让用户先确定是否执行
-prune	使用这一选项可以使 find 命令不在当前指定的目录中查找
-print	将匹配的文件输出到标准输出（默认功能，使用中可省略）
OPERATORS	find 支持逻辑运算符
!	取反 ※
-a	取交集，全拼为 and※
-o	取并集，全拼为 or※

- `find path -atime tests`查找指定时间内修改过的文件（距现在2天）

```
william  william-pc  ~/DYJ/test                    4429  22:54:45
> find . atime 2
.
./t.back
./ttt.sh.bak
./backup.tar.gz
./test2.sh
./my.txt
./1.txt
./test18.sh
./t.txt
./tecmint_monitor.sh
./test24.sh
./2.txt
./test9.sh
./test20.sh
./test15.sh
./test6.sh
./test17.sh
./DYJ
./DYJ/t.back
./DYJ/ttt.sh.bak
./DYJ/test2.sh
```

find时间查找说明

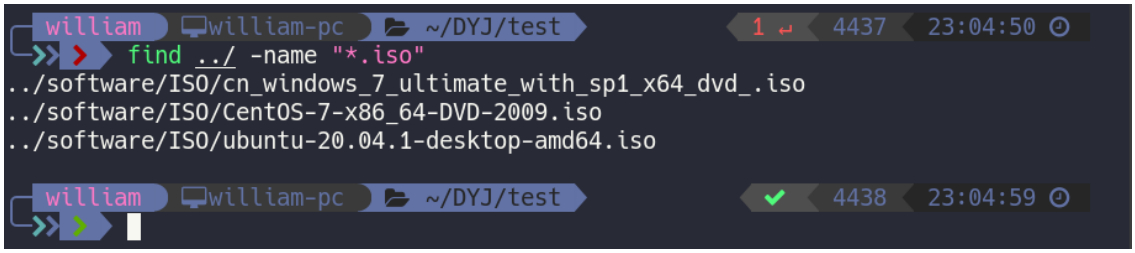


·-4表示文件更改时间距现在4天以内。

·+4表示文件更改时间距现在4天以前。

·4表示距现在第4天。

- `find -name` 用-name指定关键字查找



- `find -!` 取反，不是目录的文件

13. tail

tail命令用于显示文件内容的尾部，它默认输出文件的最后10行

tail [option] [file]

如果指定了多于一个文件，则在每一段输出前会给出文件名作为文件头

- `tail file` 显示文件后10行

- `tail -5 file` 显示文件后5行

- `tail -f file` 监控日志文件

14. useradd

useradd命令可用于创建新的用户或者更改用户的信息。

useradd [options] [login]

在使用useradd命令时，若不加任何参数选项，后面直接跟所添加的用户名，那么系统首先会读取/etc/login.defs（用户定义文件）和/etc/default/useradd（用户默认配置文件）文件中所定义的参数和规则，然后根据所设置的规则添加用户，同时还会向/etc/passwd（用户文件）和/etc/group（组文件）文件内添加新用户和新用户组记录，向/etc/shadow（用户密码文件）和/etc/gshadow（组密码文件）文件里添加新用户和组对应的密码信息的相关记录。同时系统还会根据/etc/default/useradd文件所配置的信息建立用户的家目录，并将/etc/skel中的所有文件（包括隐藏的环境配置文件）都复制到新用户的家目录中。

参数选项	解释说明（带※的为重点）
-c comment	新用户 password 文件中的说明栏（冒号分隔后的第五列）
-d home_dir	新用户每次登入时所使用的家目录
-e expire_date	用户终止日期。日期的指定格式为 YYYY-MM-DD
-f inactive_days	用户过期几日后永久停权。当值为 0 时用户立即被停权，而当值为 –1 时则关闭此功能，预设值为 –1
-g initial_group	指定用户对应的用户组。用户组名必须为系统现已存在的名称※
-G group,[...]	定义此用户为多个不同组的成员。每个用户组使用逗号（,）分隔。用户组名同 -g 选项的限制。默认值为用户的起始用户组
-m	用户目录如不存在则自动建立
-M	不建立用户家目录，优先于 /etc/login.defs 文件设定。创建虚拟用户时一般不需要建立家目录，部署应用服务时则需要创建虚拟用户
-n	默认情况下用户的用户组与用户的名称是相同的。如果命令添加了 -n 参数，就不会生成与用户同名的用户组了
-r	此参数是用来建立系统用户的。系统用户的 UID 会比定义在系统档上 /etc/login.defs. 的 UID_MIN 要小。注意此用法下 useradd 所建立的用户不会建立用户家目录，也不会在乎记录在 /etc/login.defs. 中的定义值。如果需要用户家目录必须额外指定 -m 参数来建立系统用户。这是 Red Hat 额外增设的选项
-s shell	用户登入后使用的 Shell 名称。默认值为不填写，这样系统会帮助指定预设的登入 Shell（根据 /etc/default/useradd 预设的值）※
-u uid	用户的 ID 值。这个值必须是唯一的，除非用 -o 选项。数字不可为负值※

useradd加-D选项参数说明：改变新建用户的预设值

当执行useradd带-D参数时，可以更改新建用户的默认配置值（/etc/default/useradd）或者由命令行编辑文件更改预设值。可简单理解该参数（-D）就是用于修改/etc/default/useradd配置文件的内容的，若这个文件的内容被修改，则添加新用户不加参数时默认值就会从该/etc/default/useradd中读取。

useradd -D 参数选项	注释说明
-b default_home	定义用户家目录的基本目录，当用户家目录不存在时，此目录将作为家目录生效
-e default_expire_date	用户账号停止日期，格式为 YYYY-MM-DD，同 useradd 的 -e 参数
-f default_inactive	用户过期几日后停权。同 useradd 的 -f 参数
-g default_group	新用户起始用户组名或 ID。用户组名必须为现已存在的名称。用户组 ID 也必须为现已存在的用户组。同 useradd 的 -g 参数
-s default_shell	用户登入后使用的 Shell 名称。修改后新加入的用户都将使用此 Shell 类型，同 useradd -s 参数

