许多程序要求对shell脚本中的命令施加一些逻辑流程控制。有一类命令会根据条件使脚本跳过某些命令。这样的命令通常称为结构化命令（structured command）。

## 1.使用if-then-fi语句

最基本的结构化命令就是if-then-fi语句。if-then语句有如下格式。

if command

then

commands

fi

如果你在用其他编程语言的if-then语句，这种形式可能会让你有点困惑。

与其他编程语言区别

在其他编程语言中，if语句之后的对象是一个等式，这个等式的求值结果为TRUE或FALSE。但bash shell的if语句并不是这么做的。bash shell的if语句会运行if后面的那个命令。如果该命令的退出状态码是0（该命令成功运行），位于then部分的命令就会被执行。如果该命令的退出状态码是其他值，then部分的命令就不会被执行，bash shell会继续执行脚本中的下一个命令。fi语句用来表示if-then语句到此结束。

注意点：

1. command被正确执行之后，退出状态码为0时才会执行then后面的commands
2. if必须有fi结尾，否则报错

另一种格式：

if command; then

commands

fi

通过把分号放在待求值的命令尾部，就可以将then语句放在同一行上了，这样看起来更

像其他编程语言中的if-then语句。

## 2.if-then-else-fi语句

格式

if command

then

commands

else

commands

fi

当if语句中的命令返回退出状态码0时，then部分中的命令会被执行，这跟普通的if-then

语句一样。当if语句中的命令返回非零退出状态码时，bash shell会执行else部分中的命令。

## 3.嵌套if

格式：

if command1

then

commands

elif command2

then

more commands

fi

elif语句行提供了另一个要测试的命令，这类似于原始的if语句行。如果elif后命令的退

出状态码是0，则bash会执行第二个then语句部分的命令。使用这种嵌套方法，代码更清晰，逻辑更易懂。

**窍门**：在elif语句中，紧跟其后的else语句属于elif代码块。它们并不属于之前的

if-then代码块。

可以继续将多个elif语句串起来，形成一个大的if-then-elif嵌套组合。

if command1

then

command set 1

elif command2

then

command set 2

elif command3

then

command set 3

elif command4

then

command set 4

fi

每块命令都会根据命令是否会返回退出状态码0来执行。记住，bash shell会依次执行if语句，只有第一个返回退出状态码0的语句中的then部分会被执行。

## 4.test命令

缘起：if-then语句中，不能测试命令退出状态码之外的条件，

解决方法：test命令

test命令提供了在if-then语句中测试不同条件的途径。如果test命令中列出的条件成立，

test命令就会退出并返回退出状态码0。这样if-then语句就与其他编程语言中的if-then语句以类似的方式工作了。如果条件不成立，test命令就会退出并返回非零的退出状态码，这使得if-then语句不会再被执行。

test命令的格式非常简单。

格式：test condition

condition是test命令要测试的一系列参数和值。当用在if-then语句中时，test命令看

起来是这样的。

格式：

if testcondition

then

commands

fi

如果不写test命令的condition部分，它会以非零的退出状态码退出，并执行else语句块。

实例：使用test命令后面接字符变量时，如果该字符变量不为空，则返回0

bash shell提供了另一种条件测试方法，无需在if-then语句中声明test命令。

if [ condition]

