Linux文本处理工具和正则表达式

一.查看、截取和修改文本的工具

1.查看文本的工具

cat

- 最常用的文件查看命令；当不指明文件或者文件名为一杠'-'时，读取标准输入。

```
cat [OPTION]... [FILE]...
-A：显示所有控制符(tab键:^I;行结束符:$)
-E：显示行结束符$
-n：对显示出的每一行进行编号
-b：只对非空行进行编号
-s：压缩连续的空行成一行
```

more

- 默认将多行文本满屏输出,只允许向文本末尾翻页（空格键满屏翻页；回车键单行翻页），阅读到文末自动退出。
- 用法

```
more [options] file...
```

- 一般使用管道传给more来阅读内容超过满屏的文本

```
cat big_file1 | more
```

less

- 功能和more类似，但是允许向前和向后翻页，阅读到文本末尾不会自动退出。
- 另外，less不必读取完整的文本，这使得其打开较大的文本文件时比vi等工具更具有速度优势。
- 翻页快捷键

空格键向下满屏翻页
回车键单行向下翻页
k键单行向上
u键半屏向上

f键半屏向下

- 向下的快捷键同样适用于more

nl

- nl将文本文件传给标准输出，并添给每行文本加行号；当不指明文件或者文件名为一杠'-'时，读取标准输入。
- 用法

```
nl [OPTION]... [FILE]...
    -i#   # 表示行号间隔#个数递增
```

- 下面的用法只给带root的行加行号（匹配基本正则表达式）

```
nl -b, --body-numbering=pBRE file             # BRE 表示基本正则表达式
nl --body-numbering=proot /etc/passwd
```

tac

- cat 的反用，最后一行先显示；当不指明文件或者文件名为一杠'-'时，读取标准输入。

rev

- rev读取指定的文件，倒序每行的字符，输出到标准输出；如果不指定文件，读取标准输入。

```
[root@centos8 ~]$ rev
hello    # 输入
olleh    # 输出
howdy    # 输入
ydwoh    # 输出
```