- `useradd username` 不加任何参数添加用户

```
[root@VM-0-15-centos ~]# useradd austin && ls -ld /home/austin
drwx------ 2 austin austin 4096 Feb 28 22:56 /home/austin
[root@VM-0-15-centos ~]#
```

创建用户的同时还会创建一个与用户名相同的用户组

下面再来查看/etc/passwd文件中有关新用户austin的记录

```
[root@VM-0-15-centos ~]# grep -w austin /etc/passwd
austin:x:1175:1175::/home/austin:/bin/bash
[root@VM-0-15-centos ~]#
```

这里的1175:1175就是根据/etc/login.defs内容预设的

接下来，我们再看看/etc/shadow、/etc/group和/etc/gshadow文件，是不是也存在与austin用户有关的记录

```
[root@VM-0-15-centos ~]# grep -w austin /etc/shadow
austin:!!:18686:0:99999:7:::
[root@VM-0-15-centos ~]#
```

虽然没有创建密码但是密码文件还是添加了一行信息

```
[root@VM-0-15-centos ~]# grep -w austin /etc/group
austin:x:1175:
[root@VM-0-15-centos ~]#
```

创建用户时，默认创建与用户同名的用户组，并体现在用户组配置文件中

```
[root@VM-0-15-centos ~]# grep -w austin /etc/gshadow
austin:!::
[root@VM-0-15-centos ~]#
```

组密码文件中也会有一行相关记录

根据上文的结果，我们将会发现/etc/shadow、/etc/group和/etc/gshadow几个文件都存在与austin用户相关的记录

15. chown

chown命令用于改变文件或目录的用户和用户组

chown [option] [OWNER] [:[GROUP]] file

```
chown 用户      文件或目录     #仅仅授权用户
chown :组       文件或目录     #仅仅授权组
chown 用户:组   文件或目录     #表示授权用户和组
“:”可与用“.”代替
```

- `chown user file`更改用户所属的用户属性

```
[root@william-virtualbox DYJ]# ls -l
total 128
-rw-r--r-- 1 william william  5300  6月   3  2012 create.txt
drwxr-xr-x 9 william william  4096  1月   8 20:26 hexo
drwxr-xr-x 6 william william  4096 12月  30 22:50 jd
-rw-r--r-- 1 william william  7168  6月   3  2012 populate.txt
-rw-r--r-- 1 william william 97736 10月  31  2012 README.pdf
drwxr-xr-x 4 william william  4096  1月  25 20:26 software
drwxr-xr-x 2 william william  4096  1月  19 22:03 web
[root@william-virtualbox DYJ]# chown root create.txt && ls -l |grep create.txt
-rw-r--r-- 1 root    william  5300  6月   3  2012 create.txt
```

- `chown .group file` 更改文件所属的组属性

```
[root@william-virtualbox DYJ]# chown root create.txt && ls -l |grep create.txt
-rw-r--r-- 1 root    william  5300  6月   3  2012 create.txt
[root@william-virtualbox DYJ]# chown .root create.txt && ls -l |grep create.txt
-rw-r--r-- 1 root    root     5300  6月   3  2012 create.txt
```

用 " . " 代替 " : "

- `chown user:group file` 同时更改用户和组属性（使用"."也可以）

```
[root@william-virtualbox DYJ]# chown .root create.txt && ls -l |grep create.txt
-rw-r--r-- 1 root    root     5300  6月   3  2012 create.txt
[root@william-virtualbox DYJ]# chown william:william create.txt && ls -l |grep cr
eate.txt
-rw-r--r-- 1 william william  5300  6月   3  2012 create.txt
```

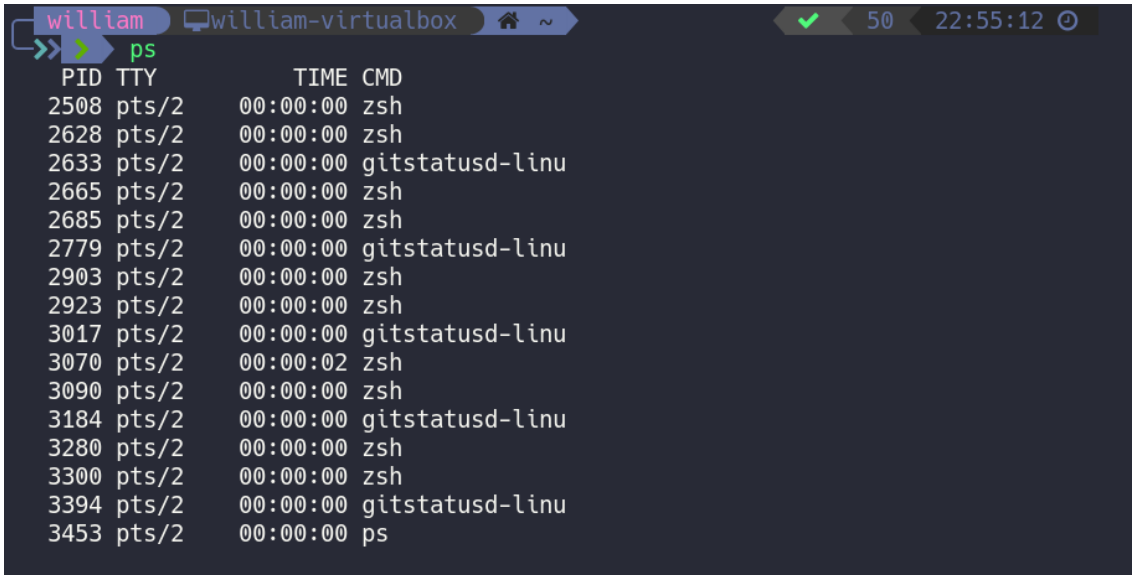
16. ps

ps命令用于列出执行ps命令的那个时刻的进程快照，就像用手机给进程照了一张照片。如果想要动态地显示进程的信息，就需要使用top命令，该命令类似于把手机切换成录像模式。

ps [option]

