

LAMBDA

MODULE LAMBDA

SYNTAX $Type ::= \text{int}$
 | bool
 | $Type \rightarrow Type$
 | $(Type)$ [\[](#)[bracket](#)([bracket](#)())]

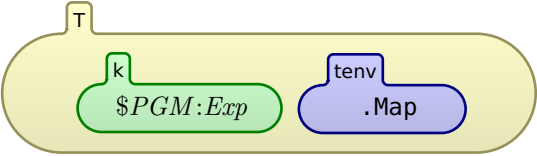
SYNTAX $Exp ::= Id$
 | $\text{lambda } Id : Type . Exp$
 | $Exp \ Exp$ [\[](#)[strict](#)([strict](#)())]
 | (Exp) [\[](#)[bracket](#)([bracket](#)())]

SYNTAX $Exp ::= Type$

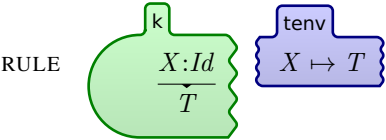
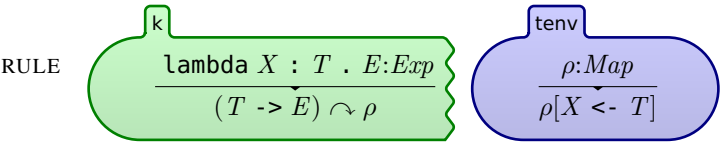
SYNTAX $Variable ::= Id$

SYNTAX $KResult ::= Type$

CONFIGURATION:



SYNTAX $Exp ::= Exp \rightarrow Exp$ [\[](#)[strict](#)([strict](#)())]



RULE
$$\frac{(T1 \rightarrow T2) \quad T1}{T2}$$

SYNTAX $Exp ::= Int$
 | $Bool$
 | $Exp * Exp$ [\[](#)[strict](#)([strict](#)())]
 | Exp / Exp [\[](#)[strict](#)([strict](#)())]
 | $Exp + Exp$ [\[](#)[strict](#)([strict](#)())]
 | $Exp <= Exp$ [\[](#)[strict](#)([strict](#)())]

RULE
$$\frac{\text{---:Int}}{\text{int}}$$

RULE
$$\frac{\text{---:Bool}}{\text{bool}}$$

RULE
$$\frac{\text{int} * \text{int}}{\text{int}}$$

RULE
$$\frac{\text{int} / \text{int}}{\text{int}}$$

RULE
$$\frac{\text{int} + \text{int}}{\text{int}}$$

RULE
$$\frac{\text{int} <= \text{int}}{\text{bool}}$$

SYNTAX $Exp ::= \text{if } Exp \text{ then } Exp \text{ else } Exp$ [\[](#)[strict](#)([strict](#)())]

RULE
$$\frac{\text{if bool then } T:Type \text{ else } T}{T}$$

SYNTAX $Exp ::= \text{let } Id : Type = Exp \text{ in } Exp$

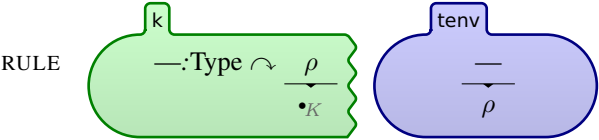
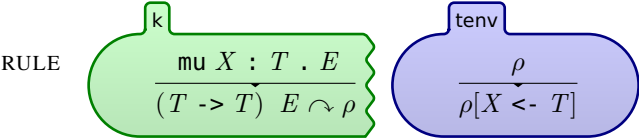
RULE
$$\frac{\text{let } X : T = E \text{ in } E'}{(\text{lambda } X : T . E') \ E}$$

[\[](#)[macro](#)([macro](#)())]

SYNTAX $Exp ::= \text{letrec } Id : Type \ Id : Type = Exp \text{ in } Exp$
 | $\text{mu } Id : Type . Exp$

RULE
$$\frac{\text{letrec } F : T1 \ X : T2 = E \text{ in } E'}{\text{let } F : T1 = \text{mu } F : T1 . \text{lambda } X : T2 . E \text{ in } E'}$$

[\[](#)[macro](#)([macro](#)())]



END MODULE