一、正则表达式（上）

P1 概述

什么是正则表达式

在一个文档中主要是过滤数据和处理数据

P2 基本正则

基本正则符号

正则符号	描述
abc	匹配abc
^	匹配开头
$	匹配结尾
[集合]	匹配集合中的任意单个字符
[^集合]	对集合取反
.	匹配任意单个字符
*	匹配前一个字符任意次（包含0次）
.*	匹配任意
\{n,m\}	匹配前一个字符n到m次
\{n,\}	匹配前一个字符至少n次
\{n\}	匹配前一个字符n次

基本正则的使用

```
[root@svr7 ~]# grep root /etc/passwd                #查找包含 root 的行
[root@svr7 ~]# grep ^root /etc/passwd               #查找以 root 开头的行
[root@svr7 ~]# grep bash$ /etc/passwd               #查找以 bash 结尾的行
[root@svr7 ~]# grep "[abc]" /etc/passwd             #查找包含 a 或者 b 或者 c 的行
[root@svr7 ~]# grep "[^abc]" /etc/passwd            #查找不包含 a 或者 b 或者 c 的其他内容
[root@svr7 ~]# grep . /etc/passwd                   #查找任意单个字符
[root@svr7 ~]# grep r.*t /etc/passwd                #查找以 r 开头以 t 结尾的
[root@svr7 ~]# grep "[0-9]*"    /etc/passwd         #查找包含数字的，*代表任意次
[root@svr7 ~]# grep "[0-9]\{3,4\}" /etc/passwd      #查找包含数字 3-4 次的
[root@svr7 ~]# grep "[0-9]\{3\}" /etc/passwd        #查找包含 3 位数的
```

P2 扩展正则

扩展正则符号

正则符号	描述
+	匹配前面的字符至少一次
?	匹配前面的字符0或1次
()	组合与保留
|	或者
{n,m}	匹配前面的字符n到m次
{n,}	匹配前面的字符至少n次
{n}	匹配前面的字符n次

扩展正则的使用

```
[root@svr7 ~]# grep -E "0{2,3}" /etc/passwd          #查找 0 出现 2-3 次
[root@svr7 ~]# grep -E "[a-z]+"    /etc/passwd       #查找 a-z 等字母至少出现一次
[root@svr7 ~]# grep -E "s?bin"    /etc/passwd        #查找 sbin 或者 bin(?匹配前面的 s 字符
0-1 次)
[root@svr7 ~]# grep -E "(root|daemon)" /etc/passwd      #查找 root 或者 daemon
[root@svr7 ~]# echo "ababab" | grep ab                  #查找 ab
ababab
[root@svr7 ~]# echo "ababab" | grep -E "(ab)"           #查找 ab
[root@svr7 ~]# echo "ababab" | grep -E "(ab){2}"        #将 ab 组合，匹配两次
```

P3 Perl 兼容的正则

Perl 兼容的正则列表

正则符号	描述
\b	匹配单词边界
\w	匹配字符数字下划线
\W	和\w相反
\s	匹配空白
\d	匹配数字
\d+	匹配多个数字
\D	匹配非数字

Perl 兼容正则的使用

```
[root@svr7 ~]# grep -P "bin" /etc/passwd       #匹配包含 bin 的行，只要包含 bin 字符的都出现
[root@svr7 ~]# grep -P "\bbin\b" /etc/passwd    # \b 单词边界，b 前面不能有内容，n 后面也
```

不能有内容，只匹配 bin

```
[root@svr7 ~]# grep -P "\w"    /etc/passwd          #查找字母数字下划线
[root@svr7 ~]# grep -P "\W"    /etc/passwd          #查找不是字母数字下划线部分
[root@svr7 ~]# grep -P "\s"    /etc/passwd          #查找空白，空格，tab 键都算
[root@svr7 ~]# grep -P "\d"    /etc/passwd          #查找数字
[root@svr7 ~]# grep -P "\D"    /etc/passwd          #查找非数字
```

二、正则表达式（下）

P1 基础练习

grep 语法格式

用法: grep [选项] 匹配模式 [文件]..