参数选项	解释说明（带 ※ 的为重点）
-a	显示所有终端下执行的进程
a	显示与终端相关的所有进程，包含每个进程的完整路径 ※
x	显示与终端无关的所有进程 ※
u	显示进程的用户信息 ※
-u	显示指定用户相关的进程信息
-e	显示所有进程 ※
-f	额外显示 UID、PPID、C 与 STIME 栏位 ※
f	显示进程树
-H	显示进程树
-l	以详细的格式来显示进程的状况
-o	自定义输出指定的字段，以逗号分隔
--sort key	key 表示为指定字段排序，默认升序，+key 升序，-key 降序

- `ps`命令不接任何参数



  默认情况下，ps命令不接任何参数时，输出的是使用者当前所在终端（窗口）的进程，其输出结果中的各项说明如下。

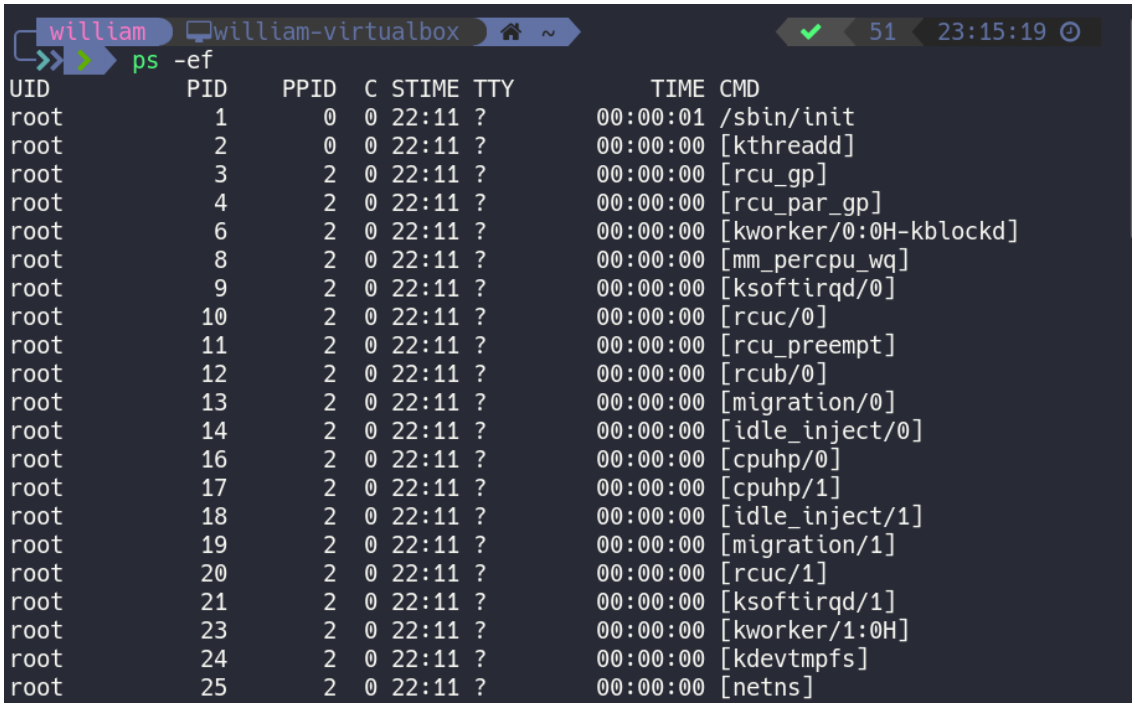
  ·PID是进程的标识号。

  ·TTY是进程所属的终端控制台。

  ·TIME列是进程所使用的总的CPU时间。

  ·CMD列是正在执行的命令行。

- `ps-ef` -e 显示所有进程，-f 格外显示UID、PPID、C与STIME



输出信息中各列的说明如下。

·UID：进程被该UID所拥有。

·PID：进程的标识号。

·PPID：进程的父进程的标识号。

·C：CPU使用的资源百分比。

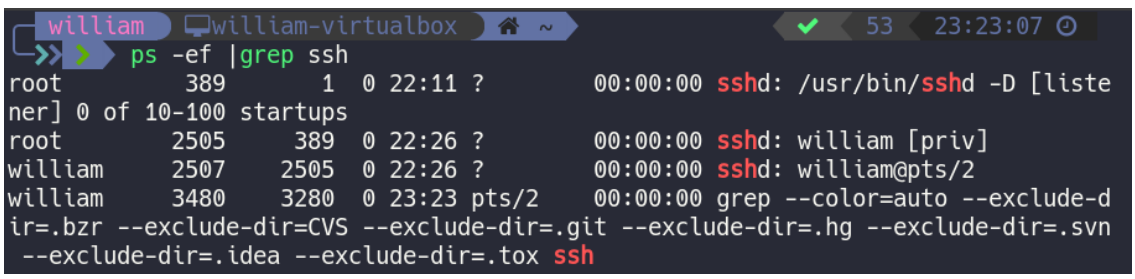
·STIME：进程开始的时间。

·TTY：该进程是在哪个终端机上面运作，若与终端机无关，则显示“?”，另外，tty1-tty6是本机上面的登入者进程，若为pts/0等，则表示为由网络连接进主机的进程。

