

算术运算：

整数：expr（运算符前后要有空格）、\$[]、let（不回显）

X++ x=x+1

X-- x=x-1

X+2 x=x+2

X*=3 x=x*3

X/=4 x=x/4

小数：（也可以整数计算）bc

Bc 回车（交互方式）

Echo "scale=4(小数位保留几位)" | bc （非交互）

测试：

字符串：[a == b] 、 [a != b] 、 [-z \$abc （测试 abc 是否为空）]

Read -p "请输入用户：" abc

[-z \$abc] && echo "你没有输入" && exit

Useradd \$abc

数字：

-eq -ne -lt -gt -le -ge

文件或目录：

-e exist(存在)

-f file（存在且为文件）

-d directory（存在且为目录） [-d /abc] || mkdir /abc

-r （读取权限）

-w （写入权限）

-x （执行权限）

For 循环

遍历/列表式循环

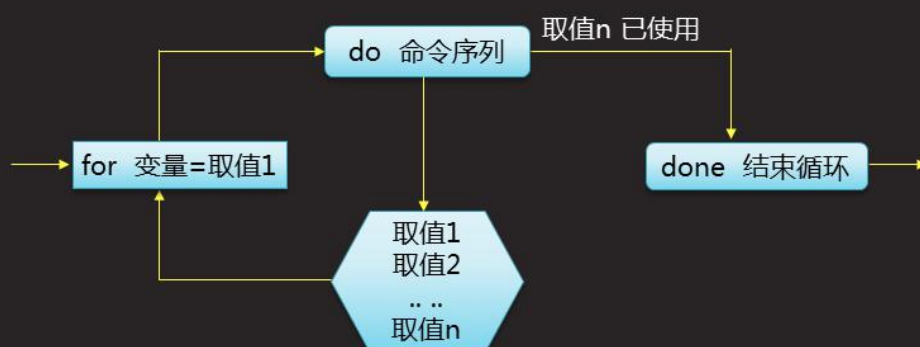
- 根据变量的不同取值，重复执行命令序列

```
for 变量名 in 值列表
do
    命令序列
done
```



```
for 收件人 in 邮件列表
do
    发送邮件
done
```

流程示意图



```
#!/bin/bash
for i in 1 8 5 7
do
    echo "NB is nb"
done
```

i 控制次数

```
[root@localhost ~]# sh 1.sh
NB is nb
NB is nb
NB is nb
NB is nb
```

i 控制次数

```
#!/bin/bash
for i in 1 8 5 7
do
    echo "NB is $i"
done
```

i 控制次数，影响输出

```
[root@localhost ~]# sh 2.sh
NB is 1
NB is 8
NB is 5
NB is 7
```

i 控制次数，影响输出

任务目标

- 批量添加用户账号（名称无规律）

```
[root@svr5 ~]# cat uaddfor.sh
#!/bin/bash
ULIST=$(cat /root/users.txt)
for UNAME in $ULIST
do
    useradd $UNAME
    echo "123456" | passwd --stdin $UNAME
done
```



```
root@svr5:~
[root@svr5 ~]# cat /root/users.txt
zhangsan
lisi
[root@svr5 ~]# ./uaddfor.sh
Changing password for user zhangsan.
passwd: all authentication tokens upd
Changing password for user lisi.
passwd: all authentication tokens upd
[root@svr5 ~]#
```

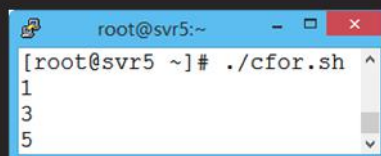
C语言风格的for循环

- 通过变量控制，条件成立时循环
- 步长可控次数

```
for ((初值; 条件; 步长控制))
do
    命令序列
done
```



```
[root@svr5 ~]# cat cfor.sh
#!/bin/bash
for ((i=1;i<=5;i+=2))
do
    echo $i
done
```



```
root@svr5:~
[root@svr5 ~]# ./cfor.sh
1
3
5
```

批量检测多个主机的存活状态

```
#!/bin/bash
for i in `seq 1 254`
do
    ping -c3 -i0.1 -W3 172.25.0.$i &> /dev/null
    if [ $? -eq 0 ];then
        echo "172.25.0.$i is up"
    else
        echo "172.25.0.$i is down"
    fi
done
```

