**第二十一章 结构化命令case和for、while循环**

**本节所讲内容：**

**21.1 case - 流程控制语句**

**21.2 循环语句**

**21.3 while循环语句和循环嵌套**

**21.4 实战-3个shell脚本实战**

# 21.1 流程控制语句：case

**控制语句：用来实现对程序流程的选择、循环、转向和返回等进行控制。case是开关语句的一个组成部分；**

**它是根据变量的不同进行取值比较，然后针对不同的取值分别执行不同的命令操作**

**适用于多分支，是一个多选择语句**

**case     变量或表达式     in**

**变量或表达式1）**

**命令序列1**

**；；**

**变量或表达式2）**

**命令序列2**

**；；**

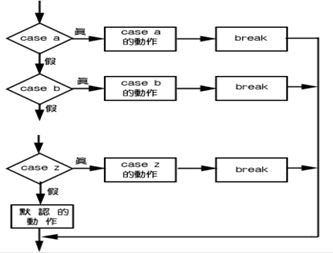
**……**

**\*）**

**默认命令序列**

**esac**

**case语句执行流程控制：**



**执行流程：**

**首先使用“变量或表达式”的值与值1进行比较，若取值相同则执行值1后的命令序列，直到遇见双分号“；； ”后跳转至esac，表示分支结束；**

**若与值1不相匹配，则继续与值2 进行比较，若取值相同则执行值2 后的命令序列，直到遇见双分号“；； ”后跳转至esac，表示结束分支。**

**依次类推，若找不到任何匹配的值，则执行默认模式“ \*） ”后的命令序列，直到遇见esac后结束分支**

**注意事项：**

* **“变量或表达式”后面必须为单词in，每一个“变量或表达式”的值必须以右括号结束。取值可以为变量或常数。匹配发现取值符合某一模式后，其间所有命令开始执行直至 ;;**
* **匹配中的值可以是多个值，通过“|”来分隔**