·TIME：进程所使用的总的CPU时间。

·CMD：正在执行的命令行。

- `ps -ef |grep name` ps与grep的组合用法，用于查找特定进程



- `ps -u root` 显示示指定用户的相关进程信息

```
william  william-virtualbox  ~                                  54  23:23:19
> ps -u root
  PID TTY          TIME CMD
    1 ?        00:00:01 systemd
    2 ?        00:00:00 kthreadd
    3 ?        00:00:00 rcu_gp
    4 ?        00:00:00 rcu_par_gp
    6 ?        00:00:00 kworker/0:0H-kblockd
    8 ?        00:00:00 mm_percpu_wq
    9 ?        00:00:00 ksoftirqd/0
   10 ?        00:00:00 rcuc/0
```

17. kill

kill命令能够终止你希望停止的进程。

kill [-lps] [pid]

- `kill -l` 列出所有信号的名称

```
william  william-virtualbox  ~                                  55  23:25:21
> kill -l
HUP INT QUIT ILL TRAP ABRT BUS FPE KILL USR1 SEGV USR2 PIPE ALRM TERM STKFLT CHLD
 CONT STOP TSTP TTIN TTOU URG XCPU XFSZ VTALRM PROF WINCH POLL PWR SYS
```

信　　号	说　　明
HUP(1)	挂起，通常因终端掉线或用户退出而引发
INT(2)	中断，通常是按下 Ctrl+c 组合键来发出这个信号
QUIT(3)	退出，通常是按下 Ctrl+\ 组合键来发出这个信号
KILL(9)	立即结束进程的运行
TERM(15)	终止，通常在系统关机时发送
TSTP(20)	暂停进程的运行，通常是按下 Ctrl+z 组合键来发出这个信号

- `kill -s` 指定要发送的信号

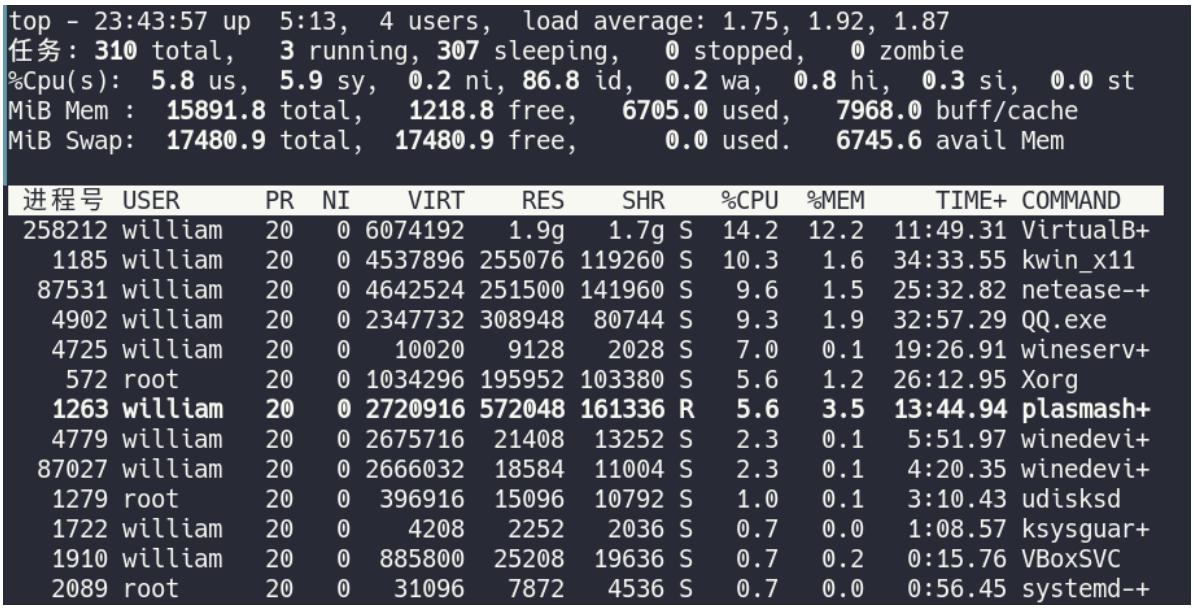
```
[root@VM-0-15-centos ~]# kill -s 15 18121 && ps
  PID TTY          TIME CMD
10883 pts/2    00:00:00 bash
18121 pts/2    00:00:00 zsh
18160 pts/2    00:00:00 bash
18345 pts/2    00:00:00 ps
```

- `kill -9 PID` KILL（9）信号强制终止进程

```
[root@VM-0-15-centos ~]# kill -9 18121 && ps
Killed
[root@VM-0-15-centos ~]#   PID TTY          TIME CMD
10883 pts/2    00:00:00 bash
18160 pts/2    00:00:00 bash
18415 pts/2    00:00:00 ps
```