2.查看非文本文件的工具

hexdump

- hexdump 将文件内容以ascii字符, decimal十进制, hexadecimal十六进制, 或者octal八进制显示在标准输出。

- 用法

```
hexdump [options] file [...]
   -C # 大写C字母表示按照标准的16进制ASCII码显示文件内容
```

```
    -n length # 只显示前n字节的内容
    -s offset # 跳过文件内容的前offset个字节显示
注：length和offset的格式为：1KiB=1024字节；1MiB=1024KiB；... 或者：1KB=1000字
节;1MB=1000KB...
也就是说下面的写法等价：
[root@centos8 /data]$ hexdump -C -n 1KiB /dev/nvme0n1 | wc -l
56                                    ^^^^
[root@centos8 /data]$ hexdump -C -n 1024 /dev/nvme0n1 | wc -l
56                                    ^^^^
###################################
[root@centos8 /data]$ hexdump -C -n 1KiB /dev/nvme0n1 | tail -n3
000003f0  00 00 00 00 02 00 00 00  00 00 00 00 77 00 20 08  |............w. .|
00000400
[root@centos8 /data]$ hexdump -C -n 1024 /dev/nvme0n1 | tail -n3
000003f0  00 00 00 00 02 00 00 00  00 00 00 00 77 00 20 08  |............w. .|
00000400
###################################
[root@centos8 /data]$hexdump -C -n 1KB /dev/nvme0n1 | tail -n3
*                                     ^^^
000003e0  00 00 00 00 00 00 00 00                           |........|
000003e8
[root@centos8 /data]$hexdump -C -n 1000 /dev/nvme0n1 | tail -n3
*                                     ^^^^
000003e0  00 00 00 00 00 00 00 00                           |........|
000003e8
```

```
[root@centos8 /data]$ hexdump  -C -n 512 /dev/nvme0n1    # 查看MBR
00000000  eb 63 90 10 8e d0 bc 00  b0 b8 00 00 8e d8 8e c0  |.c..............|
00000010  fb be 00 7c bf 00 06 b9  00 02 f3 a4 ea 21 06 00  |...|.........!..|
00000020  00 be be 07 38 04 75 0b  83 c6 10 81 fe fe 07 75  |....8.u........u|
00000030  f3 eb 16 b4 02 b0 01 bb  00 7c b2 80 8a 74 01 8b  |.........|...t..|
00000040  4c 02 cd 13 ea 00 7c 00  00 eb fe 00 00 00 00 00  |L.....|.........|
00000050  00 00 00 00 00 00 00 00  00 00 00 80 01 00 00 00  |................|
00000060  00 00 00 00 ff fa 90 90  f6 c2 80 74 05 f6 c2 70  |...........t...p|
00000070  74 02 b2 80 ea 79 7c 00  00 31 c0 8e d8 8e d0 bc  |t....y|..1......|
00000080  00 20 fb a0 64 7c 3c ff  74 02 88 c2 52 be 05 7c  |. ..d|<.t...R..||
00000090  b4 41 bb aa 55 cd 13 5a  52 72 3d 81 fb 55 aa 75  |.A..U..ZRr=..U.u|
000000a0  37 83 e1 01 74 32 31 c0  89 44 04 40 88 44 ff 89  |7...t21..D.@.D..|
000000b0  44 02 c7 04 10 00 66 8b  1e 5c 7c 66 89 5c 08 66  |D.....f..\|f.\.f|
000000c0  8b 1e 60 7c 66 89 5c 0c  c7 44 06 00 70 b4 42 cd  |..\`|f.\..D..p.B.|
000000d0  13 72 05 bb 00 70 eb 76  b4 08 cd 13 73 0d 5a 84  |.r...p.v....s.Z.|
000000e0  d2 0f 83 de 00 be 85 7d  e9 82 00 66 0f b6 c6 88  |.......}...f....|
000000f0  64 ff 40 66 89 44 04 0f  b6 d1 c1 e2 02 88 e8 88  |d.@f.D..........|
00000100  f4 40 89 44 08 0f b6 c2  c0 e8 02 66 89 04 66 a1  |.@.D.......f..f.|
00000110  60 7c 66 09 c0 75 4e 66  a1 5c 7c 66 31 d2 66 f7  |\`|f..uNf.\|f1.f.|
00000120  34 88 d1 31 d2 66 f7 74  04 3b 44 08 7d 37 fe c1  |4..1.f.t.;D.}7..|
00000130  88 c5 30 c0 c1 e8 02 08  c1 88 d0 5a 88 c6 bb 00  |..0........Z....|
00000140  70 8e c3 31 db b8 01 02  cd 13 72 1e 8c c3 60 1e  |p..1......r...\`.|
00000150  b9 00 01 8e db 31 f6 bf  00 80 8e c6 fc f3 a5 1f  |.....1..........|
00000160  61 ff 26 5a 7c be 80 7d  eb 03 be 8f 7d e8 34 00  |a.&Z|..}....}.4.|
00000170  be 94 7d e8 2e 00 cd 18  eb fe 47 52 55 42 20 00  |..}.......GRUB .|
```

```
00000180  47 65 6f 6d 00 48 61 72  64 20 44 69 73 6b 00 52  |Geom.Hard Disk.R|
00000190  65 61 64 00 20 45 72 72  6f 72 0d 0a 00 bb 01 00  |ead. Error......|
000001a0  b4 0e cd 10 ac 3c 00 75  f4 c3 00 00 00 00 00 00  |.....<.u........|
000001b0  00 00 00 00 00 00 00 00  36 87 40 47 00 00 80 04  |........6.@G....|
000001c0  01 04 83 fe c2 ff 00 08  00 00 00 00 20 00 00 fe  |............ ...|
000001d0  c2 ff 83 fe c2 ff 00 08  20 00 00 00 80 0c 00 fe  |........ .......|
000001e0  c2 ff 83 fe c2 ff 00 08  a0 0c 00 00 40 06 00 fe  |............@...|
000001f0  c2 ff 05 fe c2 ff 00 08  e0 12 00 f8 1f 06 55 aa  |..............U.|
00000200

[root@centos8 /data]$echo {a..z} | cut -d" " -f1,2,3 | hexdump -C
00000000  61 20 62 20 63 0a                                 |a b c.|
00000006

[root@centos8 /data]$hexdump -C -n 12 /dev/nvme0n1      # -n 12 只显示前12字节
00000000  eb 63 90 10 8e d0 bc 00  b0 b8 00 00              |.c..........|
0000000c

[root@centos8 /data]$hexdump -C -s 2 -n 10 /dev/nvme0n1  # -s 2 -n 跳过前2字节显示
后10字节
00000002  90 10 8e d0 bc 00 b0 b8  00 00                    |..........|
0000000c
```

