
# V19.4.5

## Expressions

## Table of Contents

Expressions....3
Basic operators....4
Functions....4
Trigonometry....5
Comparison....6
Boolean logic....7
Conditional....8
Comments....8

# Expressions

The expression evaluator supports the following fundamental mathematical operations.

| | |
|---|---|
| Basic operators: | +, -, *, /, ^ |
| Functions: | min, max, avg, sum, abs, ceil, floor, round, roundn, exp, log, log10, logn, root, sqrt, clamp, inrange, sgn, erf, erfc |
| Trigonometry: | sin, cos, tan, acos, asin, atan, atan2, cosh, cot, csc, sec, sinh, tanh, rad2deg, deg2rad, deg2grad, grad2deg, hyp |
| Comparison: | =, ==, <>, !=, <, <=, >, >= |
| Boolean logic: | and, or, xor, not, nand, nor, shr, shl, like |
| Conditional: | if-then-else |

**Types**

The following are short examples of the types of mathematical expressions that can be parsed and evaluated.

numeric

| | |
|---|---|
| sqrt(1 – (x^2)) | Analogic → DOUBLE |
| log10(-1,sin(2 * pi * x) + cos(y / 2 * pi),+1) | Analogic → DOUBLE |
| sin(2 * x) | Analogic → DOUBLE |
| if (((x + 2) == 3) and ((y + 5) <= 9), 1 + w, 2 / z) | Boolean/Analogic → DOUBLE |
| if ((a or b) and c, 5, 2) | Boolean/Analogic → INTEGER |
| inrange(-2,m,+2) == if(((-2 <= m) and (m <= +2)),1,0) | Boolean → BOOLEAN |
| ((1/1)*(1/2)+(1/3))-(1/4)^(1/5)+(1/6)-((1/7)+(1/8)*(1/9)) | Analogic → DOUBLE |
| a * exp(2 * t) + c | Analogic → DOUBLE |
| 25x^5 - 35x^4 - 15x^3 + 40x^2 - 15x + 1 | Analogic → DOUBLE |

strings

| | |
|---|---|
| x <= 'abc123' and (y in 'AString') or ('1x2y3z' != z) | String → BOOLEAN |
| (x like '*123*') | String → BOOLEAN |

same result with different sintax allowed

| | |
|---|---|
| 2x + 3y + 4z + 5w | 2 * x + 3 * y + 4 * z + 5 * w |
| 3(x + y) / 2 + 1 | 3 * (x + y) / 2 + 1 |
| (x + y)3 + 1 / 4 | (x + y) * 3 + 1 / 4 |
| (x + y)z + 1 / 2 | (x + y) * z + 1 / 2 |
| sin(x/3.14)cos(2y) + 1) | (sin(x / 3.14) * cos(2 * y) + 1 |

notes

Instead of x, y, a, b, ecc... you have to imagine tags names from PLCs or other derived tags.

**e.g.** ('tag_a:5' * **exp**(2 * 'tag_b:23') + 'tag_c')

('tag_a:4' **and** (('tag_b,5' **or** 'tag_c,10') **and** (**not**('tag_d') **and** 'tag_e:45'))

**Parsing tag names used in derived variables or in other places**

- Access to values which belong to an array of elements

'pump_40:0' it indicates the element 0 of tag named 'pump_40'

- Access to values which belong to a single element.

'pump_40' it indicates the value of tag named 'pump_40'

This syntax always refers to the element number 0, even in presence of tag with multiple elements.

**IMPORTANT !** An expression can be both simple as a number, or complex as the functions above.
In other words, it means that everywhere it's needed, you can write a constant or a complex function.
You can write functions with many levels of brackets.

Example: '((('TabF:5'^2) * cos('TagC:67')) / (TagA:0' + 2) – 'TagB')

# Basic operators

| | |
|---|---|
| + Sum | Example:'TagA:0' + 'TagB:3' |
| - Subtraction | Example:'TagA:0' - 'TagB:3' |
| * Multiplication | Example:'TagA:0' * 'TagB:3' |
| / Division | Example: 'TagA:0' / 'TagB:3' |
| ^ Exponent | Example:'TagA:0'^2 |

# Functions

min
Returns the minimum value among the supplied ones. Max 6 values at a time.
Example: min('TagA', 'TagB:2', 'TagC:0', …)
max
Returns the maximum value among the supplied ones. Max 6 values at a time.
Example: max('TagA', 'TagB:2', 'TagC:0', ...)
avg
Returns the average value among the supplied ones. Max 6 values at a time.
Example: avg('TagA', 'TagB:2', 'TagC:0', …)
sum
Returns the sum of supplied values. Max 6 values at a time.
Example: sum('TagA', 'TagB:2', 'TagC:0', …)
abs
Returns the absolute value of the supplied one.
Example: abs('TagA:2')
ceil
Rounds x upward, returning the smallest integral value that is not less than x.
Example: ceil('TagA:2')
floor
Rounds x downward, returning the largest integral value that is not greater than x.
Example: floor('TagA:2')
round
Returns the integral value that is nearest to x, with halfway cases rounded away from zero.
Example: round('TagA:2')
roundn
Returns the float value that is nearest to x.n, where n are decimals you need, with halfway cases rounded away from zero.
Example: roundn('TagA:2', 3)
exp
Returns the base-e exponential function of x, which is e raised to the power x: e^x.
Example: exp('TagA:2')
log