18. top

实时显示系统中各个进程的资源占用状况



```
top - 23:43:57 up  5:13,  4 users,  load average: 1.75, 1.92, 1.87
任务：310 total,   3 running, 307 sleeping,   0 stopped,   0 zombie
%Cpu(s):  5.8 us,  5.9 sy,  0.2 ni, 86.8 id,  0.2 wa,  0.8 hi,  0.3 si,  0.0 st
MiB Mem :  15891.8 total,   1218.8 free,   6705.0 used,   7968.0 buff/cache
MiB Swap:  17480.9 total,  17480.9 free,      0.0 used.   6745.6 avail Mem

  进程号 USER      PR  NI    VIRT    RES    SHR    %CPU  %MEM     TIME+ COMMAND
 258212 william   20   0 6074192   1.9g   1.7g S  14.2  12.2  11:49.31 VirtualB+
   1185 william   20   0 4537896 255076 119260 S  10.3   1.6  34:33.55 kwin_x11
  87531 william   20   0 4642524 251500 141960 S   9.6   1.5  25:32.82 netease-+
   4902 william   20   0 2347732 308948  80744 S   9.3   1.9  32:57.29 QQ.exe
   4725 william   20   0   10020   9128   2028 S   7.0   0.1  19:26.91 wineserv+
    572 root      20   0 1034296 195952 103380 S   5.6   1.2  26:12.95 Xorg
   1263 william   20   0 2720916 572048 161336 R   5.6   3.5  13:44.94 plasmash+
   4779 william   20   0 2675716  21408  13252 S   2.3   0.1   5:51.97 winedevi+
  87027 william   20   0 2666032  18584  11004 S   2.3   0.1   4:20.35 winedevi+
   1279 root      20   0  396916  15096  10792 S   1.0   0.1   3:10.43 udisksd
   1722 william   20   0    4208   2252   2036 S   0.7   0.0   1:08.57 ksysguar+
   1910 william   20   0  885800  25208  19636 S   0.7   0.2   0:15.76 VBoxSVC
   2089 root      20   0   31096   7872   4536 S   0.7   0.0   0:56.45 systemd-+
```

第一行，任务队列信息，同uptime命令的执行结果。

·23：43：57 当前系统时间。

·up 5:13 系统已经运行了0天5小时13分。

·4 users 当前有4个用户登录系统。

·load average：1.75，1.92，1.87 load average后面的三个数分别是1分钟、5分钟、15分钟的平均负载情况。

第二行，Tasks为任务（进程）。从上面的信息可以看出，系统现在共有310个进程，其中处于运行状态的有3个，307个在休眠（sleep），stoped状态0个，zombie状态（僵死）的有0个。

第三行，CPU状态信息。

·5.8% us 用户空间占用CPU的百分比。

·5.9% sy 内核空间占用CPU的百分比。·

.0.2% ni 改变过优先级的进程占用CPU的百分比。·

.86.8% id 空闲CPU百分比。

·0.2% wa I/O等待占用CPU的百分比。

·0.8% hi 硬中断（Hardware IRQ）占用CPU的百分比

·0.3% si 软中断（Software Interrupts）占用CPU的百分比。

·0.3%st 虚拟机占用CPU的百分比。

第四行，内存状态。

·15891.8M total 物理内存总量。

·6705.0M used 使用中的内存总量。

·1218.8M free 空闲内存总量。

·7968.0M buffers 缓冲的内存量。

第五行，swap交换分区信息。

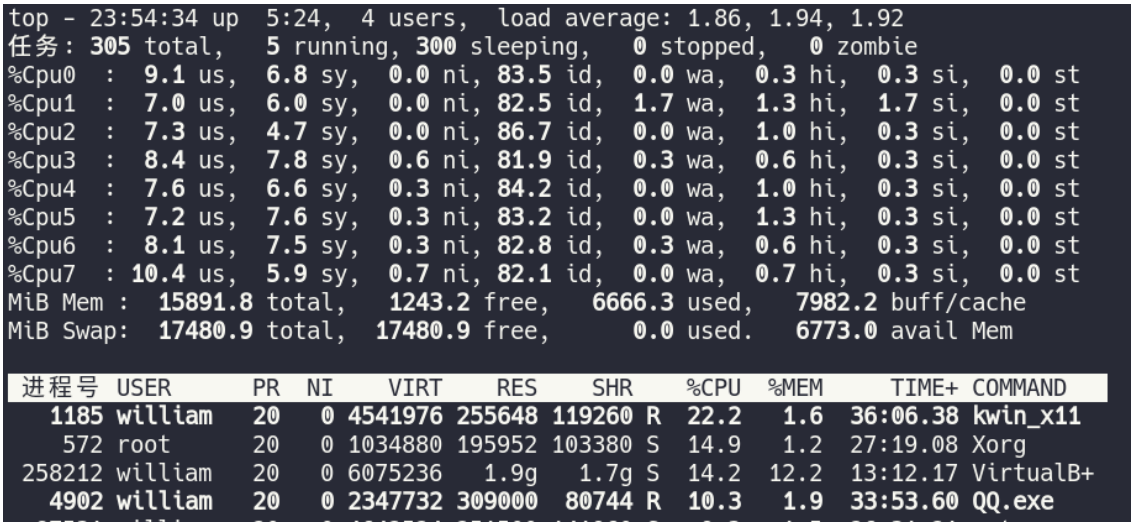
·17480.9M total 交换区总量。

·0.0M used 使用的交换区总量。

·17480.9M free 空闲交换区总量。

·7968.0M cached 缓存的内存量。

- 按数字“1”，显示每个逻辑CPU状况



19.free

free命令用于显示系统内存状态，具体包括系统物理内存、虚拟内存、共享内存和系统缓存等

free [option]

参数选项	解释说明（带※的为重点）
-b	以Byte为单位显示内存的使用情况
-m	以MB为单位显示内存的使用情况※
-K	以KB为单位显示内存的使用情况
-h	以人类可读的形式显示内存的使用情况※
-t	显示内存总和列
-s＜间隔秒数＞	根据指定的间隔秒数持续显示内存的使用情况※
-o	不显示系统缓冲区列

- `free -m` 以MB为单位显示内存使用情况

```
[dyj@VM-0-15-centos test]$ free
              total        used        free      shared  buff/cache   available
Mem:        1882064      377332      232732         924     1272000     1316332
Swap:             0           0           0
[dyj@VM-0-15-centos test]$ free -m
              total        used        free      shared  buff/cache   available
Mem:           1837         368         227           0        1242        1285
Swap:             0           0           0
```