od

- 默认将文本内容以8进制传给标准输出，使用选项可规定其他格式；当不指明文件或者文件名为一杠'-'时，读取标准输入。
- 用法

```
od [OPTION]... [FILE]...
od [-abcdfilosx]... [FILE] [[+]OFFSET[.][b]]
od --traditional [OPTION]... [FILE] [[+]OFFSET[.][b] [+][LABEL][.][b]]
```

- od -A x -t x1z -v /file # 以hexdump命令格式输出文件内容

```
[root@centos8 /data]$ od -A x -t x1z -v two_passwd
000000 72 6f 6f 74 3a 78 3a 30 3a 30 3a 72 6f 6f 74 3a  >root:x:0:0:root:<
000010 2f 72 6f 6f 74 3a 2f 62 69 6e 2f 62 61 73 68 0a  >/root:/bin/bash.<
000020 62 69 6e 3a 78 3a 31 3a 31 3a 62 69 6e 3a 2f 62  >bin:x:1:1:bin:/b<
000030 69 6e 3a 2f 73 62 69 6e 2f 6e 6f 6c 6f 67 69 6e  >in:/sbin/nologin<
000040 0a                                               >.<
000041

[root@centos8 /data]$ echo {a..z} | tr -d ' '|od -A x -t x1z
000000 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70  >abcdefghijklmnop<
000010 71 72 73 74 75 76 77 78 79 7a 0a                 >qrstuvwxyz.<
00001b
```

xxd

- xxd命令和hexdump类似，但是其可以使用-r选项反向转换一个标准的16进制ascii码的文件，还原成人类易读的内容。
- 用法

```
xxd -h[elp]
xxd [options] [infile [outfile]]
xxd -r[evert] [options] [infile [outfile]]

详细使用man一下
```

```
[root@centos8 /data]$ echo {a..z} | tr -d ' '|od -A x -t x1z > hex_abc

[root@centos8 /data]$ cat hex_abc
000000 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70  >abcdefghijklmnop<
000010 71 72 73 74 75 76 77 78 79 7a 0a                 >qrstuvwxyz.<
00001b

[root@centos8 /data]$ xxd -r hex_abc
abcdefghijklmnopqrstuvwxyz
```

- xxd和hexdump默认显示的区别

```
[root@centos8 /data]$ xxd two_passwd
00000000: 726f 6f74 3a78 3a30 3a30 3a72 6f6f 743a  root:x:0:0:root:
00000010: 2f72 6f6f 743a 2f62 696e 2f62 6173 680a  /root:/bin/bash.
00000020: 6269 6e3a 783a 313a 313a 6269 6e3a 2f62  bin:x:1:1:bin:/b
00000030: 696e 3a2f 7362 696e 2f6e 6f6c 6f67 696e  in:/sbin/nologin
00000040: 0a                                       .
[root@centos8 /data]$ hexdump two_passwd
0000000 6f72 746f 783a 303a 303a 723a 6f6f 3a74
0000010 722f 6f6f 3a74 622f 6e69 622f 7361 0a68
0000020 6962 3a6e 3a78 3a31 3a31 6962 3a6e 622f
0000030 6e69 2f3a 6273 6e69 6e2f 6c6f 676f 6e69
0000040 000a
0000041
```

