**一、date命令**

1、date +%Y-%m-%d:%H:%M:%S //年-月-日：时：分：秒

2018-04-02:21:40:49



2、date +%F



3、date +%s //时间戳



4、date +%F:%T



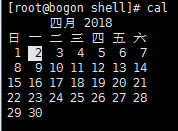
5、date +%w //星期几



6、date +%W //本年度的第几周



7、cal//日历



8、根据时间戳来定位具体时间

date -d @1522678262

date +%s -d "2018-04-02 22:22:27" //具体时间显示成时间戳





9、 date -d "+1 day" //一天后

date -d "-1 day" //一天前

10、 date -d "+1 month" //一月后

date -d "-1 month" //一月前

11、 date -d "+1 hour" //一小时后

date -d "-1 hour" //一小时前

12、 date -d "+1 min" //一分钟后

date -d "-1 min" //一分钟前

13、if 判断文件、目录属性

[ -f file ]判断是否是普通文件，且存在

[ -d file ] 判断是否是目录，且存在

[ -e file ] 判断文件或目录是否存在

[ -r file ] 判断文件是否可读

[ -w file ] 判断文件是否可写

[ -x file ] 判断文件是否可执行

14、grep -w //精确匹配

15、grep -q //不输出结果

16、if [ -z "$a" ] 表示当变量a的值为空时会怎么样 ，变量为文件可以不用双引号

if [ -n "$a" ] 表示当变量a的值不为空

if grep -q '123' 1.txt; then  表示如果1.txt中含有'123'的行时会怎么样

if [ ! -e file ]; then 表示文件不存在时会怎么样

if (($a<1)); then …等同于 if [ $a -lt 1 ]; then…

[ ] 中不能使用<,>,==,!=,>=,<=这样的符号

17、if ! (netstat -lnpt |grep -qw 80); then systemctl start httpd; fi

//如果80端口不存在，就启动httpd

18、case

格式 case变量名 in   
         value1)  
         command  
          ;;  
         value2)  
         command  
          ;;  
           \*)  
         commond  
           ;;  
           esac

在case程序中，可以在条件中使用|，表示或的意思， 比如      
2|3)   
    command  
    ;;

19、for

语法：for 变量名 in 条件; do …; done

20、while

让用户输入纯数字，如果输入的条件不满足就一直让其输入，当满足条件时才退出

while 条件; do … ; done

#!/bin/bash

while :

do

read -p "输入一个数字：" n

if [ -z "$n" ]

then

echo "请输入一个数字"

continue

fi

n1=`echo $n|sed 's/[0-9]//g'`

if [ -n "$n1" ]

then

echo "请输入一个纯数字"

continue

fi

break

done

echo $n

**二、break跳出循环**

#!/bin/bash

for i in `seq 1 5`

do

echo $i

if [ $i -eq 3 ]

then

break

echo aaa

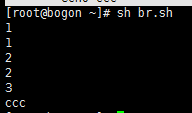
fi

echo $i

done

echo ccc

输入结果：



备注：先第一个echo $i,打印1，1 -eq 3不成立，执行第二个echo $i，打印1，跳到for,第二次执行第一个echo $i,打印2，2 -eq 3 不成立，执行第二个echo $i，打印2，跳到for,第三次执行第一个echo $i,打印3，3 -eq 3 成立，执行break,执行后跳出循环，不再执行循环里的任何操作，最后执行echo ccc

**三、continue结束本次循环**

#!/bin/bash

for i in `seq 1 5`

do

echo $i

if [ $i -eq 3 ]

then

continue

echo aaa

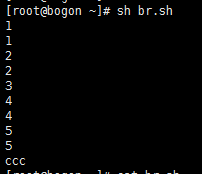
fi

echo $i

done

echo ccc

运行结果：



备注：备注：先第一个echo $i,打印1，1 -eq 3不成立，执行第二个echo $i，打印1，跳到for,第二次执行第一个echo $i,打印2，2 -eq 3 不成立，执行第二个echo $i，打印2，跳到for,第三次执行第一个echo $i,打印3，3 -eq 3 成立，执行continue，跳出本次循环，continue后面的echo aaa是不会执行的，执行下一次for循环，此时第四次执行第一个echo $i,打印4， 4 -eq 3 不成立，执行第二个 echo $i,打印4…… 打印5，最后执行到循环外面的ccc