then

commands

fi

方括号定义了测试条件。注意，第一个方括号之后和第二个方括号之前必须加上一个空格，

否则就会报错。

test命令可以判断三类条件：

 数值比较

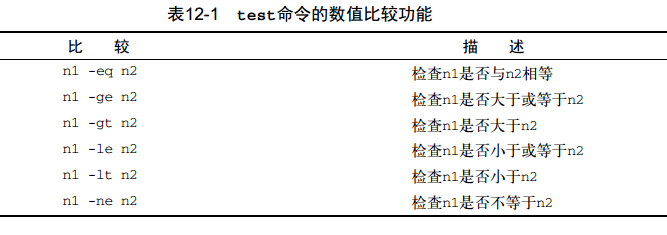
 字符串比较

 文件比较

### 4.1. 数值比较

test命令最常见的情形是对两个数值进行比较。表12-1列出了测试两个值时可用的条件

参数。



Tips: eq == equal ge == greater or equal gt == greater than le == less or equal

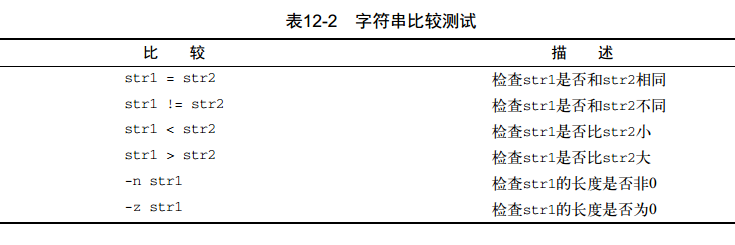
Lt == less than ne== not equal

注意事项：

**记住，bash shell只能处理整数。如果你只是要通过echo语句来显示这个结果**

### 4.2. 字符串比较

条件测试还允许比较字符串值。比较字符串比较烦琐，表12-2列出了可用的字符串比较功能。



#### 1. 字符串相等性

字符串的相等和不等条件不言自明，很容易看出两个字符串值是否相同。

记住，在比较字符串的相等性时，比较测试会将所有的标点和大小写情况都考虑在内。

#### 2. 字符串顺序

要测试一个字符串是否比另一个字符串大就是麻烦的开始。当要开始使用测试条件的大于或

小于功能时，就会出现两个经常困扰shell程序员的问题：

 A.大于号和小于号必须转义，否则shell会把它们当作重定向符号，把字符串值当作文件

名；

 B.大于和小于顺序和sort命令所采用的不同。

在比较测试中，大写字母被认为是小于小写字母的。但sort命令恰好相反。当你将同样的

字符串放进文件中并用sort命令排序时，小写字母会先出现。这是由各个命令使用的排序技术不同造成的。

比较测试中使用的是标准的ASCII顺序，根据每个字符的ASCII数值来决定排序结果。

sort命令使用的是系统的本地化语言设置中定义的排序顺序。对于英语，本地化设置指定了在排序顺序中小写字母出现在大写字母前。

重点说明：

test命令和测试表达式使用标准的数学比较符号来表示字符串比较，(>, <, =, !=, )

而用文本代码来表示数值比较(-eq, -ge,-le, -lt,-ne)。这个细微的特性被理解反了。

如果你对数值使用了数学运算符号，shell会将它们当成字符串值，可能无法得到正确的结果。

**补充说明： 字符串比较相等的时候使用=号 ，而不是其他编程语言中的==**

#### 3. 字符串大小

-n和-z可以检查一个变量是否含有数据。

注意点：特殊情况，如果变量val3未被定义过，则仍然将字符串长度为0

if [ -z $val3 ]

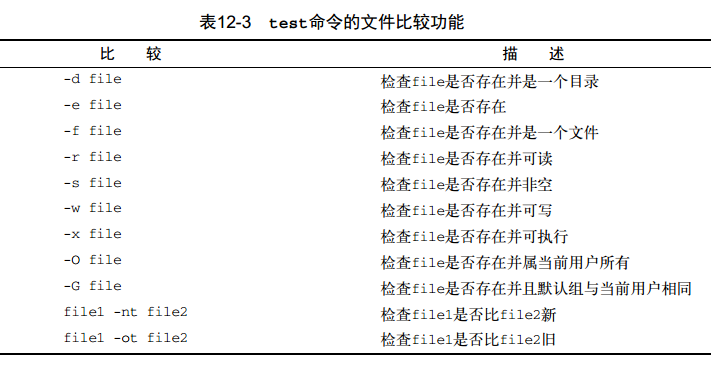
判断val3变量是否长度为0。这个变量并未在shell脚本中定义过，所以它的字符串长度仍然为0，尽管它未被定义过。

窍门 空的和未初始化的变量会对shell脚本测试造成灾难性的影响。如果不是很确定一个变量的内容，最好在将其用于数值或字符串比较之前先通过-n或-z来测试一下变量是否含有值。