3.按行截取文件的工具

head

- 使用head输出文件内容的前面部分，默认打印前10行；当不指明文件或者文件名为一杠'-'时，读取标准输入。
- 用法

```
head [OPTION]... [FILE]...
    -v # 在第一行打印文件名，使用管道传内容给head时文件名显示为：standard input
    -c bytes # 指定显示文件内容前bytes字节内容
    -n lines # 指定显示文件内容前lines行
```

tail

- 使用tail输出文件内容的后面部分，默认打印后10行；当不指明文件或者文件名为一杠'-'时，读取标准输入。
- 用法

```
tail [OPTION]... [FILE]...
    -f, --follow[={name|descriptor}]  # 动态跟踪文件新增的内容
    -c bytes            # 获取文件后bytes字节内容
    -n lines            # 获取文件后lines行
    -F                  # 跟踪文件名，相当于--follow=name --retry
```

- tailf命令：类似tail –f，当文件不增长时并不访问文件，更节省系统资源

4.按列抽取文本的工具

cut

- 使用cut命令来抽取某文件内容的某一列
- 用法

```
cut [OPTION]... [FILE]...
    -d DELIMITER: 指明分隔符，默认tab
    -f FILEDS:
     #: 第#个字段
     #,#[,#]: 离散的多个字段，例如1,3,6
     #-#: 连续的多个字段，例如1-6
                混合使用：1-3,7
    -c 按字符切割
    --output-delimiter=STRING指定输出分隔符
```

paste

- 使用paste命令合并两个文件同行号的列到一行；默认使用tab键分隔
- 用法

```
     paste [OPTION]... [FILE]...
        -d 分隔符：指定分隔符，默认用TAB
        -s : 所有行合成一行显示
```

```
        示例：
            paste f1 f2
            paste -s f1 f2
            paste -s file         # 将某个文件显示为单行，所有内容作一行显示；原来的行与行之间使用tab键分隔，也可指定新的分隔符
            如下面的示例：
[root@centos6 /data]$ cat -A testff
1$
2$
3$
4$
hello$
hi$
[root@centos6 /data]$ paste -s testff > testffgg
[root@centos6 /data]$ cat -A testffgg
1^I2^I3^I4^Ihello^Ihi$
[root@centos6 /data]$ paste -s -d: testff > testffgg
[root@centos6 /data]$ cat -A testffgg
1:2:3:4:hello:hi$
```

5.排序和统计文本内容

sort

- sort命令会把排序过的文本显示在STDOUT，不改变原始文件
- 用法

```
sort [options] file(s)
常用选项
    -r 执行反方向（由上至下）整理
    -R 随机排序
    -n 执行按数字大小整理
    -f 选项忽略（fold）字符串中的字符大小写
    -u 选项（独特，unique）删除输出中的重复行
    -t C 选项使用C做为字段界定符
    -k # 选项按照使用C字符分隔的第#列来整理能够使用多次
例子：
[root@centos8 /data]$ sort -t : -k3 -n /etc/passwd | head -n5  ## 指定用:作分隔符；取第三列按数值大小正向排序
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
[root@centos8 /data]$ sort -t : -k3 -n /etc/passwd | head -n5 | cut -d: -f3
0                          ## 指定用:作分隔符；取第三列按数值大小正向排序，取出第三列
1
2
3
4
```