**四、exit退出整个脚本**

#!/bin/bash

for i in `seq 1 5`

do

echo $i

if [ $i -eq 3 ]

then

exit

echo aaa

fi

echo $i

done

echo ccc

执行结果：

1

1

2

2

3

**五、shell中的函数**

#!/bin/bash

input(){ //定义一个函数

echo $1 $2 $# $0 //$1:第一个参数；$#:参数个数；$0:脚本的名字

}

input 1 2 //调用函数

sh fu.sh

执行结果

1 2 2 fu.sh

################################

#!/bin/bash

input(){

echo $1 $2 $# $0

}

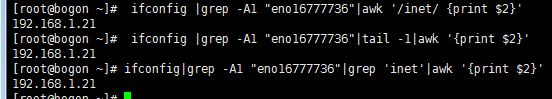
input $1 $2 //调用函数

sh fu.sh 1 2

执行结果

1 2 2 fu.sh

显示网卡IP



输入网卡名称，显示ip地址

#!/bin/bash

ip()

{

ifconfig |grep -A1 "$1: "|awk '/inet/ {print $2}' //$1是第一个参数，及输入的网卡名称

}

read -p "please input the eth name: " eth

ip $eth

执行sh ip.sh

提示输入网卡名称：

显示IP地址

#!/bin/bash

ip() {

while :

do

read -p " 输入网卡名称: " eth

a=`ifconfig |grep -A1 "$eth:" |tail -1 |awk -F" " '{print $2}'`

e=`ifconfig |grep -A1 "$eth:"`

if [ -z "$e" ]

then

echo "输入的网卡不存在，请重新输入: "

continue

else

p=`ifconfig $eth|grep inet`

if [ -z "$p" ]

then

echo $eth not ip

else

echo $eth ip is $a

fi

fi

exit

done

}

ip $eth

//输入网卡名称，显示对应的IP，如果输入的网卡名称不对，继续输入，如果输入的网卡名称对了，先判断网卡有无IP，如果无退出，如果有显示网卡ip

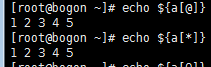
**六、 shell中的数组**

1、定义数组

a=(1 2 3 4 5 ) or a=(`seq 1 5`)

2、打印数组元素

echo ${a[@]} or echo ${a[\*]}



3、显示某个元素的值（元素从0开始）



3、显示元素个数

echo ${#a[\*]}

4、元素的赋值

a[0]=0

5、删除一个元素

unset a[0]

6、删除一个数组

unset a

7、从第一个元素开始截取3个

echo ${a[\*]:0:3}



8、从第二个元素开始截取4个

echo ${a[\*]:1:4}



9、从倒数第3个元素开始，截取3个

echo ${a[\*]:0-3:3}



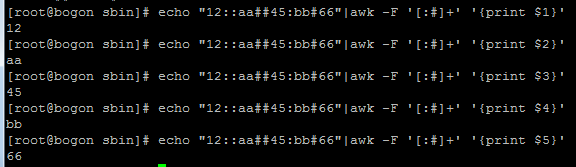
10、数组值的替换

echo ${a[\*]/5/1} //5替换成1



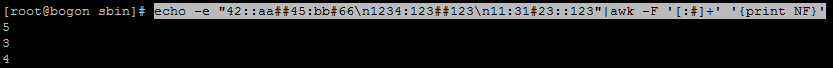
11、多个分隔符,如果打印每一段//

echo "12::aa##45:bb#66"|awk -F '[:#]+' '{print $2}'