造数--
{1..254} 中间不支持变量；不回显
`seq 1 254` 支持变量；回显
`seq 1 \$i`
\$(seq 1 \$i)

While 循环

条件式循环

- 反复测试条件，只要成立就执行命令序列

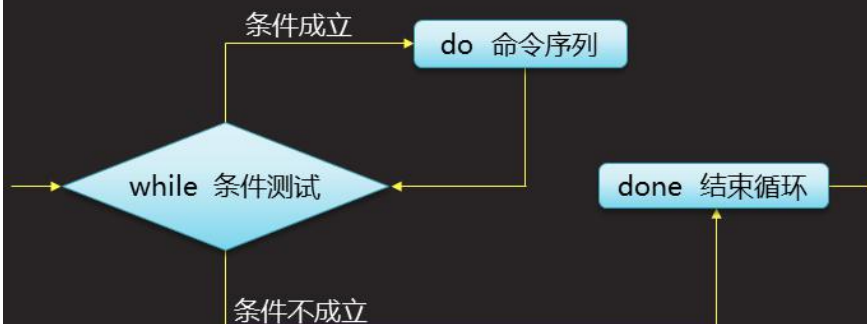
```
while 条件测试
do
    命令序列
done
```



```
while 未猜中正确价格
do
    反复猜商品价格
done
```



流程示意图



任务目标

- 批量添加用户（名称有规律）

```
[root@svr5 ~]# cat uaddwhile.sh
```

```
#!/bin/bash
```

```
PREFIX="tuser" ; i=1
```

```
while [ $i -le 5 ]
```

```
do
```

```
    useradd ${PREFIX}$i
```

```
    echo "123456" | passwd --stdin ${PREFIX}$i &> /dev/null
```

```
    let i++
```

```
done
```

```
root@svr5:~  
[root@svr5 ~]# ./uaddwhile.sh  
[root@svr5 ~]# tail -2 /etc/passwd  
tuser4:x:521:521::/home/tuser4:/bin/bash  
tuser5:x:522:522::/home/tuser5:/bin/bash
```

//递增控制，避免死循环

for 循环与 **while** 循环的区别：

for 循环固定次数，取值循环；**while** 循环不固定次数，条件判断

```
#!/bin/bash
```

```
while :
```

```
do
```

```
    echo XX
```

```
    echo YY
```

```
    sleep 0.1
```

```
done
```

#死循环

脚本实现猜数游戏

```
#!/bin/bash
num=$((RANDOM%100+1))
while :
do
    read -p "请输入1-100中任意一个整数：" x
    if [ $x -eq $num ]; then
        echo "恭喜！猜对了！"
        exit
    elif [ $x -gt $num ]; then
        echo "猜大了！请重新输入"
    else
        echo "猜小了！请重新输入"
    fi
done
```

```
#!/bin/bash
#批量添加用户帐号stu1-stu20
PREFIX="stu" #定义用户名前缀
i=1
while [ $i -le 20 ]
do
    useradd ${PREFIX}$i #添加的用户名为：前缀+编号
    echo "123" | passwd --stdin ${PREFIX}$i &> /dev/null
    let i++
done
```

```
#!/bin/bash
#批量检测某网段主机在线状况
NET="176.121.205."
i=1
while [ $i -le 254 ]
do
    IP="${NET}$i"
    ping -c 2 -i 0.1 -W 2 $IP &> /dev/null
    if [ $? -eq 0 ]; then
        echo "HOST $IP IS UP"
    else
        echo "HOST $IP IS DOWN"
    fi
    let i++
done
```