wc

- 使用wc统计文件的行总数、单词总数、字节总数和字符总数；当不指明文件或者文件名为一杠'-'时，读取标准输入。
- 用法

```
wc [OPTION]... [FILE]...
wc [OPTION]... --files0-from=F
    wc story.txt
    39     237    1901 story.txt
    行数   字数   字节数
常用选项
    -l 只计数行数
    -w 只计数单词总数
    -c 只计数字节总数
    -m 只计数字符总数
    -L 显示文件中最长行的长度
例子:
[root@centos8 /data]$wc test
  64  126 2935 test
[root@centos8 /data]$wc -l test
64 test
[root@centos8 /data]$wc -w test
126 test
[root@centos8 /data]$wc -c test
2935 test
[root@centos8 /data]$wc -m test
2935 test
[root@centos8 /data]$wc -L test
99 test
```

uniq

- 使用uniq命令，从标准输入中删除前后相接的重复行，只显示一行

```
uniq [OPTION]... [FILE]...
    -c: 显示每行重复出现的次数
    -d: 仅显示重复过的行
    -u: 仅显示不曾重复的行
        注：连续且完全相同方为重复
```

- uniq常和sort 命令一起配合使用：

```
[root@centos7 /data/test]$echo -e "1 1 2 3 4 4 5 7 7" | tr " " "\n" |sort -nr
7
7
5
4
```

```
4
3
2
1
1
[root@centos7 /data/test]$echo -e "1 1 2 3 4 4 5 7 7" | tr " " "\n" |sort -nr|
uniq
7
5
4
3
2
1
[root@centos7 /data/test]$echo -e "1 1 2 3 4 4 5 7 7" | tr " " "\n" |sort -nr|
uniq -c
      2 7
      1 5
      2 4
      1 3
      1 2
      2 1
```

6.按关键字搜索抽取文本内容

grep

- 本文第二部分------Linux文本处理三剑客之一:grep

7.比较文件和恢复文件

diff

- diff命令用来逐行比较两个文件的区别
- 用法

```
diff [OPTION]... FILES
    -y # 分两列显示比较结果
    -b # 忽略空格个数的改变
    -q # 只有文件不同才报告
    -s # 只有两个文件相同才报告
    -u, -U NUM, --unified[=NUM]  # 生成patch可以使用的"统一的(unified)"diff文件

例子:
[root@centos8 /data/diff_patch]$ diff origin modified.orig -y | head -v
==> standard input <==
root:x:0:0:root:/root:/bin/bash
root:x:0:0:root:/root:/bin/bash
                                                              >
bin:x:1:1:bin:/bin:/sbin/nologin
bin:x:1:1:bin:/bin:/sbin/nologin
                                                              >
```

```
                                                                > hello
                                                                >

daemon:x:2:2:daemon:/sbin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
sync:x:5:0:sync:/sbin:/bin/sync
```

patch

- 使用patch命令来使用diff文件恢复改变过的文件
- 用法：

```
patch [options] [originalfile [patchfile]]
    -b # 在对改变过的原文件施加diff文件前，备份原文件
```

- diff和patch搭配还原修改后的某文件

```
[root@centos8 /data/diff_patch]$ll
total 16
-rw-r--r--. 1 root root   18 Oct  8 21:10 modified
-rw-r--r--. 1 root root    7 Oct  8 21:09 origin
[root@centos8 /data/diff_patch]$cat origin
origin
[root@centos8 /data/diff_patch]$cat modified
origin
 modified

## 第一步使用diff -u 生成diff文件
[root@centos8 /data/diff_patch]$diff -u origin modified > diff_patch

[root@centos8 /data/diff_patch]$ll
total 20
-rw-r--r--. 1 root root  138 Oct  8 21:11 diff_patch
-rw-r--r--. 1 root root   18 Oct  8 21:10 modified
-rw-r--r--. 1 root root    7 Oct  8 21:09 origin
## 查看diff文件
[root@centos8 /data/diff_patch]$cat diff_patch
--- origin      2019-10-08 21:09:58.000679482 +0800
+++ modified    2019-10-08 21:10:38.776677191 +0800
@@ -1 +1,2 @@
-origin
+origin
+ modified
```

```
## 第二步，使用patch -b 对更改的文件（modified）参照diff文件还原；备份modified文件为
modified.org
[root@centos8 /data/diff_patch]$patch -b modified diff_patch

[root@centos8 /data/diff_patch]$ll
total 20
-rw-r--r--. 1 root root  138 Oct  8 21:11 diff_patch
-rw-r--r--. 1 root root    7 Oct  8 21:12 modified
-rw-r--r--. 1 root root   18 Oct  8 21:10 modified.orig    # 原来的modified文件
-rw-r--r--. 1 root root    7 Oct  8 21:09 origin

[root@centos8 /data/diff_patch]$cat modified.orig
origin
 modified
[root@centos8 /data/diff_patch]$cat modified               # modified文件被还原为
origin文件
origin
```

