Standard notation for predicates and functions is prefix notation:

\[ p(X, f(a)) \]

But sometimes one wants to have infix (or postfix) notation:

\[ 2 + 3 \quad \text{instead of} \quad +(2, 3) \]

\[ \leftarrow \quad \rightarrow \]

they are considered syntactically identical.

This is possible because + has been declared as an infix operator.

\[ ?- \quad 2+3 \quad =\quad +(2,3). \]

true

To declare operators, one has to add a directive of the following form to the program:

\[ :- \quad \text{op} \left( \text{Precedence, Type, Name(s)} \right). \]

Directives are queries written in the program.
They are proved when loading the program. This is interesting if the predicates in the directive perform side-effects. Here, the side effect is that certain symbols can be used in infix-notation etc. after having evaluated this predicate.

For $+, -, \&$, there are the following pre-defined directives:

$$
:\text{-op}(500, \text{ yfx }, \text{ [t, -j] }).
\text{-op}(400, \text{ yfx }, \text{ \# }).
$$

**Precedence** (number between 0 and 1200). States which operation has stronger binding than another. A small precedence means strong binding.

$1 + 2 \& 3$ should stand for $1 + (2 \& 3)$

$5 - 4 - 3$ should stand for $5 - (4 - 3)$ or $(5 - 4) - 3$.

**should it associate to the right or to the left?**
Types:

$f \times$, $g \times$, $\times f y$

are types for

binary infix operators

$f \times$, $f y$

types for prefix operators

$\times f$, $g f$

types for postfix operators

$\hat{f}$ operator

$x \hat{=}$ argument whose precedence is smaller than prec. of $f$

$y \hat{=}$ is smaller or equal to the precedence of $f$.

Precedence of an argument: Prec. of its leading operator

Precedence of standard (non-operator) pred. and fact. symbols is 0.

Arguments in (...) also have prec. 0.

"Functors" →

So $\hat{f} \times y$ means that arguments with the same

preced. as $f$ can only occur on $f$'s

left-hand side.

$5 - 4 - 3$ Can this stand for $5 - (4 - 3)$?
No, because then the right argument would also have prec. 500.

So it stands for

What does $1 + 2 \times 3 + 4$ stand for?

Both + and $\times$ have type $\text{yfx}$.

$$(1 + (2 \times 3)) + 4$$

$\text{yfx}$: association to the left

$\text{xfy}$: right
\( x f x \) : no association

\[
(1 + 2 + 3) \text{ would not be allowed}
\]

Overloading of operators is possible.

Ex: In addition to the binary —

there is also a unary —.

\( \vdash \text{op}(200, f1, -) \).

\(-2 -3\) stands for \(\frac{200}{3} \).

\(\text{i.e.}\) for \((-2)-3\)

Operators can also be used for a simple form of natural language processing.

"Was" is used as a binary operator

in infix notation.

It should not have any association.
Laura was young was beautiful does not make sense.

\[ \Rightarrow \text{Type } x \times f x \]

"of": binary operator in infix notation should associate to the right

"Secretary of son of John" stands for

"Secretary of (son of John)" \[ \Rightarrow \text{Type } x f y \]

"the": unary operator, no association

"the the dog" makes no sense \[ \Rightarrow \text{Type } f x \]

\[ \text{:- op}(300, x f x, \text{was}). \]

\[ \text{:- op}(250, x f y, \text{of}). \]

\[ \text{:- op}(200, f x, \text{the}). \]

"of" binds stronger than "was".

"the" binds stronger than "of".

Laura was the secretary of the head of the department.

Laura
  \[ \text{was (of (the (secr. of (...)))).} \]

The
  \[ \text{of the secr. of the dept.} \]
¿Who was the secretary of the head of the department.
Who = Laura.

¿Laura was What.
What = the secretary of the head of the department.
¿Who was the secretary of the head of What.
Who = Laura
What = the department