Lambda Calculus

Abstract Syntax

$X = \{\mathtt{x}, \mathtt{y}, \mathtt{z}, \dots\}$ — variables

$C = \{\mathtt{X}, \mathtt{Y}, \mathtt{Z}, \dots\}$ — constructors

$$\begin{array}{rcll} \Lambda = & X & \text{— variable} \\ | & C & \text{— constructor} \\ | & \lambda X.\Lambda & \text{— abstraction} \\ | & \Lambda\Lambda & \text{— application} \end{array}$$

Concrete Syntax

Variables represented as identifiers, started with a lowercase letter; constructors — as identifiers, started with an uppercase letter; "λ" repsresented by backslash; application is left-associative, abstraction spans to the right as far as possible.

Comments: "`-`" — comments out the rest of the line; "`(*`" and "`*)`" — regular multiline comments, which can be nested.

Macro definitions (substituted in-place, recursion is not allowed):

$$\begin{array}{llllll} \mathtt{def} & X_1 & = & \Lambda_1 & \mathtt{in} \\ \mathtt{def} & X_2 & = & \Lambda_2 & \mathtt{in} \\ & & \dots & & \\ \Lambda & & & & \end{array}$$

Examples:

- `(\x.x)\x.x`
- `\x y z . x z (y z)`
- `def pair = \x y s.s x y in pair A B`