8.练习

```
1、找出ifconfig “网卡名” 命令结果中本机的IPv4地址
    ifconfig | head -n2 | tail -n1 | tr -s " " | cut -d" " -f3 # centos7,8
    ifconfig | head -n2 | tail -n1 | tr -s " " | cut -d" " -f3 | cut -d: -f2 #
centos6
    ifconfig | grep -Eo "([0-9]{,3}\.){3}[0-9]{,3}" | head -n1 # centos6,7,8
    ifconfig | grep -Eo '\<(([1-9]?[0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([1-
9]?[0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\>' | head -n1   # 精确

2、查出分区空间使用率的最大百分比值
    df -h | grep -e /dev/sd -e /dev/nvme | tr -s " " | cut -d" " -f5 | sort -nr  |
head -n3

3、查出用户UID最大值的用户名、UID及shell类型
    getent passwd | sort -t : -k 3 -nr | head -n1 | cut -d: -f1,3,7
    getent passwd | sort -t : -k 3 -nr | cut -d: -f1,3,7 | head -n1

4、查出/tmp的权限，以数字方式显示
    stat /tmp/ | head -n4 | tail -n1 | cut -d \( -f2 | cut -d/ -f1
    stat /tmp/ | grep -E "(\([0-7]{4})" | grep -Eo [0-9]{4} |head -n1

5、统计当前连接本机的每个远程主机IP的连接数，并按从大到小排序
    ss -tun |grep ESTAB | tr -s " " | cut -d" " -f6 | cut -d: -f1 | sort | uniq -c
| sort -nr
    tr -s " " ":" < ss.log | cut -d: -f6 | sort | uniq -c| sort -nr
    ss -tun |grep ESTAB | tr -s " " ":" | cut -d: -f7 | sort | uniq -c | sort -nr
```

二.Linux文本处理三剑客之一:grep

grep为linux中有名的文本过滤工具

- grep 与sed、awk并称为linux下的文本处理三剑客

> sed：stream editor，文本编辑工具
> awk：Linux上的实现gawk，文本报告生成器

- 用法

```
"PATTERN" 为基本正则表达式
grep [OPTIONS] PATTERN [FILE...]
grep [OPTIONS] -e PATTERN ... [FILE...]
grep [OPTIONS] -f FILE ... [FILE...]
```

- grep默认搜索每个文件中和"模式"匹配的行并显示匹配到的行，模式由正则表达式指定；如果模式 PATTERN 后面没有跟文件而是一杠"-",则使用指定的模式匹配标准输入。

- egrep和fgrep为grep的变体

```
grep -E 等同于 egrep
grep -F 等同于 fgrep
fgrep 不支持正则表达式
```

