3. 数据的计算

+ , - , \\* , / , %

运算命令: expr

`expr $变量 + $变量`

变量=`expr $变量 + $变量`

变量=$((变量 + 变量))

**练习6：**

**从键盘输入两个数，输出所有加减乘除的计算结果。**

4. 判断的对象

(1)字符串

= , != , -z（长度为零）, -n（长度不为零）

e.g. if [[ -z “$string” ]] if [[ “$A” == “abcdef” ]]

(2)数字

-eq(等于), -ne（不等于）, -gt（大于）, -lt（小于）, -ge（大于或等于）,

-le（小于或等于）

e.g. if [ $data -ge 256 ]

(3)文件属性

-d, -f, -r, -w, -x

e.g. if [ -d “$path” ]

说明：

* **字符串比较通常**用双中括号**[[ ]]**
* **算数比较**用单中括号**[ ]——左右留空格。**
* **算数运算**用双小括号**(( ))：只要括号中的运算符、表达式符合C语言运算规则，都可用在$((exp))中，甚至是三目运算符。双括号中的变量可以不使用$符号前缀，括号内支持多个表达式用逗号分开。只要括号中的表达式符合C语言运算规则，比如可以直接使用for((i=0;i<5;i++))。如果不使用双括号, 则为for i in `seq 0 4` 或者for i in {0..4}。再如可以直接使用if(($i<5)), 如果不使用双括号, 则为if [ $i -lt 5 ]。**
* **可以使用[[ ... ]]条件判断结构，而不是[ ... ]，能够防止脚本中的一些逻辑错误。比如使用 &&、||、> 操作符能够正常存在于[[ ]]条件判断结构中**，但是**如果出现在[ ]结构中的话，会报错**。**比如，可以直接使用if [[ $a != 1 && $a != 2 ]]。如果不使用双括号, 则为if [ $a -ne 1 -a $a -ne 2 ]**，或者if [ $a != 1 -a $a != 2 ]。

5. 控制语句

1. **判断语句**

if(空格)[(空格)表达式1(空格)＝(空格)表达式2(空格)]

if(空格)[(空格)判断的对象(空格)]

then

命令

fi

-a 且的关系

-o 或的关系

**Example:**

if [ $data -ge 256 -a $data -lt 512 ]

then

echo $data

fi

if [ 判断的对象 ]

then

命令

else

命令

fi

if [ 判断的对象 ]

then

命令

elif [判断的对象]

then

命令

.......

elif [判断的对象]

then

命令

else

命令

fi

if [ condition1 ]; then

# statements if condition1 is true

elif [ condition2 ]; then

# statements if condition2 is true

else

# statements if both conditions are false

fi

**练习7:**

从键盘输入一个成绩，用if-lese语句判断成绩的等级

[80-100] A

[60-80) B

[0-60) C

如果成绩非法不是[0-100],则提示用户输入的成绩是非法的成绩

1. **case 语句**

case 变量的内容 in

匹配项1)

命令

;;

匹配项2)

命令

;;

\*)

命令

;;

esac

**Example:**

case $number in

6|7|8)

echo “……”

;;

4|5|6)

echo “……”

;;

\*)

;;

esac

**练习8:**

**从键盘输入一个成绩，用case语句判断成绩的等级 （结题思路:将成绩/10）**

**[80-100] A 8，9，10**

**[60-80 ) B 6，7**

**[0-60 ) C 0，1，2，3，4，5**

**如果成绩非法不是[0-100],则提示用户输入的成绩是非法的成绩**