### 4.3. 文件比较【重点】

比较测试很有可能是shell编程中最为强大、也是用得最多的比较形式。它允许你测

试Linux文件系统上文件和目录的状态。表12-3列出了这些比较。



这些测试条件使你能够在shell脚本中检查文件系统中的文件。它们经常出现在需要进行文件访问的脚本中。使用广泛，逐个看看。

#### 检查目录

说明：-d测试会检查指定的目录是否存在于系统中。如果你打算将文件写入目录或是准备切换到某个目录中，先进行测试总是件好事情。D= directory

#### 2. 检查对象是否存在【对象包括文件和目录】

-e比较允许你的脚本代码在使用文件或目录前先检查它们是否存在。E= exit

#### 3. 检查文件[ -f ]

-e比较可用于文件和目录。要确定指定对象为文件，必须用-f比较。

#### 4. 检查是否可读[ -r ]

在尝试从文件中读取数据之前，最好先测试一下文件是否可读。可以使用-r比较测试。

#### 5. 检查是否可写 [ -w ]

-w比较会判断你对文件是否有可写权限。

#### 6. 检查文件是否可以执行[ -w ]

-x比较是判断特定文件是否有执行权限的一个简单方法。虽然可能大多数命令用不到它，但

如果你要在shell脚本中运行大量脚本，它就能发挥作用

#### 7. 检查空文件[ -s ]

应该用-s比较来检查文件是否为空，尤其是在不想删除非空文件的时候。要留心的是，当

-s比较成功时，说明文件中有数据。

#### 8. 检查所属关系[ -O ]

-O比较可以测试出你是否是文件的属主。

#### 9. 检查默认属组关系[ -G ]

-G比较会检查文件的默认组，如果它匹配了用户的默认组，则测试成功。由于-G比较只会

检查默认组而非用户所属的所有组，这会叫人有点困惑。

#### 10. 检查文件日期

最后一组方法用来对两个文件的创建日期进行比较。这在编写软件安装脚本时非常有用。有

时候，你不会愿意安装一个比系统上已有文件还要旧的文件。

-nt比较会判定一个文件是否比另一个文件新。如果文件较新，那意味着它的文件创建日期更近。

-ot比较会判定一个文件是否比另一个文件旧。如果文件较旧，意味着它的创建日期更早。

## 5. 复合条件测试

f-then语句允许你使用布尔逻辑来组合测试。有两种布尔运算符可用：

 [ condition1 ] && [ condition2 ]

 [ condition1 ] || [ condition2 ]

第一种布尔运算使用AND布尔运算符来组合两个条件。要让then部分的命令执行，两个条件都必须满足。

**窍门** 布尔逻辑是一种能够将可能的返回值简化为TRUE或FALSE的方法。

第二种布尔运算使用OR布尔运算符来组合两个条件。如果任意条件为TRUE，then部分的命令就会执行。

## 6.if-then的高级特性

bash shell提供了两项可在if-then语句中使用的高级特性：

 用于数学表达式的双括号

 用于高级字符串处理功能的双方括号

### 1.双括号

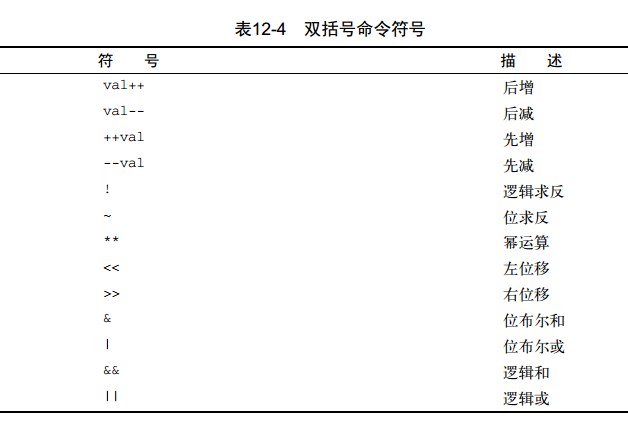
双括号命令允许你在比较过程中使用高级数学表达式。test命令只能在比较中使用简单的

算术操作。双括号命令提供了更多的数学符号，这些符号对于用过其他编程语言的程序员而言并不陌生。双括号命令的格式如下：

((expression))

expression可以是任意的数学赋值或比较表达式。除了test命令使用的标准数学运算符，

表12-4列出了双括号命令中会用到的其他运算符。



可以在if语句中用双括号命令，也可以在脚本中的普通命令里使用来赋值。

注意，不需要将双括号中表达式里的大于号转义。这是双括号命令提供的另一个高级特性。

### 2.使用双方括号

双方括号命令提供了针对字符串比较的高级特性。双方括号命令的格式如下：

[[ expression]]

双方括号里的expression使用了test命令中采用的标准字符串比较。但它提供了test命

令未提供的另一个特性——模式匹配（pattern matching）。

说明 双方括号在bash shell中工作良好。不过要小心，不是所有的shell都支持双方括号。

在模式匹配中，可以定义一个正则表达式来匹配字符串值。

## 7.case--esac命令

有了case命令，就不需要再写出所有的elif语句来不停地检查同一个变量的值了。case命

令会采用列表格式来检查单个变量的多个值。

case variablein

pattern1| pattern2) commands1;;

pattern3) commands2;;

\*) default commands;;

esac

case命令会将指定的变量与不同模式进行比较。如果变量和模式是匹配的，那么shell会执行为该模式指定的命令。可以通过竖线操作符在一行中分隔出多个模式模式。星号会捕获所有与已知模式不匹配的值。这里有个将if-then-else程序转换成用case命令的例子。