**一，字符串截取及切割**

**子串截取的三种方法：**

**1，${var:起始位置：长度}   （ 起始编号从0开始）**

**2，expr substr "$var"起始位置 长度  （ 起始编号从1开始）**

**3，echo $var | cut -b 起始位置-结束位置 （起始编号也是从1开始，与expr类似）**

**示例：**

**1)先定义一个变量**

**[root@svr7 ~]# phone=15812345678**

**[root@svr7 ~]# echo ${#phone}  
11  
2)使用${}截取时，起始位置可以省略，省略时从第一个字符开始截。起始位置是从0开始的（和数组下标编号类似） 。  
[root@svr7 ~]# echo ${phone::3}  
158  
或者 [root@svr7 ~]# echo ${phone:0:3}  
         158**

**案例1：从62个字符中，随机生成1个字符**

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

**#! /bin/bash**

**x=abcdefgihjklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789**  
**i=$[RANDOM%62]**

**echo ${x:$i:1}**

**案例2：从62个字符中，随机生成一个8字符的密码**

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

**#! /bin/bash**

**x=abcdefgihjklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789**

**num=""**  
**for i in {1..8}**  
**do**  
**sum=$[RANDOM%62]**  
**i=${x:$sum:1}**  
**num=$num$i**  
**done**  
**echo "$num"**  
  
**字符串的替换**

**只替换第一个子串: ${var/old/new}**

**替换全部子串: ${var//old/new}**

**示例：**

**[root@svr7 ~]# echo $phone**  
**15812345678**  
**[root@svr7 ~]# echo ${phone/8/x}**  
**15x12345678**  
**[root@svr7 ~]# echo ${phone//8/x}**  
**15x1234567x**  
  
**字符串的匹配删除**

**示例：**

**[root@svr7 ~]# x=`head -1 /etc/passwd`  
[root@svr7 ~]# echo $x  
root:x:0:0:root:/root:/bin/bash**

**1) 从左向右，最短匹配删除(掐头) ：${变量名#\*关键词}**

**[root@svr7 ~]# echo ${x#\*:}  
x:0:0:root:/root:/bin/bash**

**2) 从左向右，最长匹配删除 ：${变量名##\*关键词}**

**[root@svr7 ~]# echo ${x##\*:}  
/bin/bash**

**3) 从右向左，最短匹配删除(去尾) ：${变量名%关键词\*}**

**[root@svr7 ~]# echo $x  
root:x:0:0:root:/root:/bin/bash  
[root@svr7 ~]# echo ${x%root\*}  
root:x:0:0:root:/**

**4) 从右向左，最短匹配删除(去尾) ：${变量名%%关键词\*}**

**[root@svr7 ~]# echo ${x%0\*}**  
**root:x:**

**案例：批量修改扩展名.txt-->.doc**

**[root@svr7 ~]#  mkdir aa**

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

**#! /bin/bash  
for i in  \*.$1                       ###如果想对“a.$1;b.$1...h.$1”做修改，可把前面“\*.$1”改为"{a..h}.$1"  
do  
        mv $i ${i%.$1}.$2  
done**

**[root@svr7 ~]# cd aa  
[root@svr7 aa]# touch {a,b,c,d,e,f,g}.txt  
[root@svr7 aa]# ls  
a.txt  b.txt  c.txt  d.txt  e.txt  f.txt  g.txt**

**[root@svr7 aa]# sh /root/aa.sh txt doc  
[root@svr7 aa]# ls  
a.doc  b.doc  c.doc  d.doc  e.doc  f.doc  g.doc  
字符串初值的处理**

**通过${var:-word}判断变量是否存在，决定是否给变量赋初始值。**

**示例：**

**[root@svr7 ~]# echo ${i:-111}  
111  
[root@svr7 ~]# i=123  
[root@svr7 ~]# echo ${i:-111}  
123  
案例：输入1个数x，脚本执行1+2+3+4+...+x；如果没有输入任何数，则默认x为100。**

