Linux-day02笔记

使用命令必须养成的习惯

```
1、tab键自动补全
2、Ctrl + l ： 清理屏幕
3、Ctrl + c ： 终止当前命令的执行
```

常用远程连接软件

```
# 终端仿真程序，其实就是Windows下登录UNIX或Linux服务器主机的软件，支持ssh、telnet
1、Xshell
2、Secure CRT

# xshell实现文件互传
1、xshell图形界面：新建文件传输
2、安装：lrzsz,是一款可在linux里可代替ftp上传和下载的程序
```

常见服务的端口号

```
MySQL - 3306
MongoDB - 27017
Redis - 6379
redis-sentinel - 26379
SSH - 22
HHTP - 80
NGINX - 80
HTTPS - 443
TELNET - 23
FTP - 21
```

文本处理工具 - awk

语法格式

```
awk 选项 '动作' 文件列表
```

常用方式

```
Linux命令  |  awk  选项  '动作'
```

作业

```
# nginx的访问日志目录 ：  /var/log/nginx/access.log
问题1：把访问过自己的IP地址输出
       # awk '{print $1}' access.log
问题2：统计有多少个IP访问过我
       # awk '{print $1}' access.log | sort | uniq | wc -l
问题3：统计每个IP地址的访问次数,输出前10个访问量最大的用户IP
       # awk '{print $1}' access.log | sort | uniq -c | sort  -rn -k 1 | head -10
```

grep命令之正则表达式

```
# 正则表达式元字符集 - 使用grep命令
^    :   以 ... 开头
$    :   以 ... 结尾
.     :   任何1个字符
*    :   0次或多次

# 正则表达式扩展字符集 - 使用 egrep 命令
+    :   1次或多次
{n} :   出现n次
()  :   分组

[a-z]   :  所有小写字母
[A-Z]  :  所有大写字母
[a-Z]  :  所有字母
[0-9]  : 所有数字
[a-Z0-9]  : 所有的字母和数字
```

应用场景

```
# Mac地址正则匹配
([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}
```

shell编程

Shell格式

```
1、扩展名：xxx.sh
2、正文第一行必须指定解释器：#!/bin/bash
```

shell执行方式

```
# 方式一：加权限， ./xxx.sh 执行
1、chmod +x  xxx.sh
2、./xxx.sh

# 方式二：手动指定解释器
bash xxx.sh
```

变量

- **自定义变量**

```
# 1. 定义变量
变量名=值     ---->  注意：=两侧绝对不能有空格
eg1: name="take me to your heart"

# 2. 调用变量的格式
echo $变量名

# 3. 小细节：单引号和双引号的区别
单引号：无法获取变量的值
双引号：可以获取变量的值
```

- **环境变量+位置变量+预设变量**

```
# 环境变量
echo $USER    --  当前用户
echo $UID     --  当前用户的UID号
echo $PWD     --  当前路径
echo $PATH    --  命令搜索路径

# 位置变量
$1 $2 $3 ... ... shell的位置变量

# 预定义变量
$# $* $?

# $? ：返回上一条命令执行的状态(0代表正确,非0代表失败)
```

示例

```
输出$1+$2,例如输出结果：3+5
#!/bin/bash
echo $1+$2
```

- **变量赋值 - 接收用户从终端输入的值**

```
# 语法格式
```

```
read -p 提示信息 变量名

# 示例
#!/bin/bash
read -p 请输入姓名: name
echo "您输入的姓名是:$name"

# 指定超时时间
read -p 提示信息 变量名
read -t n -p 提示信息 变量名

# 示例
#!/bin/bash
read -t 3 -p 请输入用户名: username
```

