$(x \text{ op } y) \text{ op } z$ left assoc $+, -, \forall$

$x \text{ op } (y \text{ op } z)$ right assoc $\wedge$

$2 \div 3 \div 5$

$2/(3/5)$ $\dfrac{2}{(3/5)}$ or $\dfrac{2/3}{5}$?

$\uparrow$ right $\uparrow$ left

Pony lang

$(((x \text{ op}_1 \ y) \text{op}_2 \ z) \text{ op}_3 \ w)$ uniform

$((2+3) \times 5)$

Perl6 / rakulang

defining custom operators.

Ⓕ Ⓧ

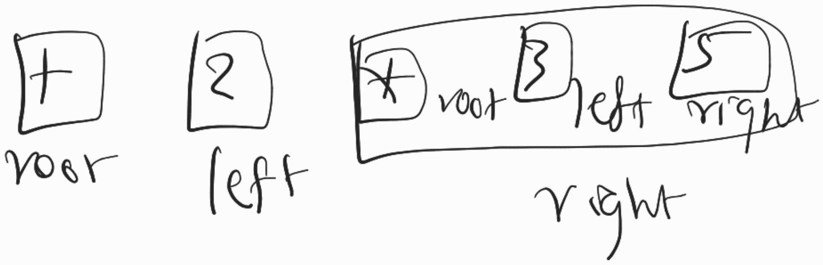
Ocaml

define custom operators

$\underline{+}\$$ $\underline{*}\$$

Scheme/Lisp

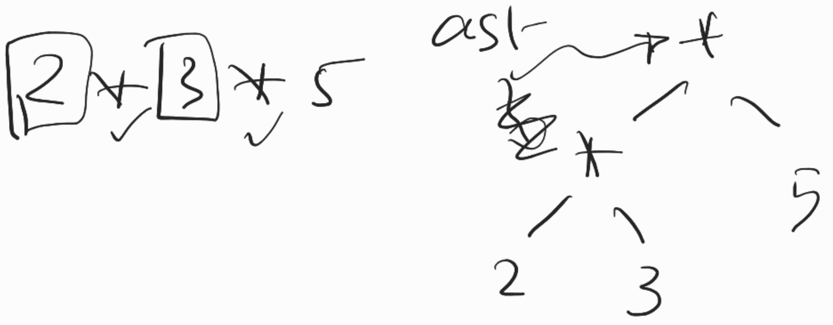
(+ 2 (* 3 5))



(root first-subtree second-subtree ----)

Forth/factor

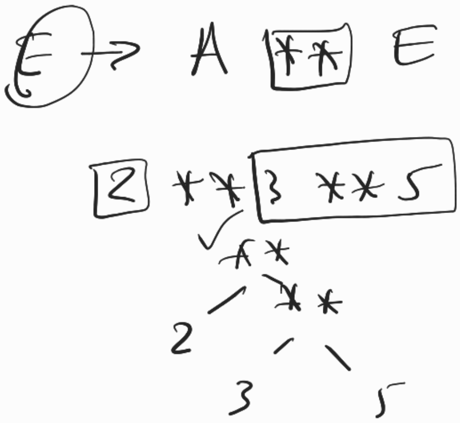
2 3 5 * +



Parse_exp

⋮

ast



Operator Precedence/ Precedence Climbing Parser

Pratt Parsing

$E \to T + E \mid T$

$T \to F * A \mid A$

$A \to$ Int | Float ~~number~~ $\mid (\overset{?\ A}{E})$

Parse_atom

⋮

Parse_add

(2+3) > (5+6)

