shell day 04

=============================================

**一，字符串处理**

**字符串删除**

a=abcdabcd //定义变量为素材

echo ${a#\*c} //从左往右删除到第1个c

echo ${a##\*c} //从左往右删除到最后1个c

echo ${a%b\*} //从右往左删除到第1个b

echo ${a%%b\*} //从右往左删除到最后1个b

**定义变量初值(备用)**

**${变量名:-初值}**

#!/bin/bash

read -p "请输入用户名:" u

[ -z $u ] && echo "必须输入用户名！" && exit

useradd $u

read -p "请输入密码(默认123456):" n

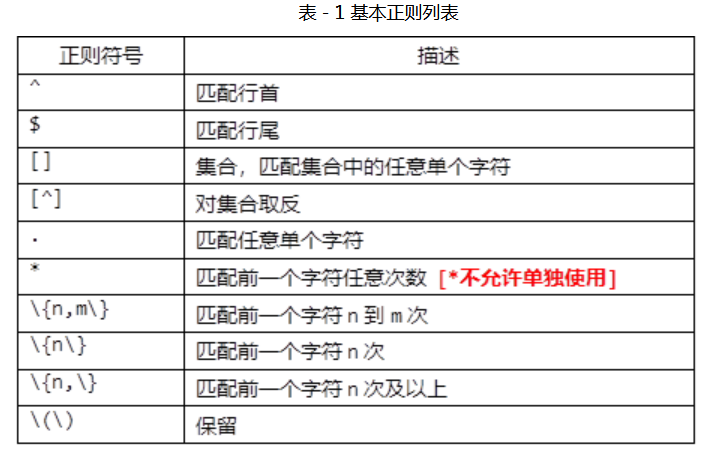
echo ${n:-123456} | passwd --stdin $u

------------------------------------------------------

**二，正则表达式，可以使用若干符号配合某工具对字符串进行**

**增删改查操作**

head -5 /etc/passwd > user //准备素材



grep ^root user //找以root开头的行

grep bash$ user //找以bash结尾的行

grep ^$ user //找空行

grep -v ^$ user //显示除了空行的内容

grep "[root]" user //找r、o、t任意一个字符

grep "[rot]" user //效果同上

grep "[^rot]" user //显示r或o或t以外的内容

grep "[0123456789]" user //找所有数字

grep "[0-9]" user //效果同上

grep "[^0-9]" user //显示数字以外内容

grep "[a-z]" user //找所有小写字母

grep "[A-Z]" user //找所有大写字母

grep "[a-Z]" user //找所有字母

grep "[^0-9a-Z]" user //找所有符号

grep "." user //找任意单个字符，文档中每个字符都可

以理解为任意字符

grep "r..t" user //找rt之间有2个任意字符的行

grep "r.t" user //找rt之间有1个任意字符的行，没有匹

配内容，就无输出

grep "\*" user //错误用法，\*号是匹配前一个字符任意

次，不能单独使用

grep "ro\*t" user //找rt，中间的o有没有都行，有几次都行

grep ".\*" user //找任意，包括空行 .与\*的组合在正则中相当

于通配符的效果

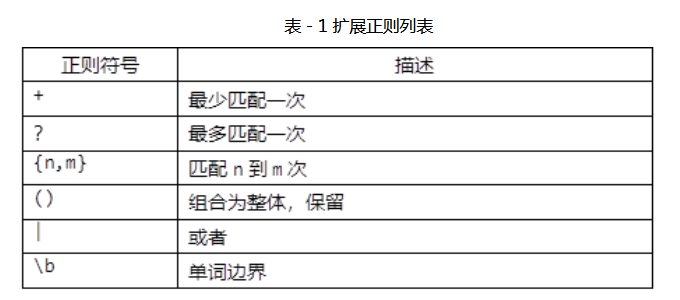
grep "ro\{1,2\}t" user //找rt，中间的o可以有1~2个