- grep选项

```
"PATTERN" 为基本正则表达式
grep [OPTIONS] PATTERN [FILE...]
grep [OPTIONS] -e PATTERN ... [FILE...]
grep [OPTIONS] -f FILE ... [FILE...]
    -E   # 使用扩展正则表达式
    -o   # 只显示匹配到的字符串而不显示其所在的行的内容
```

```
    -v   # 显示未匹配到的行
    -i   # 忽略大小写
    -q   # 静默模式，无论是否匹配到都不打印标准输出
    -f file,--file=FILE # 从文件file中读取正则表达式
    -e # 使用多个操作来匹配模式
        grep -e root -e bash /etc/passwd # 匹配包含root和bash的行
    -e # 匹配单词
    -c, --count # 匹配到的行计算总数
    -s, --no-messages # 对不存在或则不可读文件的错误不打印
    -n, --line-number # 对匹配到的行加行号
    -A NUM, --after-context=NUM # 匹配到某模式时不仅打印匹配到的行还打印其后的NUM行
    -B NUM, --before-context=NUM # 匹配到某模式时不仅打印匹配到的行还打印其前的NUM行
    -C NUM, -NUM, --context=NUM # 匹配到某模式时不仅打印匹配到的行还打印其前和其后的的
NUM行
    --color=auto # 对匹配到的文本着色显示
    -m  #  匹配#次后停止
```

三.基本正则表达式

- 正则表达式(REGEXP:Regular Expressions)是由一类特殊字符及文本字符所编写的模式,其中有些字符（元字符）不表示字符字面意义，而表示控制或通配的功能.
- 很多程都支持正则表达式：vim, less,grep,sed,awk, nginx,varnish 等.
- 正则表达式分两类：

> 基本正则表达式：BRE，grep，vim 扩展正则表达式：ERE，grep -E, egrep，nginx

- 正则表达式引擎-采用不同算法，检查处理正则表达式的软件模块 如：PCRE（Perl Compatible Regular Expressions）
- 元字符分类：字符匹配、匹配次数、位置锚定、分组
- man帮助：man 7 regex
- 基本正则表达式元字符
- 字符匹配

```
.           匹配任意单个字符
[]          匹配指定范围内的任意单个字符，示例：[wang]   [0-9]    [a-z]   [a-zA-Z]
[^]         匹配指定范围外的任意单个字符
[:alnum:] 字母和数字
[:alpha:] 代表任何英文大小写字符，亦即 A-Z, a-z
[:lower:] 小写字母    [:upper:] 大写字母
[:blank:] 空白字符（空格和制表符）
[:space:] 水平和垂直的空白字符（比[:blank:]包含的范围广）
[:cntrl:] 不可打印的控制字符（退格、删除、警铃...）
[:digit:] 十进制数字 [:xdigit:]十六进制数字
[:graph:] 可打印的非空白字符
[:print:] 可打印字符
[:punct:] 标点符号
```

- 匹配次数
- 用在要指定次数的字符后面，用于指定前面的字符要出现的次数

```
*  匹配前面的字符任意次，包括0次
   贪婪模式：尽可能长的匹配
.* 任意长度的任意字符
\? 匹配其前面的字符0或1次
\+ 匹配其前面的字符至少1次
\{n\} 匹配前面的字符n次
\{m,n\} 匹配前面的字符至少m次，至多n次
\{,n\} 匹配前面的字符至多n次
\{n,\} 匹配前面的字符至少n次
```

- 位置锚定

- 位置锚定：定位出现的位置

```
^ 行首锚定，用于模式的最左侧
$ 行尾锚定，用于模式的最右侧
^PATTERN$  用于模式匹配整行
^$  空行
^[[:space:]]*$  空白行
\< 或 \b 词首锚定，用于单词模式的左侧
\> 或 \b 词尾锚定，用于单词模式的右侧
\<PATTERN\> 匹配整个单词
```

- 后向引用

- 后向引用：引用前面的分组括号中的模式所匹配字符，而非模式本身

- 分组：() 将一个或多个字符捆绑在一起，当作一个整体处理如：(root)+

- 分组括号中的模式匹配到的内容会被正则表达式引擎记录于内部的变量中，这些 变量的命名方式为: \1, \2, \3, ... ;\1 表示从左侧起第一个左括号以及与之匹配右括号之间的模式所匹配到的字符

```
示例：  \(string1\(string2\)\)

      \1 : string1\(string2\)

      \2 : string2
```

```
或者：\|
a\|b
     a或b
C\|cat
```

```
    C或cat
\(C\|c\)at
    Cat或cat
```