Decimal points ~ IntToken

FloatToken

type(2) ~ int

type(2.0) ~ float

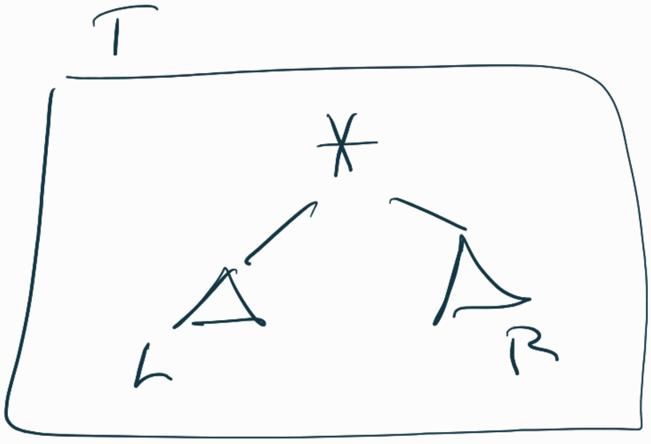
type(2*2) ~ int

type(2*2.0) ~ float

type(4/2) ~ float

type(5/2) ~ float

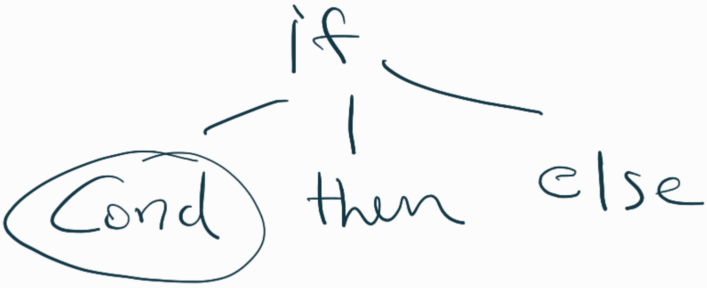
type(2/0) ~ should be float



"hello" * 2

"hello" * "world"

type(T) is computable from label of root and type(L) and type(R)



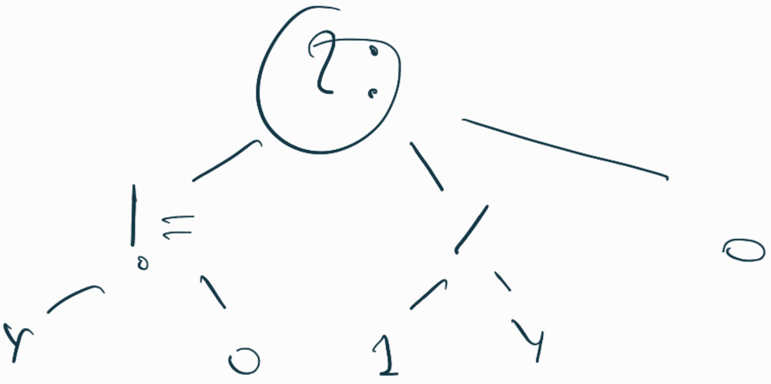
How can we know this expr is fine?

type =? bool

z = if x > y then x else y

$z = (x > y) ? x : y$

$z = y != 0 ? 1/y : 0$



- Add string literals to the language
- add comparisons
- add if-then-else

logical? if (a && b)

```
if (!a)          c
    b          }
  }            if (a)
if (a)             if (b)
    pass               c
else
    b
```

$\boxed{\text{"hello"}}$

StringToken

2fast

$\boxed{\text{"hello"}}$ $\boxed{+}$ $\boxed{\text{"world"}}$

how many tokens?

"Hello \"world\""

"Hello \\world" ⇝ Hello \world.

$\div$ $2 \div 3/5 \sim \frac{2}{(3/5)}$

$2/3/5 \sim (2/3)/5$

Unicode has invisible characters.

$$ab \sim a \times b$$

$$\downarrow$$

$$a \underbrace{\qquad\qquad}_{\text{invisible mult}} b$$

$$b^2 - 4ac$$

invisible plus

$$5\,3/4$$

$$1 - -1 \qquad\qquad 1 - \sim 1$$

$-$ binary subtraction

$\sim$ unary negation

$$- - 2 \qquad\qquad - \sqcup {}^{-}2$$