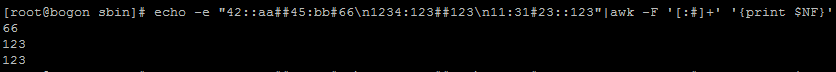
12、echo -e是和后面的\n配合用的，打印换行，NF打印出每行有几段

echo -e "42::aa##45:bb#66\n1234:123##123\n11:31#23::123"|awk -F '[:#]+' '{print NF}'



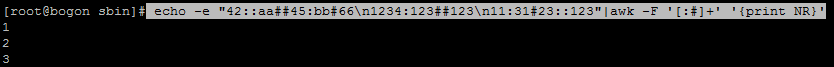
$NF 打印出每行最后一段的内容

echo -e "42::aa##45:bb#66\n1234:123##123\n11:31#23::123"|awk -F '[:#]+' '{print $NF}'



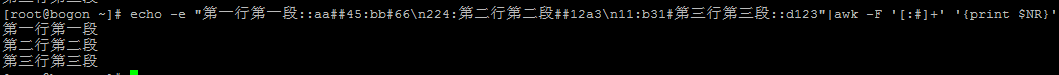
NR 打印行号

echo -e "42::aa##45:bb#66\n1234:123##123\n11:31#23::123"|awk -F '[:#]+' '{print NR}'



$NR 打印第几行的第几段（总共有5行，分别打印第一行第一段，第二行第二段……第5行第5段）

echo -e "第一行第一段::aa##45:bb#66\n224:第二行第二段##12a3\n11:b31#第三行第三段::d123"|awk -F '[:#]+' '{print $NR}'



**七、expect**

1、安装包

yum install -y expect

cat 1.expect //自动登录脚本

#! /usr/bin/expect

set host "192.168.1.41"

set passwd "123456"

spawn ssh root@$host

expect {

"yes/no" { send "yes\r"; exp\_continue}

"password:" { send "$passwd\r" }

}

interact //留在登录的主机里，不退出来

#######################

2.expect //自动登录到远程主机，执行相关命令后，退出来

#!/usr/bin/expect

set user "root"

set passwd "123456"

spawn ssh $user@192.168.1.41

expect {

"yes/no" { send "yes\r"; exp\_continue}

"password:" { send "$passwd\r" }

}

expect "]\*"

send "touch /tmp/12.txt\r"

expect "]\*"

send "echo 1212 >> /tmp/12.txt\r"

expect "]\*"

send "exit\r"

3、cat 3.exect //执行脚本的时候传递参数

#!/usr/bin/expect

set user [lindex $argv 0]

set host [lindex $argv 1]

set passwd [lindex $argv 2]

set cm [lindex $argv 3]

spawn ssh $user@$host

expect {

"yes/no" { send "yes\r"}

"password:" { send "$passwd\r" }

}

expect "]\*"

send "$cm\r"

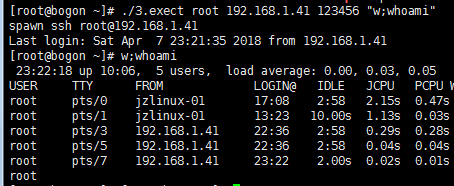
set timeout 5 //设置执行命令后的超时时间，-1为永不退出

expect "]\*"

send "exit\r"

chmod a+x 3.expect

./3.expect root 192.168.1.41 123456 "ls;w"



4、将远程的主机的mysql拷贝到本地目录里面

#!/usr/bin/expect

set passwd "123456"

spawn rsync -av root@192.168.1.30:/usr/local/src/mysql-5.6.35-linux-glibc2.5-x86\_64.tar.gz /tmp/

expect {

"yes/no" {send "yes\r"}

"password:" {send "$passwd\r"}

interact //方大括号后面会报错，有待测试

}