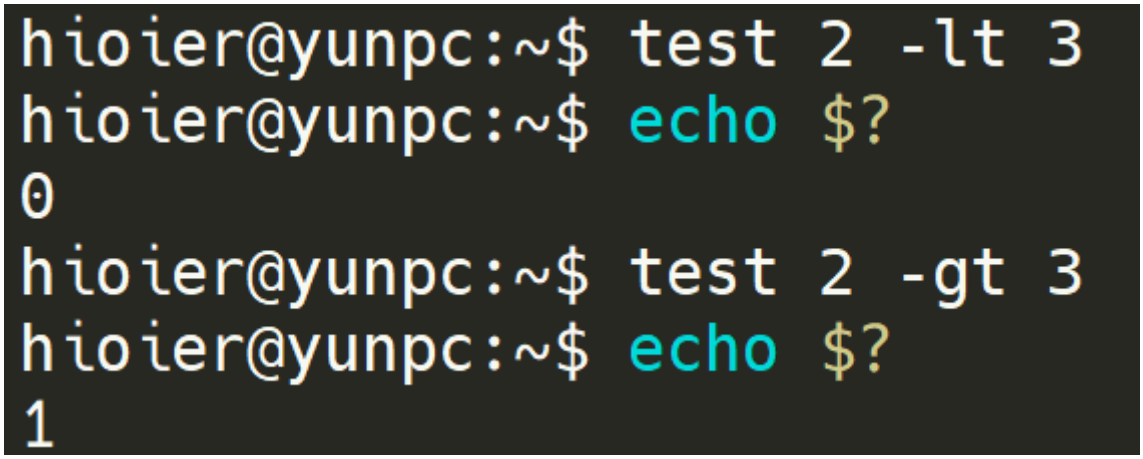
test命令

Shell中的 test 命令用于检查某个条件是否成立，它可以进行数值、字符和文件三个方面的测试。

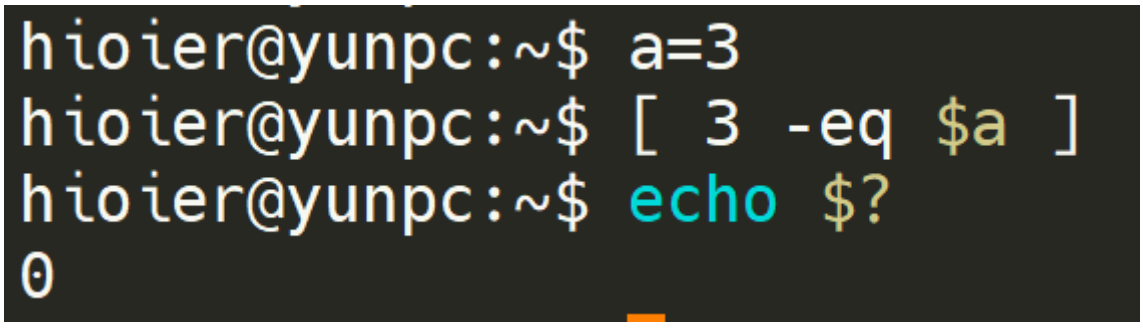
test命令用exit code返回结果，而不是使用stdout。0表示真，非0表示假。

数值测试

参数	说明
-eq	等于则为真
-ne	不等于则为真
-gt	大于则为真
-ge	大于等于则为真
-lt	小于则为真
-le	小于等于则为真



test也可以使用[]代替



字符串测试

参数	说明
=	等于则为真
!=	不相等则为真
-z 字符串	字符串的长度为零则为真
-n 字符串	字符串的长度不为零则为真

```
hioier@yunpc:~$ name="blackcat"
hioier@yunpc:~$ test $name = "blackcat"
hioier@yunpc:~$ echo $?
0
hioier@yunpc:~$ test $name ≠ "blackcat"
hioier@yunpc:~$ echo $?
1
hioier@yunpc:~$ test -z $name
hioier@yunpc:~$ echo $?
1
hioier@yunpc:~$ test -n $name
hioier@yunpc:~$ echo $?
0
```

文件测试

参数	说明
-e 文件名	如果文件存在则为真
-r 文件名	如果文件存在且可读则为真
-w 文件名	如果文件存在且可写则为真
-x 文件名	如果文件存在且可执行则为真
-s 文件名	如果文件存在且至少有一个字符则为真
-d 文件名	如果文件存在且为目录则为真
-f 文件名	如果文件存在且为普通文件则为真
-c 文件名	如果文件存在且为字符型特殊文件则为真
-b 文件名	如果文件存在且为块特殊文件则为真

```
hioier@yunpc:~/scripts$ test -e test01.sh
hioier@yunpc:~/scripts$ echo $?
0
```

逻辑运算符&&和||

```
&& 表示与，|| 表示或
二者具有短路原则：
expr1 && expr2：当expr1为假时，直接忽略expr2
expr1 || expr2：当expr1为真时，直接忽略expr2
表达式的exit code为0，表示真；为非零，表示假。
hioier@yunpc:~/scripts$ test -e test.sh && echo "exist" || echo "Not exist"
Not exist
```

if语句

单分支

```
if condition
then
    command1
    command2
    ...
    commandN
fi
a=3
b=4

if [ $a -lt $b ] && [ $a -gt 2 ]
then
    echo "YES"
fi
```

双分支

```
if condition
then
    command1
    command2
    ...
    commandN
else
    command
fi
a=3
b=4

```

```
if [ $a -gt $b ] && [ $a -gt 2 ]
then
    echo "YES"
else
    echo "NO"
fi
```

多分支

```
if condition1
then
    command1
elif condition2
then
    command2
else
    commandN
fi
```

另外，Shell 还提供了与(-a)、或(-o)、非(!)三个逻辑操作符用于将测试条件连接起来，其优先级为：！最高， -a 次之， -o 最低。

```
declare -i score=69

if test $score -ge 90 -a $score -le 100
then
    echo "A"
elif test $score -ge 80 -a $score -lt 90
then
    echo "B"
elif test $score -ge 60 -a $score -lt 80
then
    echo "C"
else
    echo "D"
fi
```

case语句

case ... esac 为多选择语句，与其他语言中的 switch ... case 语句类似，是一种多分支选择结构，每个 case 分支用右圆括号开始，用两个分号 ;; 表示 break，即执行结束，跳出整个 case ... esac 语句，esac（就是 case 反过来）作为结束标记。

可以用 case 语句匹配一个值与一个模式，如果匹配成功，执行相匹配的命令。

case ... esac 语法格式如下：

```
case $变量名称 in
    值1)
        command1
        command2
        ...
```

```
        commandN
        ;;  # 类似于C/C++中的break
    值2)
        command1
        command2
        ...
        commandN
        ...
        ;;
    *)  # 类似于C/C++中的default
        command1
        command2
        ...
        commandN
        ...
        ;;
esac
declare -i n
read -p "请输入1-4之间的数字：" n

case $n in
    1) echo "选择1"
    ;;
    2) echo "选择2"
    ;;
    3) echo "选择3"
    ;;
    4) echo "选择4"
    ;;
    *) echo "请输入正确的数字"
    ;;
esac
```