Numeric:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x + y | x – y | x \* y | x / y | x % y | x \*\* y |
| x ~ y | x++ | ++x | x-- | --x |  |

Compare:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x == y | x === y | x != y | x !== y | x < y | x > y |
| x <=> y | ?x |  |  |  |  |

Ternary:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x => y : z | x <= y : z |  |  |  |  |

Access:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x.y | x->y | &x | x.&y | x->&y |  |

Logical:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| !x | x && y | x || y | x ^^ y |  |  |

Bitwise:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x << y | x >> y | !!x | x & y | x | y | x ^ y |
| x !& y | x !| y |  |  |  |  |

Assignment:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x = y | x =+ y | x =– y | x =\* y | x =/ y | x =% y |
| x =^ y | x =~ y | x =! y | x =<< y | x =>> y | x =!! y |
| x =& y | x =| y | x =^ y | x =!& y | x =!| y | x ??= y |

You can override operators behaviour or define your own following the syntax below, however, ternary and access operators cannot be overridden. Only binary custom operators are supported.

*override operator++(void) -> any {*

*<? pre-increment operator redefinition here ?>*

*}*

*operator %%=!//\* (any var) -> any {*

*<? Custom operator definition here ?>*

*}*

*operator feet (any var) -> any {*

*<? Custom operator definition here ?>*

*}*

ASCII characters only are allowed for custom operators. Unlike symbolic operators, literal operators must always be separated by whitespaces. The operator name cannot contain a mixture of letters and symbols, you either use one or another.

**Operator priority:**

From left to right. Assignment operators have more priority than other operators though, yet they still follow the rule. Parentheses have the highest priority and follow the rule.

Literal custom operators must not start with a digit nor contain whitespaces or tab characters.