

# Lambda Calculus

## Abstract Syntax

$X = \{x, y, z, \dots\}$  — variables

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

$\Lambda =$	$X$	— variable
	$  \quad C$	— constructor
	$  \quad \lambda X. \Lambda$	— abstraction
	$  \quad \Lambda \Lambda$	— application

## Concrete Syntax

Variables represented as identifiers, started with a lowercase letter; constructors — as identifiers, started with an uppercase letter; “ $\lambda$ ” represented 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):

```
def X1 =  $\Lambda_1$  in
def X2 =  $\Lambda_2$  in
...
 $\Lambda$ 
```

Examples:

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