- `free -h` 根据实际大小自动转换成KB、MB、GB单位

```
[dyj@VM-0-15-centos test]$ free -h
              total        used        free      shared  buff/cache   available
Mem:           1.8G        369M        225M       924K        1.2G        1.3G
Swap:            0B          0B          0B
[dyj@VM-0-15-centos test]$
```

·Linux系统的特性是将不用的物理内存缓存起来，因此327MB不是系统的真实剩余内存。

·系统真正可用的内存为390MB。

·buffers为写入数据缓冲区。

·cache为读取数据的缓存区。

20. netstat

netstat命令用于显示本机网络的连接状态、运行端口和路由表等信息。

netstat [option]

参数选项	解释说明（带※的为重点）
-r	显示路由表信息，该功能类似于前面学过的 route 和 ip route
-g	显示多播功能群组成员，该功能类似于前面学过的 ip maddr
-i	显示网络接口信息，该功能类似于前面学过的 ip -s link
-s	显示各类协议的统计信息
-n	显示数字形式的地址而不是去解析主机、端口或用户名。默认情况下，netstat 命令会尝试解析并显示主机的主机名，这个过程通常比较长也是非必需的 ※
-a	显示处于监听状态和非监听状态的 socket 信息 ※
-A	显示指定网络类型的网络连接状态
-c < 秒数 >	后面跟的秒数表示每隔几秒就刷新显示一次 ※
-l	仅显示连接状态为“LISTEN”的服务的网络状态
-t	显示所有的 TCP 连接情况 ※
-u	显示所有的 UDP 连接情况 ※
-p	显示 socket 所属进程的 PID 和名称 ※

- `netstat-an` 显示所有连接信息

```
[dyj@VM-0-15-centos ~]$ netstat -an
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 172.17.0.15:22          120.228.254.245:15712   ESTABLISHED
tcp        0      0 172.17.0.15:22          221.202.243.22:22729    ESTABLISHED
tcp        0      0 172.17.0.15:22          113.116.223.45:18678    ESTABLISHED
tcp        0     36 172.17.0.15:22          121.69.103.245:2082     ESTABLISHED
tcp        0      0 172.17.0.15:22          221.202.243.22:21513    ESTABLISHED
tcp        0      0 172.17.0.15:22          120.228.254.245:15408   ESTABLISHED
tcp        0      0 172.17.0.15:22          221.202.243.22:20123    ESTABLISHED
tcp        0      0 172.17.0.15:22          221.202.243.22:53472    ESTABLISHED
tcp        0      0 172.17.0.15:22          121.69.103.245:2079     ESTABLISHED
tcp        0      0 172.17.0.15:22          120.228.254.245:16060   ESTABLISHED
tcp        0      0 172.17.0.15:22          113.116.223.45:18677    ESTABLISHED
tcp        0      0 172.17.0.15:40070       169.254.0.55:5574       ESTABLISHED
tcp        0      0 172.17.0.15:22          120.228.254.245:15551   ESTABLISHED
tcp        0      0 172.17.0.15:22          120.228.254.245:15550   ESTABLISHED
tcp        0      0 172.17.0.15:22          221.202.243.22:53476    ESTABLISHED
tcp6       0      0 :::3307                 :::*                    LISTEN
udp        0      0 0.0.0.0:68              0.0.0.0:*
udp        0      0 172.17.0.15:123         0.0.0.0:*
udp        0      0 127.0.0.1:123           0.0.0.0:*
udp6       0      0 fe80::5054:ff:fef4::123 :::*
```

“Active Internet connections （servers and established）活动的TCP/IP网络连接”

```
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  3      [ ]         DGRAM                    7441     /run/systemd/notify
unix  2      [ ]         DGRAM                    7443     /run/systemd/cgroups-agent
unix  2      [ ACC ]     STREAM     LISTENING     18038    /usr/local/qcloud/YunJing/conf/ydrpc
_1
unix  2      [ ACC ]     STREAM     LISTENING     18074    /usr/local/qcloud/YunJing/conf/ydrpc
_2
unix  2      [ ACC ]     STREAM     LISTENING     7460     /run/systemd/journal/stdout
unix  5      [ ]         DGRAM                    7463     /run/systemd/journal/socket
unix  27     [ ]         DGRAM                    7465     /dev/log
unix  2      [ ACC ]     STREAM     LISTENING     13876    /var/run/acpid.socket
unix  2      [ ACC ]     STREAM     LISTENING     38599    /var/lib/mysql/mysql.sock
unix  2      [ ACC ]     STREAM     LISTENING     13042    @ISCSID_UIP_ABSTRACT_NAMESPACE
unix  2      [ ACC ]     STREAM     LISTENING     10595    /run/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     10869    /run/lvm/lvmetad.socket
unix  2      [ ACC ]     STREAM     LISTENING     13446    /var/run/lsm/ipc/simc
unix  2      [ ACC ]     STREAM     LISTENING     13448    /var/run/lsm/ipc/sim
unix  2      [ ACC ]     STREAM     LISTENING     10659    /run/lvm/lvmpolld.socket
unix  2      [ ACC ]     STREAM     LISTENING     13038    @ISCSIADM_ABSTRACT_NAMESPACE
unix  2      [ ACC ]     SEQPACKET  LISTENING     10700    /run/udev/control
unix  2      [ ]         DGRAM                    10971    /run/systemd/shutdownd
unix  2      [ ACC ]     STREAM     LISTENING     13039    /run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM     LISTENING     18164    /usr/local/yd.socket.server
```

