

Expressions

An expression is a grouping of variables, constants and operators that require evaluation

Two types of expressions in BASH

Arithmetic, Conditional

Examples of an arithmetic expression:

```
((z = $a ** 2 + $b))      # new fangled way
let "z = $a ** 2 + $b"   # old fashioned way
```

Note that the use of *double parentheses* is easier to implement...don't need *let* or *double-quotes*...closer to C/C++ programming style...I use the term "closer" *very loosely* here...

Examples of a conditional expression:

```
if [[ $a -ge $b ]] ...   # double-square bracket
if [ $a -ge $b ] ...     # single-square bracket
if test "$a -ge $b" ...  # old fashioned way
```

Note that all three are functionally identical. I recommend the square bracket syntax...select single or double and stay with it.

BIG NOTE: THERE MUST BE A SPACE BETWEEN THE LEFT BRACKET AND THE FIRST CHARACTER OF THE EXPRESSION, AND A SPACE BETWEEN THE LAST CHARACTER OF THE EXPRESSION AND THE RIGHT BRACKET!

Various Operators Used with ((...))

Operator	Name	Example
var++	increment	((i = \$x++))
var--	decrement	((i = \$x--))
**	exponentiation	((i = \$y ** 3))
*	multiply	((i = \$y * \$z))
/	integer division	((i = \$y / \$z))
+	addition	((i = \$y + \$z))
-	subtraction	((i = \$y - \$z))
%	modulus	((i = \$y % \$z))
==	equality	((\$y == 3))
!=	not equal	((\$y != 3))
>=	greater than or equal to	((\$y >= 3))
<=	less than or equal to	((\$y <= 3))
>	greater than	((\$y > 3))
<	less than	((\$y < 3))
&&	logical AND	((\$y == 5 && \$x < 4))
 	logical OR	((\$y == 5 \$x < 4))

Notes

Single equal sign is the *assignment* operator

Double equal sign is the *equality comparison* operator

Integer division means $5 / 2 = 2$...not 2.5

Modulus yields the *remainder* of integer division

$$5 \% 2 = 1 \dots 20 \% 3 = 2$$

Division by ZERO is undefined

Order of Operations

Operators
()
var++, var--
** , * , / , %
+, -
<=, >=, <, >
==, !=
&&

Notes:

Evaluations are made left to right

Use of internal parentheses is highly recommended to avoid ambiguity

Example:

`((y = 12 * 3 - 2))` yields 34

however:

`((y = 12 * (3 - 2)))` yields 12