grep "ro\{2,6\}t" user //找rt，中间的o可以有2~6个

grep "ro\{1,\}t" user //找rt，中间的o可以有1个以及1个以上

grep "ro\{3\}t" user //找rt，中间的o必须只有有3个

-------------------------------------------------------------------------



以上命令均可以加-E选项并且去掉所有\，改成扩展正则的用法，比如

grep "ro\{1,\}t" user可以改成 grep -E "ro{1,}t" user

或者egrep "ro{1,}t" user

grep "ro\{1,\}t" user //使用基本正则找o出现1次以及1次以上

egrep "ro{1,}t" user //使用扩展正则，效果同上，比较精简

egrep "ro+t" user //使用扩展正则，效果同上，最精简

grep "roo\{0,1\}t" user //使用基本正则找第二个o出现0~1次

egrep "roo{0,1}t" user //使用扩展正则，效果同上，比较精简

egrep "roo?t" user //使用扩展正则，效果同上，最精简

egrep "(0:){2}" user //找连续的2个0: 小括号的作用是将字符组合为一个整体

egrep "root|bin" user //找有root或者bin的行

egrep "the\b" abc.txt //在abc.txt文件中找the，右边不允许出

现数字、字母、下划线

egrep "\bthe\b" abc.txt //两边都不允许出现数字、字母、下划线

egrep "\<the\>" abc.txt //效果同上

----------------------------------------------------

**三，sed 流式编辑器**

**可以对文档进行非交互式增删改查，逐行处理**

**用法：**

**1，前置指令 | sed 选项 条件 指令**

**2，sed 选项 条件 指令 被处理文档**

**选项 -n 屏蔽默认输出 -r 支持扩展正则 -i 修改源文件**

**指令 p 输出 d 删除 s 替换**

**条件 行号 /字符串/**

sed -n 'p' user //输出所有行

sed -n '1p' user //输出第1行

sed -n '2p' user //输出第2行

sed -n '3p' user //输出第3行

sed -n '2,4p' user //输出2~4行

sed -n '2p;4p' user //输出第2行与第4行

sed -n '3,+1p' user //输出第3行以及后面1行

sed -n '/^root/p' user //输出以root开头的行

sed -n '/root/p' user //输出包含root的行

sed -nr '/^root|^bin/p' user //输出以root开头的行

或bin开头的行，|是扩展正则，需要r选项

sed -n '1!p' user //输出除了第1行的内容，!是取反

sed -n '$p' user //输出最后一行

sed -n '=' user //输出行号，如果是$=就是最后一行的行号

以上操作，如果去掉-n,在将p指令改成d指令就是删除

输出所有行 sed -n 'p' abc.txt

输出第4行 sed -n '4p' abc.txt

输出第4~7行 sed -n '4,7p' abc.txt

输出以bin开头的行 sed -n '/^bin/p' abc.txt

输出文件的总行数 sed -n '$=' abc.txt

删除第3~5行 sed '3,5d' abc.txt

删除所有包含xml的行 sed '/xml/d' abc.txt

删除不包含xml的行 sed '/xml/!d' abc.txt

删除以install开头的行 sed '/^install/d' abc.txt

删除文件的最后一行 sed '$d' abc.txt

删除所有空行 sed '/^$/d' abc.txt

替换

sed 's/2017/6666/' shu.txt //把所有行的第1个2017替换成6666

sed 's/2017/6666/2' shu.txt //把所有行的第2个2017替换成6666

sed '1s/2017/6666/' shu.txt //把第1行的第1个2017替换成6666

sed '3s/2017/6666/3' shu.txt //把第3行的第3个2017替换成6666

sed 's/2017/6666/g' shu.txt //所有行的所有个2017都替换

sed '/2024/s/2017/6666/g' shu.txt //找含有2024的行，将里面的

所有2017替换成6666

如果想把 /bin/bash 替换成 /sbin/sh 怎么操作？

sed -i '1s/bin/sbin/' user //传统方法可以一个一个换，先换一个

sed -i '1s/bash/sh/' user //再换一个

如果想一起一步替换：

sed 's//bin/bash//sbin/sh/' user //直接替换，报错