**例1：编写一个操作文件的脚本**

**[root@xuegod63 ~]# cat case-1.sh**

**#########################################################################**

**# File Name: case-1.sh**

**# Author: xuegod**

**#########################################################################**

**#!/bin/bash**

**cat <<eof**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\* 1. backup \*\***

**\*\* 2. copy \*\***

**\*\* 3. quit \*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**eof**

**read -p "Input a choose: " OP**

**case $OP in**

**1|backup)**

**echo "BACKUP......"**

**;;**

**2|copy)**

**echo "COPY....."**

**;;**

**3|quit)**

**exit**

**;;**

**\*)**

**echo error**

**esac**

**例2：编写一个启动apache服务脚本**

**[root@xuegod63 ~]# yum install httpd -y**

**[root@xuegod63 ~]# vim case-2.sh**

**#!/bin/bash**

**case $1 in**

**start)**

**systemctl $1 httpd**

**ps aux|grep httpd**

**echo "httpd start"**

**;;**

**stop)**

**systemctl $1 httpd**

**ps aux|grep httpd**

**echo "httpd stop"**

**;;**

**status)**

**systemctl $1 httpd**

**;;**

**restart)**

**systemctl $1 httpd**

**echo "httpd restart"**

**;;**

**\*)**

**echo "USAGE: $0 start|stop|restart"**

**esac**

# 21.2 循环语句

## 21.2.1 for-do-done

**语法格式：**

**for var in list**

**do**

**commands**

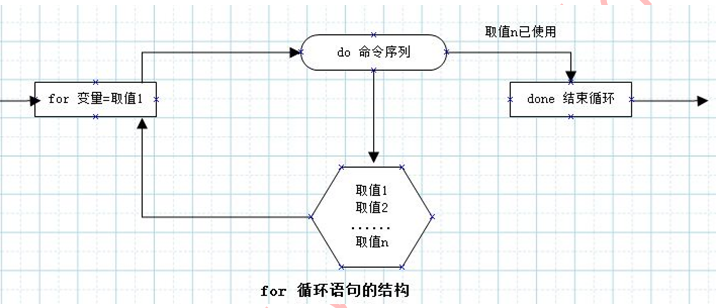
**done**

**或：**

**for var in list ; do**

**commands**

**done**

****

**取值列表有多种取值方式，比如**

**可以直接读取in 后面的值，默认以空格做分隔**

**[root@xuegod63 ~]# vim for-1.sh**

**#!/bin/bash**

**for var in a1 b1 c1 d1**

**do**

**echo the text is $var**

**done**

**列表中的复杂值，可以使用 引号或转义字符”/”来加以约束**

**[root@xuegod63 ~]# cat for-2.sh**

**#!/bin/bash**

**for var in a1 b1 "c1 d1" e2 "hello world"**

**do**

**echo the text is $var**

**done**

**[root@xuegod63 ~]# cat for-3.sh**

**#!/bin/bash**

**for var in a1 b\'1 "c1 d1" e2 "hello world" I\'s a22**

**do**

**echo the text is $var**

**done**

**从变量中取值**

**[root@xuegod63 ~]# cat for-4.sh**

**#!/bin/bash**

**list="a1 b1 c1 d1"**

**for i in $list**

**do**

**echo is a $i**

**done**

**4、从命令中取值**

**[root@xuegod63 ~]# cat for-5.sh #以空格做分隔符**

**#!/bin/bash**

**for i in `cat /etc/hosts`**

**do**

**echo "$i"**

**done**

**5、自定义shell分隔符**

**默认情况下，base shell会以空格、制表符、换行符做为分隔符。通过IFS来自定义为分隔符**

**指定单个字符做分隔符：**

**IFS=： #以：冒号做分隔符**

**可以指定多个**

**如 IFS='\n':;" #这个赋值会将反斜杠、n、冒号、分号和双引号作为字段分隔符。**

**注：$'\n'与'\n'时的区别**

**IFS='\n' #将字符\和字符n作为IFS的换行符。**

**IFS=$'\n' #正真的使用换行符做为字段分隔符。**

**[root@xuegod63 ~]# cat for-6.sh #指定以\n回车做为 for语句的分隔符**

**#!/bin/bash**

**IFS=$'\n'**

**for i in `cat /etc/hosts`**

**do**

**echo "$i"**

**done**

**[root@xuegod63 ~]# cat for-7.sh #以：冒号做分隔符**

**#!/bin/bash**

**IFS=:**

**list=`head -1 /etc/passwd`**

**for i in $list**

**do**

**echo $i**

**done**

**6、C语言风格的for**

**语法格式：**

**for ((i=0;i<10;i++))**

**do**

**commmands**

**done**

**例1：单个变量。 输出1到10之间的数字**

**[root@xuegod63 ~]# cat for-8.sh**

**#!/bin/bash**

**for (( i=1 ; i<=10 ; i++ ))**

**do**

**echo num is $i**

**done**

**注：**

**互动： i++ 这一条语句在for循环体中哪个位置执行？**

**for (( i=1 ; i<=10 ; )) #i=1 只赋值一次。然后执行 i <= 10**

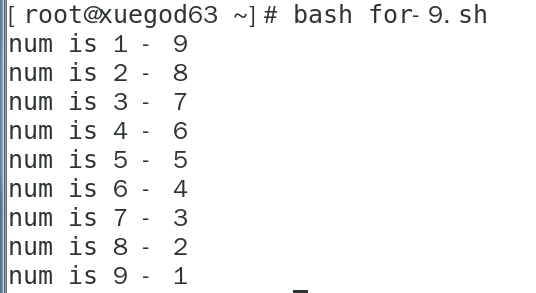