“Active UNIX domain sockets （servers and established）活动的unix socket连接”

Proto：socket使用的协议（TCP、UDP、RAW）

Recv-Q：接收到但是还未处理的字节数

Send-Q：已经发送但是未被远程主机确认收到的字节数

Local Address：本地主机地址和端口

Foreign Address：远程主机地址和端口

State：socket的状态，通常仅仅有TCP的状态，状态值可能有ESTABLISHED、SYN_SENT、SYN_RECV、FIN_WAIT1、FIN_WAIT2、TIME_WAIT等

状　态	含　义
ESTABLISHED	socket 已经建立连接，表示处于连接的状态，一般认为有一个 ESTABLISHED 是一个服务的并发连接。该连接状态在生产场景中很重要，需要重点关注
SYN_SENT	socket 正在积极尝试建立一个连接，即处于发送后连接前的一个等待但未匹配进入连接的状态
SYN_RECV	已经从网络上收到一个连接请求
FIN_WAIT1	socket 已关闭，连接正在或正要关闭
FIN_WAIT2	连接已关闭，并且 socket 正在等待远端结束
TIME_WAIT	socket 正在等待关闭处理仍在网络上的数据包，这个连接状态在生产场景中很重要，需要重点关注
CLOSED	socket 不再被占用了
CLOSE_WAIT	远端已经结束，等待 socket 关闭
LAST_ACK	远端已经结束，并且 socket 也已关闭，等待 acknowledgement
LISTEN	socket 正在监听连接请求
CLOSING	socket 关闭，但是我们仍旧没有发送数据
UNKNOWN	socket 状态未知

- `netstat -lntup` 显示所有TCP和UDP正在监听的连接信息

```
[dyj@VM-0-15-centos ~]$ netstat -lntup
(No info could be read for "-p": geteuid()=1150 but you should be root.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program nam
e
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -

tcp6       0      0 :::3307                 :::*                    LISTEN      -

udp        0      0 0.0.0.0:68              0.0.0.0:*                           -

udp        0      0 172.17.0.15:123         0.0.0.0:*                           -

udp        0      0 127.0.0.1:123           0.0.0.0:*                           -

udp6       0      0 fe80::5054:ff:fef4::123 :::*                                -

udp6       0      0 ::1:123                 :::*                                -

[dyj@VM-0-15-centos ~]$
```

21. su

su命令用于将当前用户切换到指定用户或者以指定用户的身份执行命令或程序。

su [option] [user]

从root用户切换到普通用户时，不需要任何密码；从普通用户切换到root用户时，需要输入root密码。

- `su - user` **-, -l, --login 切换用户的同时，将用户的家目录、系统环境等重新按切换后的用户初始化**

```
william william-pc ~ 4334 21:05:25
>> whoami
william

william william-pc ~ 4335 21:05:39
>> su root
密码：
[william-pc william]# pwd
/home/william
[william-pc william]# env | egrep "USER|MAIL|PWD|LOGNAME"
PWD=/home/william
LOGNAME=william
USER=william
MAIL=/var/spool/mail/william
OLDPWD=/home/william
[william-pc william]# exit
exit

william william-pc ~ 4336 21:06:18
>> su - root
密码：
[william-pc ~]# pwd
/root
[william-pc ~]# env | egrep "USER|MAIL|PWD|LOGNAME"
PWD=/root
LOGNAME=root
USER=root
[william-pc ~]#
```

- `su -c user` 向Shell传递单个命令

22. echo

echo命令能将指定的文本显示在Linux命令行上

echo [option] [string]

- `echo -n string` 不换行输出

vi / vim

vim [option] [file]

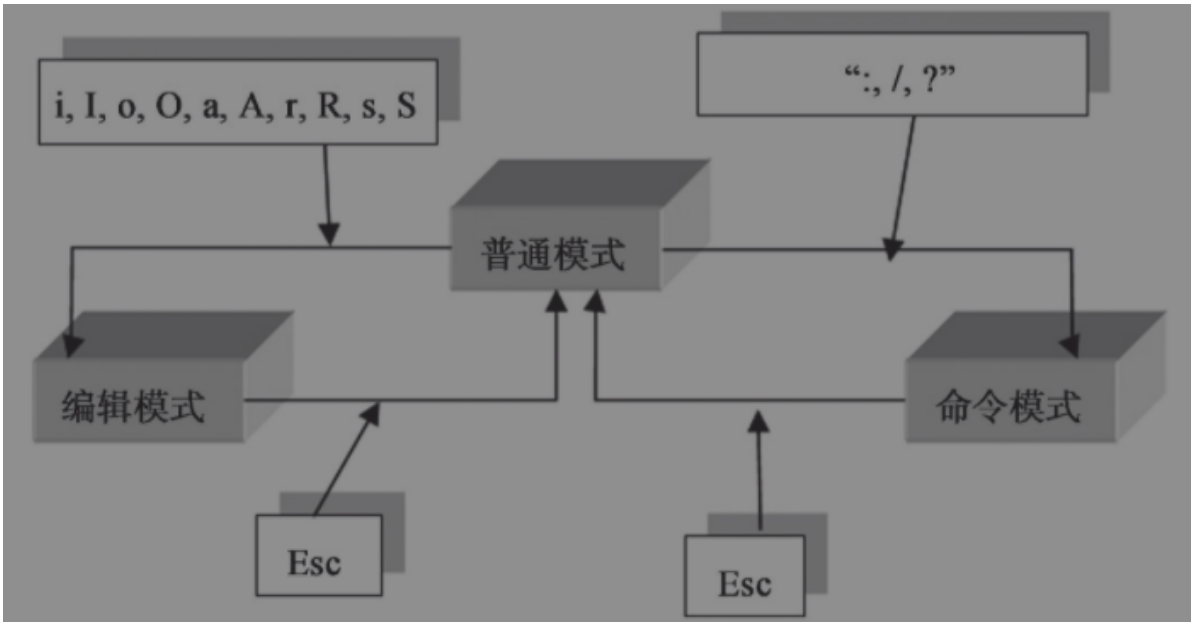
vi是Linux命令行界面下的文字编辑器，几乎所有的Linux系统都安装了vi，只要学会了vi这个编辑工具，就可以在任何Linux系统上使用它。而vim是vi命令的增强版（Vi IMproved），与vi编辑器完全兼容，此外还有很多增强功能，例如用不同颜色高亮显示代码。因此，如果系统有vim命令，那么建议大家就使用vim编辑文本。

