shell_day01_回顾

- **变量**

```
#  = 左右两侧不能有空格
1、自定义变量
2、环境变量
3、位置变量
4、预定义变量 : $# $* $?
```

- **算术运算命令**

```
1、`expr n1 + n2`
2、$[n1+n2]
3、let i++
```

- **条件判断**

```
 # [ 条件 ]
 1、字符比较:  ==  !=  -z
 2、数值比较:  -gt -ge -lt -le -eq
 3、文件|目录比较: -e !-e -f -d
```

- **if分支结构**

```
if [ 条件 ];then
  语句
elif [ 条件 ];then
  语句
else
  语句
fi
```

- **for循环**

```
# for
for 变量名 in `seq {1..5}`
do
  循环体
done

# cfor
for ((i=1;i<=10;i++))
do
  循环体
done
```

- **while循环**

```
while [ 条件 ]
do
  循环体
done

# 死循环
while :
do
  循环体
done
```

- **sed**

```
sed -i '' file.txt
sed -n '' file.txt

1、增   : a | i
2、删   : d
3、改   : c
4、查   : p
5、替换 : s
```

- **其他**

```
1、获取随机数
  $[RANDOM%num]
```

shell_day02_笔记

- **作业**

用户从终端输入名字，创建该目录，回车直接退出

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

```
2
3  while :
4  do
5    read -p '请输入要创建的文件夹名字,直接回车退出:' dirname
6    directory="/home/tarena/$dirname"
7    if [ -z $dirname ];then
8      echo "程序退出"
9      exit
10   elif [ -e $directory ];then
11     echo "已存在"
12   else
13     mkdir $directory
14     echo "创建成功"
15   fi
16 done
```

- **until循环**

```
1  # 1、特点
2  条件判断不成立时执行循环体,成立时循环结束
3
4  # 2、语法格式
5  until [ 条件 ]
6  do
7    循环体
8  done
9
10 # 3、示例:把172.40.91.10-15内不在线的IP输出来
11 #!/bin/bash
12 x=10
13 until [ $x -gt 15 ]
14 do
15         ping -c 2 176.136.10.$x &> /dev/null
16         if [ $? -ne 0 ];then
17                 echo "176.136.10.$x"
18         fi
19
20         let x++
21 done
```

- **case分支结构**

```
1  # 1、特点
2  根据变量值的不同,执行不同的操作
3
4  # 2、语法格式
5  $变量名 in
6  模式1)
7    代码块
8    ;;
9  模式2)
10   代码块
11   ;;
```

```
*)
  代码块
  ;;
esac

# 3、示例 ： 输入一个字符,判断是数字、字母还是其他字符
#!/bin/bash

while :
do
  echo "++++++++++++++++++++++++++++++"
  echo "   Welcome(q to quit)         "
  echo "++++++++++++++++++++++++++++++"

  read -p "请输入一个字符:" char
  if [ ${#char} != 1 ];then
    echo "${char}不是一个字符"
  elif [ $char == 'q' ];then
    echo "程序退出"
    exit
  fi

  case $char in
  [a-z]|[A-Z])
    echo "字母"
    ;;
  [0-9])
    echo "数字"
    ;;
  *)
    echo "其他字符"
    ;;
  esac
done

# 4、练习 ： 编写1个nginx的启动脚本，包含: start stop restart
#!/bin/bash

read -p "操作(start|stop|restart):" op
case $op in
"start")
  sudo /etc/init.d/nginx restart
  ;;
"stop")
  sudo /etc/init.d/nginx stop
  ;;
"restart")
  sudo /etc/init.d/nginx restart
  ;;
*)
  echo "Please choice in start|stop|restart"
  ;;
esac
```

知识点总结

```
# 1、获取字符串长度
${#变量名}

# 2、字符串索引及切片
${string:index:number}
key='ABCDE'
${key:0:1} # A 获取下表索引为0的元素
${key:1:2} # BC

# 3、vim批量缩进
1、进入命令行模式 : shift + :
2、1,3> + Enter  : 1-3行缩进
3、1,3< + Enter  : 1-3行往回缩进
```

- **函数**

```
# 1、语法格式
函数名(){
  代码块
}
函数名  # 函数调用,不能加()

# 2、示例: 打印10个*
star(){
  echo "**********"
}
star # 第1次调用
star # 第2次调用

# 3、练习: 写1个计算器程序,计算 加 减 即可 -- 函数+case
#!/bin/bash

sumx(){
  echo $(($n1+$n2))
}

subx(){
  echo $(($n1-$n2))
}



read -p "输入第一个数字: " n1
read -p  "输入第二个数字: " n2
read -p "选择操作(+|-):" op


case $op in
"+")
  sumx
  ;;
"-")
  subx
  ;;
*)
```

```
  echo "Invalid"
  ;;
esac
```

练习

```
在用户主目录下创建一个目录,如果存在则提示,否则提示创建成功
#!/bin/bash
is_directory(){
    read -p "请输入要创建的目录名称:" dir
    if [ -d /home/tarena/$dir ];then
    echo "该目录已存在"
    else
    mkdir /home/tarena/$dir
    echo "目录 /home/tarena/$dir 创建成功"
    fi
}
is_directory
```

- **字符串处理 - ${变量名 替换符号 匹配条件}**

从左向右删除

```
# 1、语法
${变量名##匹配条件}

# 2、示例
directory="/home/tarena/mysql"   # 注意{}中不需要加空格
echo ${directory##*/}   --> mysql
echo ${directory#*/}    --> home/tarena/mysql
```

