* Shell运算：

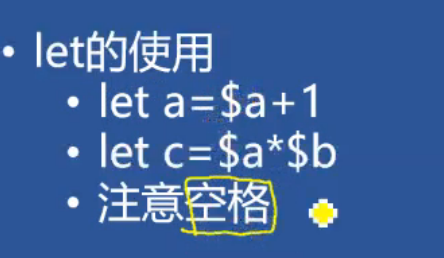
1. Expr 可以把字符串处理为数字运算

注意乘法，用\\* 或者“\*”，除法不需要用转义 直接/

注意空格：+ / \\* “\*”两边都需要空格



1. Let 用let不能有空格，且let\*不用转义，且要用赋值符

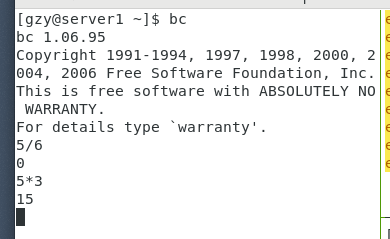


1. 双括号：让双括号中间进行算数运算

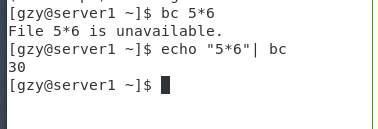
（（a=a+1））

例子 echo $((a=(a+5)\*(j=4+3))) a初始为2 结果为49

1. 直接bc进入环境：

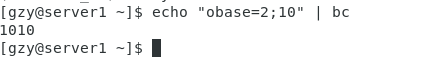


可以再此环境中进行运算

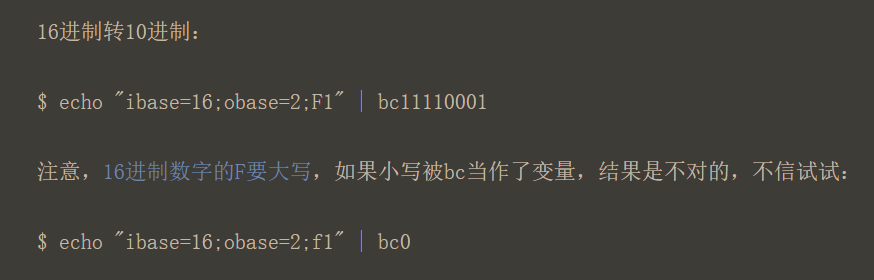


Bc可以处理文件，所以可以用管道变成文件来算

Bc还可以进行进制转换 如



Obase表示输出显示是2进制，同样也可以指明输入是几进制（默认是10进制）



* 分支与循环

Test command用于判别，与if连用

* 比较string test a ->/</…… b

= > >= < <= != z n(z表示zero判断长度是否为零，为0则表达式成立，反之不成立，n表示nonze)

注意判断符前后都要用空格

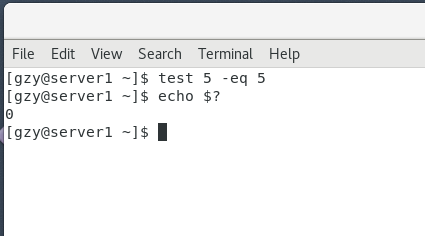
* 比较数字

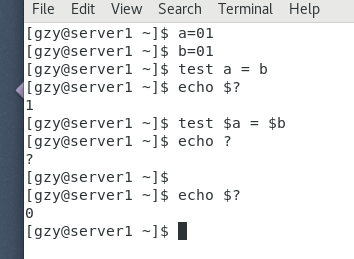
- 加上eq lt le gtg e ne（eq相等 lt less than le 小于等于）