The natural logarithm is the base-e logarithm: the inverse of the natural exponential function (exp).
Example: log('TagA:2')

log10

Returns the common (base-10) logarithm of x.
Example: log10('TagA:2')

logn

Returns the common (base-n) logarithm of x.
Example: logn('TagA:2', 7)

sqrt

Returns the square root of x.
Example: sqrt(TagA:2)

clamp

Return a value clamped within the given range.
Example: clamp(-5.0, 'TagA:2', 5.0)

sgn

Return -1 if the value is less than 0 otherwise it returns 1 if the value is greater than 0.
Example: sgn('TagA:2')

erf

Returns the error function value for x.
Example: erf('TagA:2')

erfc

Returns the complementary error function value for x.
Example: erfc('TagA:2')

# Trigonometry

atan

Returns the value of the arc tangent of x, expressed in radians.
Example: atan('TagA:0')

asin

Returns the value of the arc tangent of x, expressed in radians.
Example: asin('TagA:0')

acos

Returns the value of the arc cosine of x, expressed in radians.
Example: acos('TagA:0')

tan

Returns the tangent of an angle of x radians.
Example: tan('TagA:0')

cos

Returns the cosine of an angle of x radians.
Example: cos('TagA:0')

sin

Returns the sine of an angle of x radians.
Example: sin('TagA:0')

atan2

Returns the value of the arc tangent of y/x, expressed in radians.
Example: atan2('TagA:0', 'TagB:0')

cosh
Returns the hyperbolic cosine of x radians.
Example: cosh('TagA:0')
cot
Returns the cotangent of x radians.
Example: cot('TagA:0')
sinh
Returns the hyperbolic sine of x radians.
Example: sinh('TagA:0')
tanh
Returns the hyperbolic tangent of x radians.
Example: tanh('TagA:0')
hyp
Returns the Hypotenuse of a right-angled triangle.
Example: hyp('TagA:0', 'TagB:0')
rad2deg
Transforms rad to deg.
Example: rad2deg('TagA:0')
deg2rad
Transforms deg to rag.
Example: deg2rad('TagA:0')
grad2deg
Transforms grad to deg.
Example: grad2deg('TagA:0')
deg2grad
Transforms deg to grad.
Example: deg2grad('TagA:0')
csc
Cosecant.
Example: csc('TagA:0')
sec
Secant.
Example: sec('TagA:0')

# Comparison

=
Returns 1 if the two values are equal.
Example: ('TagA:2' = 'TagB:3')
<>
Returns 1 if the two values are not equal.
Example: ('TagA:2' <> 'TagB:3')
<
Returns 1 if the first value is less than the second one.
Example: ('TagA:2' < 'TagB:3')
<=
Returns 1 if the first value is equal to or less than the second one.

Example: ('TagA:2' <= 'TagB:3')

>

Returns 1 if the first value is greater than the second one.

Example: ('TagA:2' > 'TagB:3')

>=

Returns 1 if the first value is equal to or greater than the second one.

Example: ('TagA:2' >= 'TagB:3')

# Boolean logic

and

1 and 1 = 1
0 and 1 = 0
1 and 0 = 0
0 and 0 = 0

Example: 'TagA:0,2' and 'TagB:3'

or

1 or 1 = 1
0 or 1 = 1
1 or 0 = 1
0 or 0 = 0

Example: 'TagA:0,2' or 'TagB:3'

nand

1 and 1 = 0
0 and 1 = 1
1 and 0 = 1
0 and 0 = 1

Example: 'TagA,2' nand 'TagB:3'

nor

1 or 1 = 0
0 or 1 = 0
1 or 0 = 0
0 or 0 = 1

Example: 'TagA,2' nor 'TagB:3'

xor

1 or 1 = 0
0 or 1 = 1
1 or 0 = 1
0 or 0 = 0

Example: 'TagA,2' xor 'TagB:3'

not

Returns the opposite value of a boolean expression.

Example: not('TagA,2' and 'TagB:3')

shr

Returns the given value after a shift-right action of its original bits using the specified number of bits.

Example: shr('TagA:2', 2)

shl

Returns the given value after a shift-left action of its original bits using the specified number of bits.

Example: shl('TagB', 3)

inrange
Returns 1 if the value is within the given range.
Example: inrange(-10, 'TagA:23' , 10)

like - (string like mask) is case sensitive
Returns 1 if the string mask matches with the first string. * and ? can be used as mask.
Example: ("'TagA:0'" like "*eScada*")

ilike - (string like mask) is not case sensitive
Returns 1 if the string mask matches with the first string. * and ? can be used as mask.
Example: ("'TagA:0'" like "*eScada*")

in - (string1 in string2) is case sensitive
Returns 1 if the string1 is contained in string2.
Example: ("'TagA:0'" in "'TagB:10'")

# Conditional

if-then-else
Returns the value depending on the condition result.
Example: if (('TagA:0' or 'TagB:3'), 'TagC:3' * 5, 'TagC:3' * 8)

# Comments

C or C++ style comments can be used

examples:

```
/*
  my own comment ….
  bla, bla, bla ….
*/

/* my own comment …. */

// my own comment ….

//
// my own comment ….
// bla, bla, bla ….
//
```