sed 's/\/bin\/bash/\/sbin\/sh/' user //使用转义符号可以成功，但

不方便

sed 's!/bin/bash!/sbin/sh!' user //最佳方案，更改s的替换符

sed 's(/bin/bash(/sbin/sh(' user //替换符号可以用所有数字键上的

----------------------------------------------------------------------

**编写脚本，搭建httpd服务，用82号端口开启服务**

#!/bin/bash

setenforce 0 //关闭selinux

yum -y install httpd &> /dev/null //安装网站

echo "sed-test~~~" > /var/www/html/index.html //定义默认页

sed -i '/^Listen 80/s/0/2/' /etc/httpd/conf/httpd.conf //修改

配置文件，将监听端口修改为82

systemctl restart httpd //开服务

systemctl enable httpd //设置开机自启

然后运行脚本

curl 192.168.2.5:82 //脚本运行之后，测试82端口看到页面即可

sed-test~~~

netstat -ntulp | grep httpd //检查服务的端口是否为82

----------------------------------------------------------------------------

**编写脚本，找到系统使用bash的账户名，然后按照**

**“用户名 --> 密码” 的格式存储在一个文件中**

#!/bin/bash

u=$(sed -n '/bash$/s/:.\*//p' /etc/passwd) //找到passwd文档

中以bash结尾的行，然后将行中冒号以及冒号后面内容都删除

此处的p代表仅仅显示s替换成功的行，最后赋值给u

for i in $u //将那些用bash的账户名交给for循环

do

n=$(grep $i: /etc/shadow) //用每个账户名去shadow中

找对应信息

n=${n#\*:} //掐头，从左往右删除到第1个冒号

n=${n%%:\*} //去尾，从右往左删除到最后一个冒号

经过上述步骤，n就是最终要的密码了

echo "$i --> $n" //按格式喊出，如果要存到文件中就用追加重定向

done

-----------------------------------------------------------------------------------

运维课程需要的环境

情况一：如果电脑是Windows的话：

创建虚拟机，用最小化方式安装，按要求配置好ip，同网段之间要能互通，配置好yum，修改主机名

proxy 192.168.2.5(vmnet2) 192.168.4.5(vmnet4) //必须

web1 192.168.2.100(vmnet2) //必须

web2 192.168.2.200(vmnet2) //必须

client 192.168.4.10(vmnet4) //可选，主要作为测试，如果没有该主机将来可以用真机测试

真机要安装火狐浏览器

上述需求可以参考视频：

链接：https://pan.baidu.com/s/1d8H9bqsJ83eQOSsvNeYWOA

提取码：11ie

情况二：如果电脑是Linux的话

在码云找 operation环境准备.docx 这个文档，里面有安装过程

----------------------------------------------------------------

练习：

1 简述grep工具的-q选项的含义（egrep同样适用）。

2 正则表达式中的+、？、\*分别表示什么含义？

3 如何编写正则表达式匹配11位的手机号？

4 简述sed条件的作用及表示方式。

参考答案：

1 简述grep工具的-q选项的含义（egrep同样适用）。

选项-q的作用是静默、无任何输出，效果类似于正常的grep操作添加了&> /dev/null来屏蔽输出

2 正则表达式中的+、？、\*分别表示什么含义？

这三个字符用来限制前面的关键词的匹配次数，含义分别如下：

+：最少匹配一次，比如a+可匹配a、aa、aaa等

？：最多匹配一次，比如a?可匹配零个或一个a

\*：匹配任意多次，比如a\*可匹配零个或任意多个连续的a

3 如何编写正则表达式匹配11位的手机号？

准备测试文件：

[root@svr5 ~]# cat tel.txt

01012315

137012345678

13401234567

10086

18966677788

提取包含11位手机号的行：

[root@svr5 ~]# egrep '^1[0-9]{10}$' tel.txt //这里^代表以数字1开头，后面有10个数字作为结尾

13401234567

18966677788

4 简述sed条件的作用及表示方式。

作用：条件控制sed需要处理文本的范围，不加则逐行处理所有行

表示方式：条件可以使用行号或正则表达式