常用选项:

-i 忽略大小写

-V 取反匹配

-w 匹配单词，和正则里面的\b 是一样的

-q 静默匹配，不将结果显示在屏幕（无论结果是否匹配成功）

正则基本练习

```
[root@svr7 ~]# mkdir /root/shell/day04
[root@svr7 ~]# cd /root/shell/day04
[root@svr7 day04]# ls
python.txt
```

从《python.txt》文件中过滤如下数据:，python.txt 文件是提前给好的

过滤包含 the 的行

```
[root@svr7 day04]# grep the python.txt
```

不区分大小写过滤包含 the 的行

```
[root@svr7 day04]# grep -i the python.txt
```

过滤不包含 the 的行

```
[root@svr7 day04]# grep -v the python.txt
```

过滤包含数字的行

```
[root@svr7 day04]# grep "[0-9]" python.txt
[root@svr7 day04]# grep -P "\d" python.txt        #使用 Perl 正则过滤包含数字的行
```

过滤包含 bet 或者 better 的行

```
[root@svr7 day04]# grep -E "(bet|better)" python.txt
```

过滤包含 2 个字母 o 的行

```
[root@svr7 day04]# grep "oo" python.txt
[root@svr7 day04]# grep "o\{2\}" python.txt
[root@svr7 day04]# grep -E "o{2}" python.txt
```

过滤包含 1-2 个字母 o 的行

```
[root@svr7 day04]# grep -E "o{1,2}" python.txt
```

过滤不包含字母 o 的行

```
[root@svr7 day04]# grep -v "o" python.txt
```

过滤大写字母开头的行
[root@svr7 day04]# grep "^[A-Z]" python.txt
过滤小写字母开头的行
[root@svr7 day04]# grep "^[a-z]" python.txt #因为 python.txt 没有以小写字母开头的行，所以过滤结果为空
过滤 ou 前面不是 th 的行
[root@svr7 day04]# grep -E "[^(th)]ou" python.txt
过滤不以标点符号结束的行
[root@svr7 day04]# grep "[^.]$" python.txt
过滤以.结尾的行
[root@svr7 day04]# grep "\.$" python.txt #\为转译
过滤空白行
[root@svr7 day04]# grep "^$" python.txt
过滤以数字开始的行
[root@svr7 day04]# grep "^[0-9]" python.txt
过滤包含 2 个以上 z 的行
[root@svr7 day04]# grep -E "z{2,}" python.txt
过滤所有字母
[root@svr7 day04]# grep "[a-zA-Z]" python.txt
过滤所有标点符号
[root@svr7 day04]# grep -P "\W" python.txt

三、sed 基础

P1 什么是 sed

sed 非交互式修改文档，逐行处理，可以对文本进行增删改查等操作
sed 语法格式
语法：命令 | sed 选项 (定位符)指令
sed 选项 (定位符)指令 文件名 #(定位符)指令 想对文件的哪一行进行操作
选项
-n 屏蔽默认输出
-r 支持扩展正则
-i 写入文件

P2 数据定位

定位符：行号定位
sed 可以使用行号来定位自己需要修改的数据内容
[root@svr7 day04]# sed "2p" /etc/hosts #不加-n 选项，默认全文打印，第二行打印了两边，因为有一遍是人为要求打印的
[root@svr7 day04]# sed -n "2p" /etc/hosts # -n 屏蔽默认输出，打印第二行内容
[root@svr7 day04]# sed -n "3p" /etc/passwd #打印第三行
[root@svr7 day04]# sed -n "1,3p" /etc/passwd #打印一到三行

```
[root@svr7 day04]# sed -n "1~2p" /etc/passwd   #输出（奇数行）第 1 行开始步长为 2
[root@svr7 day04]# sed -n "2~2p" /etc/passwd   #输出（偶数行）第 2 行开始步长为 2
[root@svr7 day04]# sed -n "2,+3p" /etc/passwd  #输出第 2 行,以及后面的 3 行
```

P3 正则定位

sed 可以使用正则匹配需要的数据，然后在编辑对应的内容，sed 里面在使用正则是需要两个/ / 括起来

过滤 root 开头的行，使用 sed 打印

```
[root@svr7 day04]# grep "^root" /etc/passwd
[root@svr7 day04]# sed -n "/^root/p" /etc/passwd
```

过滤包含三个数字的行

```
[root@svr7 day04]# grep    -E "[0-9]{3}" /etc/passwd
[root@svr7 day04]# sed -rn "/[0-9]{3}/p" /etc/passwd        #使用扩展正则加 r 选项
```