一般来说，vim可分为三种模式：普通模式、编辑模式、命令模式。这三种模式的作用分别如下。

（1）普通模式用vim命令打开一个文件，默认的状态就是普通模式。在这个模式中，不能进行编辑输入操作，但可以按“上下左右”键来移动光标，也可以执行一些操作命令进行如删除、复制、粘贴等之类的工作。

（2）编辑模式在普通模式下不能进行编辑输入操作，只有按下“i，I，o，O，a，A，r，R，s，S”（其中“I”最常用）等字母进入编辑模式之后才可以执行录入文字等编辑操作。看文件是否处于编辑模式状态有一个重要的特征，那就是在窗口的左下角要有插入的标记“--INSERT--”或“--插入--”

（3）命令模式在普通模式下，输入“：”或“/”或“?”时，光标会自动定位在那一行，在这个模式中，可以执行保存、退出、搜索、替换、显示行号等相关操作。



vim命令的参数选项及说明

命　令	说　明
普通模式：移动光标的操作	
G 或（shift+g）	将光标移动到文件的最后一行
gg	将光标移动到文件的第一行，等价于 1gg 或 1G
0	数字 0，将光标从所在位置移动到当前行的开头
$	从光标所在位置将光标移动到当前行的结尾
n<Enter>	n 为数字，<Enter> 为回车键，将光标从当前位置向下移动 *n* 行

（续）

命　令	说　明
ngg	n 为数字，移动到文件的第 *n* 行，如 11gg 可移动到第 11 行，可配合“:set nu”查看，同 nG
H	光标移动到当前窗口最上方的那一行
M	光标移动到当前窗口中间的那一行
L	光标移动到当前窗口最下方的那一行
h 或（←）	光标向左移动一个字符
j 或（↓）	光标向下移动一个字符
k 或（↑）	光标向上移动一个字符
l 或（→）	光标向右移动一个字符
普通模式：搜索与替换操作	
/oldboy	从光标位置开始，向下寻找名为 oldboy 的字符串
?oldboy	从光标位置开始，向上寻找名为 oldboy 的字符串
n	从光标位置开始，向下重复前一个搜索的动作
N	从光标位置开始，向上重复前一个搜索的动作
:g/A/s//B/g	把符合 A 的内容全部替换为 B，斜线为分隔符，可以用 @、# 等替代
:%s/A/B/g	把符合 A 的内容全部替换为 B，斜线为分隔符，可以用 @、# 等替代
:n1,n2s/A/B/gc	n1、n2 为数字，在第 n1 行和 n2 行之间寻找 A，用 B 替换
普通模式：复制、粘贴、删除等操作	
yy	复制光标所在的当前行
nyy	n 为数字，复制光标开始向下共 *n* 行
p/P	p 将已复制的数据粘贴到光标的下一行，P 则为粘贴到光标的上一行
dd	删除光标所在的当前行
ndd	n 为数字，删除从光标开始向下共 *n* 行
u	恢复（回滚）前一个执行过的操作
.	点号。重复前一个执行过的动作
x	向后删除字符
X	向前删除字符
d1G	删除当前行至第一行
dG	删除当前行至最后一行
d0	删除当前光标文本至行首

（续）

命　令	说　明
d$	删除当前光标文本至行尾
进入编辑模式命令	
i	在当前光标所在处插入文字
a	在当前光标所在的下一个字符处插入文字
I	在当前所在行行首的第一个非空格符处开始插入文字，和 A 相反
A	在当前所在行行尾的最后一个字符处开始插入文字，和 I 相反
O	在当前所在行的上一行处插入新的一行
o	在当前所在行的下一行处插入新的一行
Esc	退出编辑模式，回到命令模式中
命令行模式	
:wq	退出并保存
:wq!	退出并强制保存，“!”为强制的意思
:q!	强制退出，不保存
:n1,n2 w filename	n1、n2 为数字，将 n1 行到 n2 行的内容保存成 filename 这个文件
:n1,n2 co n3	n1、n2 为数字，将 n1 行到 n2 行的内容复制到 n3 位置下
:n1,n2 m n3	n1、n2 为数字，将 n1 行到 n2 行的内容剪切至 n3 位置下
:!command	暂时离开 vi 到命令行模式下执行 command 的显示结果！例如 :! ls /etc
:set nu	显示行号
:set nonu	与 set nu 相反，取消行号
:vs filename	垂直分屏显示，同时显示当前文件和 filename 对应文件的内容
:sp filename	水平分屏显示，同时显示当前文件和 filename 对应文件的内容
I + # + Esc	在可视块模式下（Ctrl+V），一次性注释所选的多行，取消注释可用“:n1,n2s/#//gc”，这里的操作是一个通用的方法，# 号可以换成别的操作，例如 Tab 键，这样就是批量缩进
Del	在可视块模式下（Ctrl+V），一次性删除所选内容
r	在可视块模式下（Ctrl+V），一次性替换所选内容