Case 语句（简化版的 if 循环，功能简单）

检查变量的实际取值

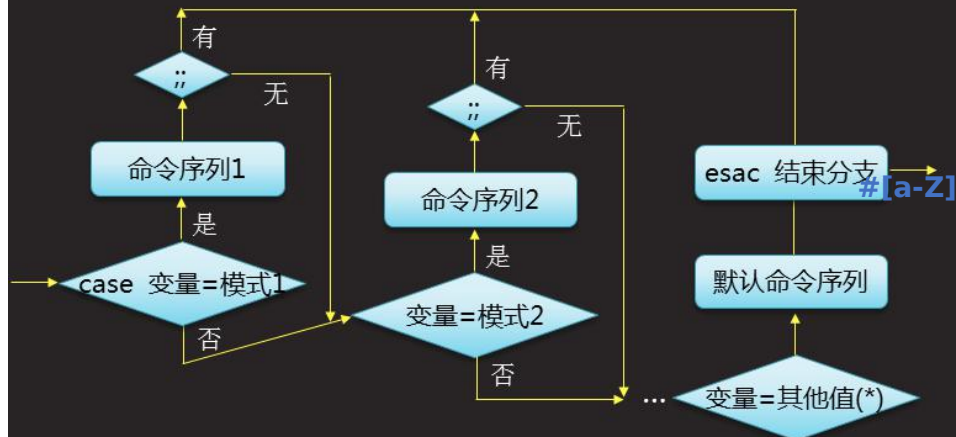
- 如果与预设的值相匹配，则执行对应的操作

```
case 变量值 in
模式1)
    命令序列1 ;;
模式2)
    命令序列2 ;;
...
*)
    默认命令序列
esac
```



```
case 控制参数 in
start)
    启动XX服务 ;;
stop)
    停止XX服务 ;;
...
*)
    显示服务脚本用法
esac
```

• 流程示意图



```

#!/bin/bash
case $1 in
fedora)
    echo "redhat";;
redhat)
    echo "fedora";;
*)
    echo "Usage: xxoo";;
esac
    
```

• 应用示例

– 判断用户按键类型

```

root@svr5:~# ./hitkey.sh
请输入一个字符: k
字母。
root@svr5:~# ./hitkey.sh
请输入一个字符: 8
数字。
root@svr5:~# ./hitkey.sh
请输入一个字符: ^[[21~
空格、功能键或其他控制字符
root@svr5:~#
    
```

```

[root@svr5 ~]# cat hitkey.sh
#!/bin/bash
read -p "请输入一个字符: " KEY
case "$KEY" in
[a-z]|[A-Z])
    echo "字母。" ;;
[0-9])
    echo "数字。" ;;
*)
    echo "空格、功能键或其他控制字符。"
esac
    
```

```

#!/bin/bash
case $1 in
-n)
    touch $2;;
-e)
    vim $2;;
-c)
    cat $2;;
-r)
    rm -rf $2;;
*)
    echo "Usage: $0 [ -n | -e | -c | -r ] name"
esac
    
```



```
[ root@localhost ~]# sh case.sh
Usage: case.sh [ -n | -e | -c | -r ] name
[ root@localhost ~]# sh case.sh -n 1.txt
[ root@localhost ~]# sh case.sh -e 1.txt
[ root@localhost ~]# sh case.sh -c 1.txt
haha
[ root@localhost ~]# sh case.sh -r 1.txt
```

SHELL 函数（给一段代码取个别名）

如何定义一个函数

```
function 函数名 {
    命令序列
    ...
}
```

或者

```
函数名() {
    命令序列
    ...
}
```

```
[ root@localhost lele]# echo -e "\033[ 33mOK\033[ 0m"
OK
[ root@localhost lele]# echo -e "\033[ 41mOK\033[ 0m"
OK
[ root@localhost lele]# echo -e "\033[ 1mOK\033[ 0m"
OK
```

#3x 字体色
4x 背景色
0x 字体样式

```
#!/bin/bash
cecho() {
    echo -e "\033[ $1m$2\033[ 0m"
}
cecho 31 OK
cecho 32 ERROR
```

#脚本中可以定义函数让输出更加醒目

```
[ root@localhost lele]# sh cecho.sh
OK
ERROR
```

#脚本中可以定义函数让输出更加醒目

任务目标

- 新建函数mkcd，用来创建一个目录，并切换到此目录

```
[root@svr5 ~]# mkcd() {
> mkdir $1
> cd $1
> }
```



```
root@svr5:~
[root@svr5 ~]# mkcd /opt/newdir1
[root@svr5 newdir1]# pwd
/opt/newdir1
[root@svr5 newdir1]#
```