从右向左删除

```
# 1、语法
${变量名%%匹配条件}

# 2、示例
directory="/home/tarena/mysql"
echo ${directory%%/mysql}   --> /home/tarena
echo ${directory%/*}        --> /home/tarena
echo ${directory%%/*}       --> ""
```

案例

```
输出系统中的前10个用户

#!/bin/bash
for filename in `head -10 /etc/passwd`
do
  echo ${filename%%:*}
done
```

练习

```
批量修改文件名 ： 把当前目录下的.txt文件全部改为.doc文件

#!/bin/bash
for filename in `ls *.txt`
do
  name=${filename%.txt}
  mv $filename $name.doc
done
```

- **shell磨练**

1、依次提示用户输入3个整数，脚本根据数字大小依次排序输出 3个数字

```
#!/bin/bash

read -p "请输入一个整数:" num1
read -p "请输入一个整数:" num2
read -p "请输入一个整数:" num3
#不管谁大谁小，最后都打印echo "$num1,$num2,$num3"
#num1中永远存最小的值，num2中永远存中间值，num3永远存最大值
#如果输入的不是这样的顺序，则改变数的存储顺序，如：可以将num1和num2的值对调 tmp=0
#如果num1大于num2，就把num1和和num2的值对调，确保num1变量中存的是最小值
if [ $num1 -gt $num2 ];then
    tmp=$num1  num1=$num2  num2=$tmp
fi
#如果num1大于num3，就把num1 和num3对调，确保num1变量中存的是最小值
if [ $num1 -gt $num3 ];then
    tmp=$num1  num1=$num3  num3=$tmp
fi
#如果num2大于num3，就把num2 和num3对标，确保num2变量中存的是小一点的值
if [ $num2 -gt $num3 ];then
    tmp=$num2  num2=$num3  num3=$tmp
fi
echo "排序后数据为:$num1,$num2,$num3"
```

2、编写脚本，实现人机<石头，剪刀，布>游戏

```
# 提示 - Linux中数组使用
# Linux数组：(元素1 元素2 元素3)  元素之间用空格隔开
game=("石头" "剪刀" "布")
```

代码实现

```
#!/bin/bash

game=(石头 剪刀 布)
num=$[RANDOM%3]
computer=${game[$num]}
#通过随机数获取计算机的出拳
#出拳的可能性保存在一个数组中，game[0],game[1],game[2]分别是3中不同的可能

echo "请根据下列提示选择您的出拳手势"
echo "1.石头"
echo "2.剪刀"
```

```
echo "3.布"

read  -p  "请选择1-3:"  person

case  $person  in
1)
  if [ $num -eq 0 ];then
    echo "平局"
  elif [ $num -eq 1 ];then
    echo "你赢"
  else
    echo "计算机赢"
  fi;;
2)
  if [ $num -eq 0 ];then
    echo "计算机赢"
  elif [ $num -eq 1 ];then
    echo "平局"
  else
    echo "你赢"
  fi;;
3)  if [ $num -eq 0 ];then
    echo "你赢"
  elif [ $num -eq 1 ];then
    echo "计算机赢"
  else
    echo "平局"
  fi;;
*)
  echo "必须输入1-3的数字"

esac
```

3、每2秒中检测一次MySQL数据库的连接数量

```
# mysqladmin命令
mysql服务器管理任务的工具，它可以检查mysql服务器的配置和当前工作状态
```

代码实现

```
#!/bin/bash
#每2秒检测一次MySQL并发连接数

user=root
passwd=123456

while :
do
  sleep 2
  count=`mysqladmin  -u"$user"  -p"$passwd" status |  awk '{print $4}'`
  echo "`date +%Y-%m-%d` 并发连接数为:$count"
done
```

4、根据md5校验码，检测文件是否被修改

```
# 1、生成md5的文件校验码
md5sum nginx.conf
```

代码实现

```
#!/bin/bash
#本示例脚本检测的是/etc 目录下所有的conf结尾的文件
#本脚本在目标数据没有被修改时执行一次，当怀疑数据被人篡改，再执行一次
#将两次执行的结果做对比，MD5码发生改变的文件，就是被人篡改的文件
for  i  in  $(ls /etc/*.conf)
do
  md5sum "$i" >> /home/tarena/md5log.txt
done
```

5、备份MySQL数据库

```
# 备份MySQL数据库中的mysql库
#!/bin/bash

user="root"
passwd="123456"
dbname="mysql"
date=$(date +%Y%m%d)

#测试备份目录是否存在，不存在则自动创建该目录
[  ! -d  /home/tarena/mysqlbackup  ] &&  mkdir  /home/tarena/mysqlbackup
#使用mysqldump命令备份数据库
mysqldump -u"$user"  -p"$passwd" "$dbname" > /home/tarena/mysqlbackup/"$dbname"-${date}.sql
```

6、随机生成8为密码

```
#!/bin/bash
#设置变量key，存储密码的所有可能性（密码库），如果还需要其他字符请自行添加其他密码字符
#使用$#统计密码库的长度

key="0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
num=${#key}
#设置初始密码为空
pass=''
#循环8次，生成 8为随机密码
#每次都是随机数对密码库的长度取余，确保提取的密码字符不超过密码库的长度
#每次循环提取一位随机密码，并将该随机密码追加到pass变量的最后
for i in {1..8}
do
  index=$[RANDOM%num]
  pass=$pass${key:$index:1}
done
echo $pass
```