**do**

**echo num is $i**

**i=$(($i+1)) # i++在这里执行。 当for循环体中所有命令执行完后，再执行i++**

**done**

**例2：多个变量。 同时输出1-9的升序和降序**



**[root@xuegod63 ~]# cat for-8.sh**

**#!/bin/bash**

**for ((a=1,b=9 ; a<10 ; a++,b--))**

**do**

**echo num is $a - $b**

**done**

# 21.3 while循环语句和循环嵌套

## 21.3.1 while-do-done

**重复测试指令的条件，只要条件成立就反复执行对应的命令操作，直到命令不成立或为假；**

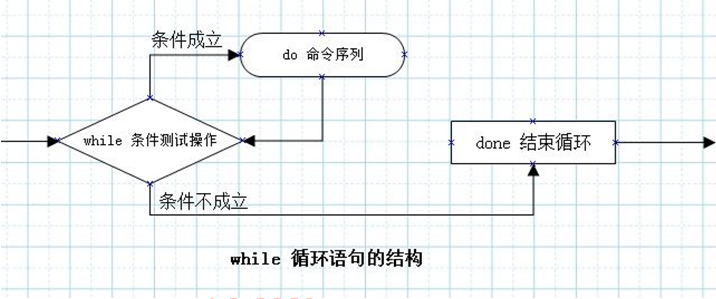
**语法格式如下：**

**while 测试命令**

**do**

**命令**

**done**

****

**注意：避免陷入死循环 while true**

**例1：降序输出10到1的数字**

**[root@xuegod63 ~]# cat while-1.sh**

**#!/bin/bash**

**var=10**

**while [ $var -gt 0 ]**

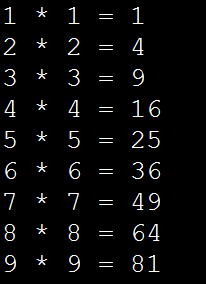
**do**

**echo $var**

**var=$[$var-1]**

**done**

**例2：输出如下图两数相乘的效果**

****

**自增操作 let var++**

**自减操作 let var--**

**[root@xuegod63 ~]# cat while-2.sh**

**#!/bin/bash**

**num=1**

**while [ $num -lt 10 ]**

**do**

**sum=$(( $num \* $num ))**

**echo "$num \* $num = $sum"**

**((num++))**

**# let num++**

**done**

## 21.3.2 嵌套循环

**例1：批量添加a.txt文件5个用户**

**[root@xuegod63 ~]# vim a.txt #添加5个用户**

**mk**

**cd**

**ls**

**find**

**ln**

**编写脚本的思路**

**1 明确脚本的功能**

**2 编写脚本时会使用到那些命令 ？ useradd passwd for**

**3 把变化的数据使用变量表示**

**4 选择适合的流程控制 （选择 、 循环 、分支）**

**[root@xuegod63 ~]# cat for-adduser.sh**

**#!/bin/bash**

**for name in `cat /root/a.txt`**

**#for name in $(cat /root/a.txt)**

**do**

**id $name &> /dev/null**

**if [ $? -ne 0 ];then**

**useradd $name**

**echo "123456" |passwd --stdin $name &> /dev/null**

**echo "user $name created"**

**else**

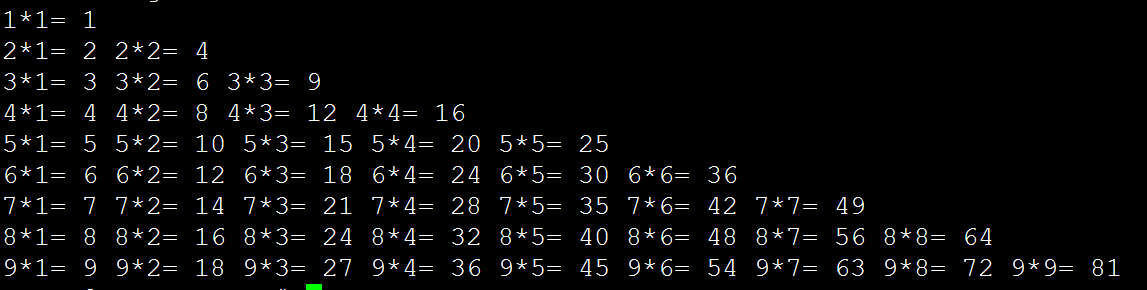
**echo "user $name is exist"**

**fi**

**done**

**注：&> 是正确和错误的信息都重定向到/dev/null里面**

**例2 ：打印九九乘法表**

****

**注：外层循环循环行，内层循环循环列**

**规律: 内层循环的变量<=外层循环的变量**

**[root@xuegod63 ~]# cat for-for.sh**

**for i in `seq 9`**

**do**

**for j in `seq $i`**

**do**

**echo -n "$i\*$j= `echo $(($i\*$j))` "**

**done**

**echo " "**

**done**

# 21.4 实战-3个shell脚本实战

## 21.4.1 实战-将/opt目录下所有的日志文件全自动打包

**[root@xuegod63 ~]# vim log-back.sh**

**#!/bin/sh**

**SRC\_DIR=/var/log/**

**DES\_DIR=/opt/backup/`date +%Y%m%d`**

**if**

**[ ! -d $DES\_DIR ] ; then**

**mkdir -p $DES\_DIR**

**fi**

**for i in `find $SRC\_DIR -name "\*.log"`**

**do**

**tar czf $i.tgz $i**

**done**

**mv /var/log/\*.tgz $DES\_DIR**

**ls -lh $DES\_DIR**

**echo "The scripts exec end, Files tar successfully !"**

## 21.4.2 实战-找出192.168.1.1-10网段中，服务器已经关机的IP地址

**[root@xuegod63 ~]# vim ping.sh**

**#!/bin/bash**

**i=1**

**for (( i=1;i<10;i++ ))**

**do**

**ping -c 3 192.168.1.$i &> /dev/null**

**if [ $? -ne 0 ];then**

**echo 192.168.1.$i is shutdown**

**fi**

**done**

## 21.4.3 批量创建帐号并生成随机密码

**[root@xuegod63 ~]# vim adduser.sh**

**#!/bin/bash**

**for i in xuegoda{1..10}**

**do**

**useradd $i**

**pass=`date +%s|md5sum|cut -c 1-8`**

**sleep 1**

**echo "$i:$pass" >> /tmp/passwd.log**

**echo $pass |passwd --stdin $i > /dev/null 2>&1**

**if [ $? -eq 0 ];then**

**echo "create user is successfully!"**

**else**

**echo "create user is failed!"**

**fi**

**done**

**总结：**

**21.1 case - 流程控制语句**

**21.2 循环语句**

**21.3 while循环语句和循环嵌套**

**21.4 实战-3个shell脚本实战**