练习

```
    grep "^[sS].*" /proc/meminfo
    grep "^[s]\|^[S].*" /proc/meminfo
    grep "^s\|^S.*" /proc/meminfo

2、显示/etc/passwd文件中不以/bin/bash结尾的行
    grep -v "\<bin/bash\>$" /etc/passwd
    grep -v "\bbin/bash\b$" /etc/passwd

3、显示用户rpc默认的shell程序
    getent passwd | grep -w "rpc" | cut -d: -f1,7
    grep -w "rpc" /etc/passwd | cut -d: -f1,7

4、找出/etc/passwd中的两位或三位数
    grep --color=auto -o "[0-9]\{2,3\}" /etc/passwd

5、显示CentOS7的/etc/grub2.cfg文件中，至少以一个空白字符开头的且后面有非
空白字符的行
    grep "^[[:space:]].*" /etc/grub2.cfg
    grep "^ .*" /etc/grub2.cfg

6、找出“netstat -tan”命令结果中以LISTEN后跟任意多个空白字符结尾的行
    netstat -tan | grep "LISTEN \+"

7、显示CentOS7上所有UID小于1000以内的用户名和UID
    grep --color=auto -v ".*[0-9]\{4\}.*" /etc/passwd | cut -d: -f1,3

8、添加用户bash、testbash、basher、sh、nologin(其shell为/sbin/nologin),找
出/etc/passwd用户名和shell同名的行
    grep "^\(\<.*\>\).*/\1$" /etc/passwd

9、利用df和grep，取出磁盘各分区利用率，并从大到小排序
    df -h | grep -o "\<[0-9]\{,3\}%" | sort -nr
    df -h | grep -o "\<[0-9]\{,3\}%" | sort -nr | cut -d% -f1
```

四.扩展正则表达式

- 扩展正则表达式与基本正则表达式的对比：

基本正则表达式与	扩展正则表达式与
\{\}	{}

基本正则表达式与	扩展正则表达式与
\(\)	()
\b;\<;\>	\b;\<;\>
\+	+
\?	?
\1, \2, ...	\1, \2, ...
|	|

练习

1、显示三个用户root、mage、wang的uid和默认shell

```
getent passwd | grep -w -e ^root -e ^mage -e ^wang | cut -d: -f3,7
```

2、找出/etc/rc.d/init.d/functions文件中行首为某单词(包括下划线)后面跟一个小括号的行

```
grep "^\(.*\)_\?\(.*\)()"  /etc/rc.d/init.d/functions
```

3、使用egrep取出/etc/rc.d/init.d/functions中其基名

```
echo /etc/rc.d/init.d/functions|egrep -o "[^/]+/?$"
```

4、使用egrep取出上面路径的目录名

```
echo /etc/rc.d/init.d/functions/ |egrep -o "/.*[^/]"|egrep -o "/.*/"
```

5、统计last命令中以root登录的每个主机ip地址登录次数

```
last | grep root | egrep "\b(([1-9]?[0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}
([1-9]?[0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\b" | tr -s " " | cut -d" " -f3 | uniq
-c | sort -nr
```

6、利用扩展正则表达式分别表示0-9、10-99、100-199、200-249、250-255

```
[0-9]
[1-9][0-9]
1[0-9]{2}
2[0-4][0-9]
25[0-5]
```

7、显示ifconfig命令结果中所有ipv4地址

```
ifconfig | egrep "\b(([1-9]?[0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([1-9]?
[0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\b"
```

8、将此字符串：welcome to magedu linux 中的每个字符去重并排序，重复次数多的排到前面

```
echo "welcome to  magedu linux" | grep -o . | sort | uniq -c | sort -nr
```