

LAMBDA

MODULE LAMBDA

SYNTAX $Exp ::= Id$
 | $\lambda Id.Exp$
 | $Exp\ Exp$ [strict]
 | (Exp) [bracket]

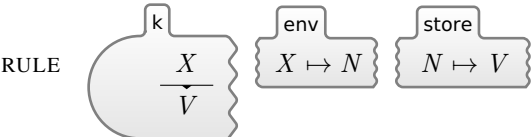
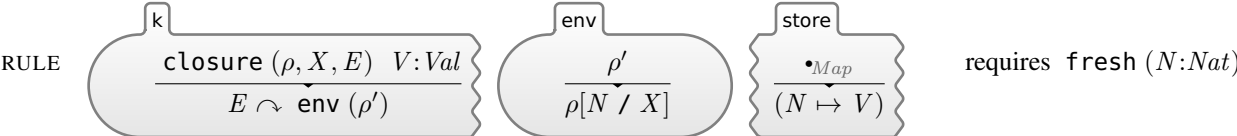
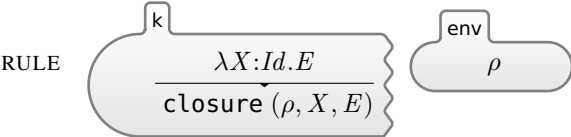
CONFIGURATION:



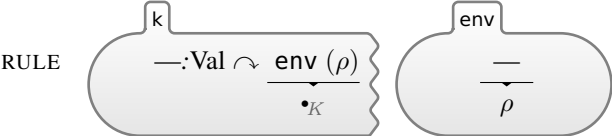
SYNTAX $Val ::= \text{closure } (Map, Id, Exp)$

SYNTAX $Exp ::= Val$

SYNTAX $KResult ::= Val$

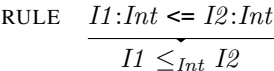
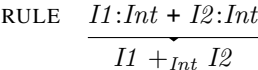
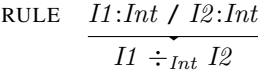
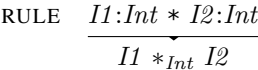


SYNTAX $K ::= \text{env } (Map)$

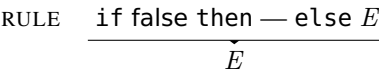
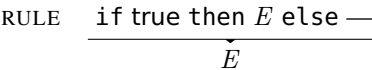


SYNTAX $Val ::= Int$
 | $Bool$

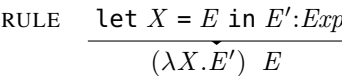
SYNTAX $Exp ::= Exp * Exp$ [strict]
 | Exp / Exp [strict]
 | $Exp + Exp$ [strict]
 | $Exp <= Exp$ [strict]



SYNTAX $Exp ::= \text{if } Exp \text{ then } Exp \text{ else } Exp$ [strict(1)]

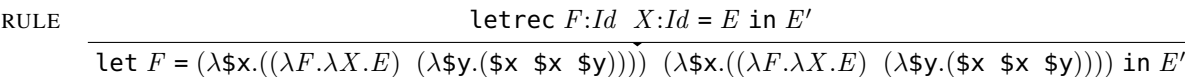


SYNTAX $Exp ::= \text{let } Id = Exp \text{ in } Exp$



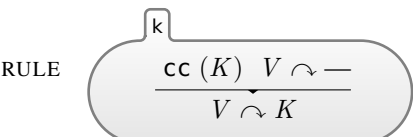
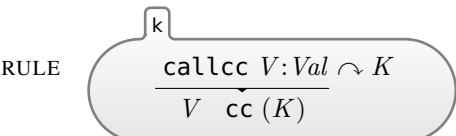
SYNTAX $Exp ::= \text{letrec } Id\ Id = Exp \text{ in } Exp$

SYNTAX $Id ::= \$x$
 | $\$y$



SYNTAX $Exp ::= \text{callcc } Exp$ [strict]

SYNTAX $Val ::= \text{cc } (K)$



END MODULE