**一    for 循环**

**1.1  for循环的语法结构**

**for  变量名     in      值列表**

**do**

**命令序列**

**done**

**案例1：**

**[root@svr7 ~]# vim baojing.sh**  
**#! /bin/bash**  
**for i in {aa,1,88,7,c};**  
**do**  
**echo hehe**  
**echo xixi**  
**done**  
**案例2：**

**[root@svr7 ~]# vim baojing.sh**

**#! /bin/bash**  
**for i in {aa,1,88,7,c};**  
**do**  
**echo $i**  
**done**  
  
**扩展内容：显示数字的2条命令**  
**[root@svr7 ~]# echo {1..5}**  
**1 2 3 4 5**  
**[root@svr7 ~]# seq 5**  
**1**  
**2**  
**3**  
**4**  
**5**  
**[root@svr7 ~]# seq 5 10**  
**5**  
**6**  
**7**  
**8**  
**9**  
**10**  
**案例3：批量检测主机的状态**

**[root@room8pc205 ~]# vim ping.sh**  
**#! /bin/bash**  
**for i in {1..100};**  
**do**  
**ping -c2 -i0.1 -W1 176.121.209.$i &> /dev/null**  
**if [ $? -eq 0 ];then**  
**echo "176.121.209.$i is up" >> /root/tong.txt**  
**else**  
**echo "176.121.209.$i is down" >> /root/butong.txt**  
**fi**  
**done**  
===================================================================================

**1.2 while 循环**

**判断测试是否为真，当结果为真时，执行do 和done 厘米的所有命令；**

**当测试为假的是好，循环结束。**

**语法： while 条件测试**

**do**

**命令序列**

**done**

**案例1：死循环的模式**

**[root@svr7 ~]# vim while.sh**

**#! /bin/bash**  
**i=1**  
**while [ $i -le 5 ]**  
**do**  
**echo $i**  
**done**

**案例2：死循环变活循环的模式**

**[root@svr7 ~]# vim while.sh**

**#! /bin/bash**  
**i=1**  
**while [ $i -le 5 ]**  
**do**  
**echo $i**

**let** **i+=1**  
**done**

**案例3：批量创建用户**

**[root@svr7 ~]# vim user.sh**  
**#! /bin/bash**  
**i=1**  
**while [ $i -lt 5 ]**  
**do**  
**useradd $i &> /dev/null**  
**echo '123' | passwd --stdin $i > /dev/null**  
**let i++**

**done**

**案例4：批量删除用户**

**[root@svr7 ~]# vim user.sh**

**#! /bin/bash**  
**i=1**  
**while [ $i -lt 11 ]**  
**do**  
**userdel -r user$i &> /dev/null**  
**let i++**  
**done**

**案例5：猜数字游戏**

**[root@svr7 ~]# vim test.sh**

**#! /bin/bash**

**i=0**  
**num=$[RANDOM%100]**  
**while :**  
**do**  
**read -p "你猜猜我出了几？" cai**

**let i++**  
**[ -z $cai ] && echo "请出100以内的数字" && exit**  
**if [ $cai -eq $num  ] ; then**  
**echo "恭喜，你猜对了。"**

**echo "你一共猜了$i次"**  
**exit**  
**elif [ $cai -lt $num ] ;then**  
**echo "猜错了，太小了。"**  
**else**  
**echo "猜错了，太大了。"**  
**fi**  
**done**  
==============================================================================================  
**三 基于case分支编写脚本**

**3.1 case分支的语法结构：**

**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**

**案例1：**

* 能使用redhat、fedora控制参数
* 控制参数通过位置变量$1传入
* 当用户输入redhat参数，脚本返回fedora
* 当用户输入fedora参数，脚本返回redhat
* 当用户输入其他参数，则提示错误信息

**[root@svr7 ~]# vim case1.sh**

**#! /bin/bash**  
**case $1 in**  
**redhat)**  
**echo "fedora";;**  
**fedora)**  
**echo "redhat";;**  
**\*)**  
**echo "用法：$0 {redhat|fedora}"**  
**esac**  
===============================================================================  
**四 使用shell函数**

**4.1 函数的定义方法：**

**格式1 ：**

**function 函数名 {**

**命令序列**

**}**

**格式2：**

**函数名（）{**

**命令序列**

**}**

**案例1：**

**[root@svr7 ~]# mymkdir(){**  
**> mkdir $1**  
**> cd $1**  
**> }**  
**[root@svr7 ~]# mymkdir  yyy**  
**[root@svr7 yyy]#**  
  
**扩展内容：[root@svr7 ~]# echo -e "\033[35mwangjie\033[0m"**

**###   -e 是扩展的意思； \033是打开颜色的意思；**

**35m是可改的：3x是字体色；4x是背景色；x是字体样式**

**wangjie  是要显示的内容**  
**案例2：fork炸弹**

**[root@svr7 ~]# vim zhadan.sh**  
**#! /bin/bash**  
**.(){**  
**.|. &**  
**}**  
**.**  
=========================================================================================  
**五 中断及退出**

**通过break、continue、exit在shell脚本中实现中断与退出的功能。**

**break可以结束整个循环；continue结束本次循环并进入下一个循环；exit结束整个脚本。**

**案例1：**

**[root@svr7 ~]# vim case3.sh  
#! /bin/bash  
for i in {1..5}  
do  
        [ $i -eq 3 ] && continue  
        echo $i  
done  
        echo over**

**案例2：**

**[root@svr7 ~]# vim case3.sh  
#! /bin/bash  
sum=0  
while :  
do  
        read -p "请输入一个数字：" num  
        [ $num -eq 0 ] && break  
        let sum+=num  
done  
echo $sum**

**在后台运行多个进程时，终端有可能会提前出现，导致终端提示符有一种卡住的感觉（按回车，再次出现命令提示行）**

**解决方法：在脚本后面添加一个循环**

**for i in {1..254}**

**do**

**wait $i**

**done**