练习

```
1、输入学生姓名: 赵敏
2、输入学生成绩: 88
3、输出: 赵敏的成绩为88分
```

shell - 算术运算符

```
# 运算符
1、+ - * / %
2、++ : 自加1运算,类似于python中 i++  等同于 i+=1
3、-- : 同++

# 运算命令
1、let 运算表达式
  i=1
  let i++
  echo $i
2、expr 运算表达式
  i=1
  sum=`expr $i + 5` # +两侧要有空格
  echo $sum
3、$[]
  echo $[1+1]
  echo $[1-1]
  echo $[a+a] # 调用变量不用多次添加$符号
  echo $[1*1] # 乘法无需转义
```

练习

```
使用 位置变量+以上方法一、二中任何一种,实现2个数字的相加
#!/bin/bash
echo $[$1+$2]
echo `expr $1 + $2`
```

shell - 比较运算符

```
# 语法格式
  [  判断语句  ]  # 注意括号必须有空格

# 1、字符比较
  [ A == A ]  #相等(等号两边需要有空格)
  [ A != B ]  #不相等
  [ -z $变量 ]  #判断是否为空

# 2、数字比较
  -eq 等于(equal)
  -ne 不等于(not equal)
  -gt 大于(greater than)
  -ge 大于等于(great or equal)
  -lt 小于(less than)
  -le 小于等于(less or equal)

# 3、文件|目录比较
   [ -e 文件或目录 ]    #是否存在exist
   [ -f  文件      ]    #存在且为文件file
   [ -d  目录      ]    #存在且为目录directory
   [ -r 文件或目录 ]    #判断是否可读read
   [ -w 文件或目录 ]    #判断是否可写write
   [ -x 文件或目录 ]    #判断是否可执行
```

shell - if分支结构

```
# 1、单分支语法格式
     if 判断 ;then
        命令
        命令
     fi
# 2、双分支语法格式
  if 判断 ;then
    命令1
  else
    命令2
  fi
# 3、多分支语法格式
  if 判断;then
    命令1
  elif 判断 ;then
    命令2
  else
    命令3
  fi
# 示例
#!/bin/bash
if [ $USER == tarena ];then
  echo "Yes,You are Tarena."
```

```
else
  echo "You are other man."
fi
```

练习:使用shell编写猜数字游戏,无须循环

```
#!/bin/bash
num=$RANDOM
read -p "我有一个随机数,你猜:"  guess
if [ $guess -eq $num ];then
  echo "恭喜,猜对了."
  exit
elif [ $guess -gt $num ];then
  echo "你猜大了"
else
  echo "你猜小了"
fi
```

shell - for循环

```
# 语法格式
for 变量 in 值序列
do
  命令
done
# 示例
for i in 1 2 3 4 5
do
  echo "hello world"
done
```

练习:判断指定网段的IP地址哪些可以用,哪些不能用?

```
#!/bin/bash

for i in {1..254}
do
   ping -c 2 172.40.91.$i &>/dev/null
   if [ $? -eq 0 ];then
    echo "172.40.91.$i is up."
   else
    echo "172.40.91.$i is down"
   fi
done
```

shell - while循环

```
# 语法格式
while 条件判断
do
  命令
done

# 示例
#!/bin/bash
i=1
while [ $i -lt 5 ]
do
   echo baby
   let i++
done
```

sehll - case分支结构

```
# 1、特点
根据变量值的不同,执行不同的操作

# 2、语法格式
$变量名 in
模式1)
  代码块
  ;;
模式2)
  代码块
  ;;
*)
  代码块
  ;;
esac
```

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
```

shell实战

1、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 $4}'`
  echo "`date %F` 并发连接数为:$count"
done
```

2、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

```bash
#!/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
```

3、备份MySQL数据库

```bash
# 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
if [  ! -d  /home/tarena/mysqlbackup ];then
    mkdir  /home/tarena/mysqlbackup
fi

#使用mysqldump命令备份数据库
mysqldump -u"$user"  -p"$passwd" "$dbname" > /home/tarena/mysqlbackup/"$dbname"-${date}.sql

```

4、随机生成8为密码

```bash
#!/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
```