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

**#! /bin/bash  
read -p "请输入一个整数：" x  
x=${x:-100}  
i=1  
sum=0  
while [ $i -le $x ]  
do  
        let sum=sum+i  
        let i=i+1  
done  
echo "从1到$x的和为：$sum"  
[root@svr7 ~]# sh aa.sh  
请输入一个整数：50  
从1到50的和为：1275  
[root@svr7 ~]# sh aa.sh  
请输入一个整数：  
从1到100的和为：5050  
使用shell数组**

**建立数组的方法：**

**格式1， 整体赋值：数组名=(值1 值2 ..值n)**

**格式2，单个元素赋值：数组名[下标]=值**

**查看数组元素的方法：**

**获取单个数组元素： ${数组名[下标]}**

**获取所有数组元素： ${数组名[\*]}**

**连续多个数组元素：${数组名[@]:起始下标:元素个数}**

**截取数组元素值的一部分：${数组名[下标]:起始下标:字符数}**

**示例：**

**[root@svr7 ~]# i=(11 22 33)  
[root@svr7 ~]# echo ${i[0]}  
11  
[root@svr7 ~]# echo ${i[1]}  
22  
[root@svr7 ~]# echo ${i[2]}  
33**

**[root@svr7 ~]# echo ${i[\*]}**  
**11 22 33**

**案例：输入名字，并保存每个输入的名字，当输入为空时则退出循环**

**[root@svr7 ~]# vim aaa.sh**  
**#! /bin/bash**  
**i=1**  
**while :**  
**do**  
**read -p "请输入名字：" user**  
**[ -z $user ] && break**  
**a[$i]=$user**  
**let i++**  
**done**  
**echo "一共输入的名字有：${a[\*]}"**  
  
**expect预期交互**

**常见的expect指令：**

**1)定义环境变量：set 变量名 变量值**

**2)创建交互式进程：spawn 交互式命令行**

**3)触发预期交互：expect "预期会出现的文本关键词:" {send "发送的文本\r"}**

**4)在spawn建立的进程中允许交互指令：interact**  
**案例：**

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

**#! /bin/bash**  
**expect << EOF**

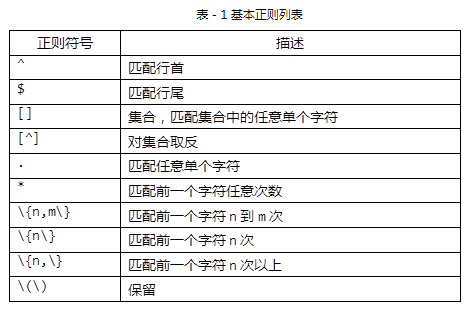
**set timeout 300**              **###修改expect的默认等待时间为300秒**  
**spawn ssh 192.168.4.207**  
**expect "password" {send "123456\n"}**  
**expect "#" {send "touch /hh.txt\n"}**  
**expect "#" {send "exit\n"}**                **####实际中，这一行的命令没有执行**  
**EOF**  
**备注： "#"  为期待出现的内容**

**\n   为回车的意思**

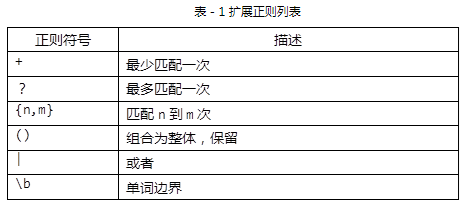
**#rm /root/.ssh/known\_hosts      此文件为第一次ssh时提示yes/no的**

**正则表达式(十几个特殊符号）**

**基本的正则列表**



**扩展的正则列表**



**基本正则：复杂，兼容性强**

**扩展正则：简单，兼容性差**

**\(\)或 （）  ：保留（复制的功能，把括号里的内容放到粘贴板里）**

**（a)(b)(c)    \1 \2 \3    :\1是粘贴a;\2是粘贴b;\3是粘贴c;**

**.\*     代表所有**

**^..\*.$ :  代表匹配所有字符！**