Shell版fork炸弹

- 仅13个字符：`.0{ .|& };`
- 递归死循环，可迅速耗尽系统资源

代码解析

<code>.0</code>	//定义一个名为.的函数
<code>{</code>	//函数块开始
<code>. &</code>	//在后台递归调用函数.
<code>}</code>	//函数块结束
<code>;</code>	//与下一条执行语句的分隔
<code>.</code>	//再次调用函数

脚本的终端及退出

• 中断、继续、退出

类 型	含 义
break	跳出当前所在的循环体，执行循环体后的语句块
continue	跳过循环体内余下的语句，重新判断条件以决定是否需 要执行下一次循环 #结束本次循环，进入下次循环
exit	退出脚本，默认的回值是 0

```
#!/bin/bash
for i in {1..5}
do
    [ $i -eq 3 ] && continue
    echo $i
done
echo over
```

```
[ root@localhost ~]# sh test.sh
1
2
4
5
over
```

```
#!/bin/bash
for i in {1..5}
do
    [ $i -eq 3 ] && break
    echo $i
done
echo over
```

```
[ root@localhost ~]# sh test.sh
1
2
over
```



```
#!/bin/bash
for i in {1..5}
do
    [ $i -eq 3 ] && exit
    echo $i
done
echo over
```

```
[root@localhost ~]# sh test.sh
1
2
```

任务目标

- 从键盘循环取整数（0结束）并求和，输出最终结果

```
[root@svr5 ~]# cat brkwhile.sh
```

```
#!/bin/bash
```

```
while read -p "请输入待累加的整数（0表示结束）：" x
```

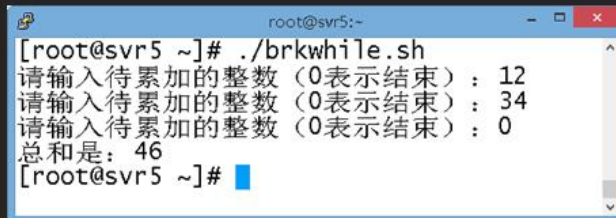
```
do
```

```
    [ $x -eq 0 ] && break
```

```
    SUM=$((SUM+x))
```

```
done
```

```
echo "总和是：$SUM"
```



```
root@svr5:~
[root@svr5 ~]# ./brkwhile.sh
请输入待累加的整数（0表示结束）： 12
请输入待累加的整数（0表示结束）： 34
请输入待累加的整数（0表示结束）： 0
总和是： 46
[root@svr5 ~]#
```

任务目标

- 跳过1~20以内非6的倍数，输出其他数的平方值

```
[root@svr5 ~]# cat cntwhile.sh
```

```
#!/bin/bash
```

```
i=0
```

```
while [ $i -le 20 ]
```

```
do
```

```
    let i++
```

```
    [ ${i%6} -ne 0 ] && continue
```

```
    echo ${i*i}
```

```
done
```



```
root@svr5:~
[root@svr5 ~]# ./cntwhile.sh
36
144
324
[root@svr5 ~]#
```

任务目标

- 利用位置参数获取2个整数，计算出这两个整数的和
- 如果参数不够2个，则提示正确用法并退出脚本

```
[root@svr5 ~]# cat exit.sh
#!/bin/bash
if [ $# -ne 2 ]; then
    echo "用法:$0 num1 num2"
    exit 10 //退出脚本，返回值设为10
fi
expr $1 + $2
```

1. 计算 $1+2+3+4+\dots+100$ 的和

```
#!/bin/bash
sum=0
for i in {1..100}
do
    let sum=sum+i #sum=$((sum+i))
done
echo "1-100的和是：$sum"
```

2. 让用户输入数字，求和，直到输入 **over** 为止

```
#!/bin/bash
sum=0
while :
do
    read -p "请输入数字(输入over结束)：" num
    if [ $num != over ];then
        let sum=sum+$num #sum=$((sum+num))
    else
        echo "输入所有数字的和为：$sum"
        exit
    fi
done
```

```
#!/bin/bash
sum=0
while :
do
    read -p "请输入数字(输入over结束)：" num
    [ $num == "over" ] && break
    let sum=sum+$num #sum=$((sum+num))
done
echo "输入所有数字的和是$sum"
```