* 最后的比较结果要查看 $?,,

$?的结果表示命令是否成功（也就是对应条件表达式是否正确），成功为0

不成功为1，反过来

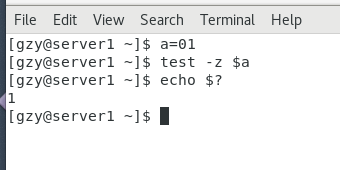


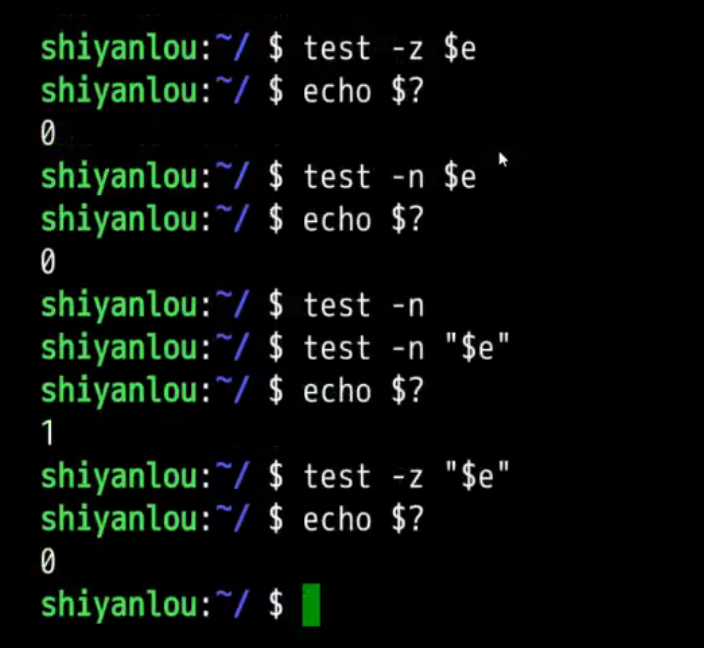


变量直接比较毫无意义 比较的是 a b这两个字符，所以要$取值在比较

* -z -n的用处

判断字符串长度是否为0，如果是0，表达式成立，返回0，否则返回1



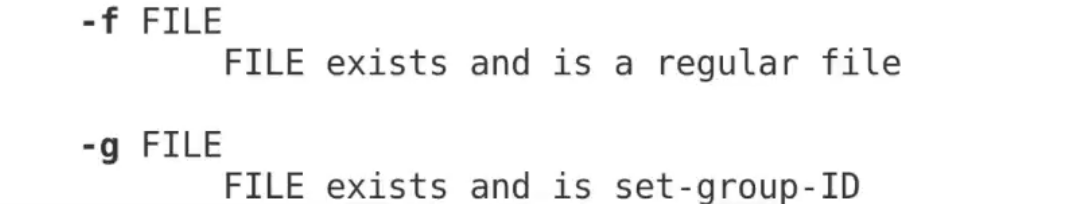


-z -n特别要注意，如果直接 test -n/-z $e，如果$e是空的那么系统接收到的就直接是 test -n/-z 无论怎么执行都是正确的，所以要区分就需要加双引号，让系统知道后有字符串

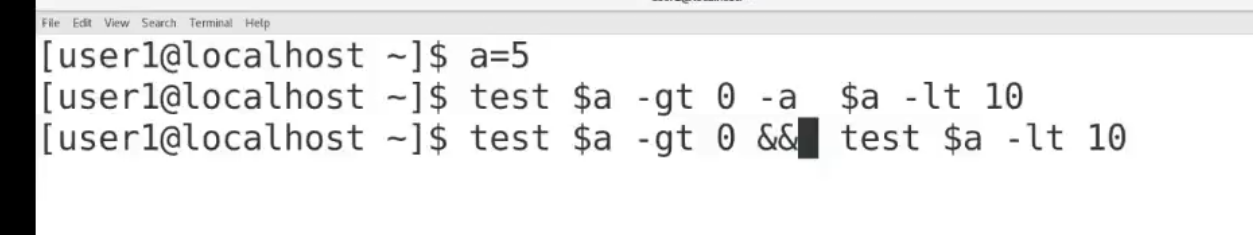
* 文件和目录的判断

-r -w -x -d -f判断三权限 判断目录 判断文件

格式 Test -f file



* 复合

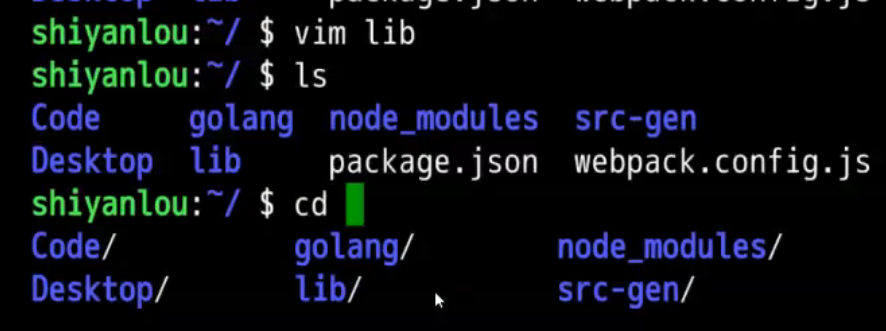


两个是等价的。

区别只是 -a(与)/-o（或者）是连接两个条件表达式，此处判断a是否>0 <10

&& /|| 连接两个命令。

* 查看目录下内容



Ls 可以看 cd 加tab补全可以看

Vim目录名也可以看