LAMBDA

MODULE LAMBDA

SYNTAX *Exp* ::= *Id*
| λ*Id*.*Exp*
| *Exp* *Exp* [strict]
| (*Exp*) [bracket]

CONFIGURATION:

T: k: $\$PGM{:}Exp$ | env: $\bullet_{Map}$ | store: $\bullet_{Map}$

SYNTAX *Val* ::= `closure` (*Map*, *Id*, *Exp*) [klabel('closure)]

SYNTAX *Exp* ::= *Val*

SYNTAX *KResult* ::= *Val*

RULE k: $\dfrac{\lambda X{:}Id.E}{\texttt{closure}\ (\rho, X, E)}$ … env: ρ [structural]

RULE k: $\dfrac{\texttt{closure}\ (\rho, X, E)\quad V{:}Val}{E \curvearrowright \rho'}$ … env: $\dfrac{\rho'}{\rho[X \texttt{ <- } N]}$ store: … $\dfrac{\bullet_{Map}}{(N{:}Int \mapsto V)}$ …

RULE k: $\dfrac{X}{V}$ … env: … $X \mapsto N$ … store: … $N \mapsto V$ …

RULE k: $\_{:}\mathrm{Val} \curvearrowright \dfrac{\rho}{\bullet_K}$ … env: $\dfrac{\_}{\rho}$ [structural]

SYNTAX *Val* ::= *Int*
| *Bool*

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

RULE $\dfrac{I1 * I2}{I1 *_{Int} I2}$

RULE $\dfrac{I1 \ /\ I2}{I1 \div_{Int} I2}$

RULE $\dfrac{I1 + I2}{I1 +_{Int} I2}$

RULE $\dfrac{I1 \texttt{ <= } I2}{I1 \leq_{Int} I2}$

SYNTAX *Exp* ::= `if` *Exp* `then` *Exp* `else` *Exp* [strict(1)]

RULE $\dfrac{\texttt{if true then } E \texttt{ else } \_}{E}$

RULE $\dfrac{\texttt{if false then } \_ \texttt{ else } E}{E}$

SYNTAX *Exp* ::= `let` *Id* = *Exp* `in` *Exp*

RULE $\dfrac{\texttt{let } X = E \texttt{ in } E'{:}Exp}{(\lambda X.E')\ \ E}$ [macro]

SYNTAX *Exp* ::= `letrec` *Id* *Id* = *Exp* `in` *Exp*

SYNTAX *Id* ::= `$x`
| `$y`

RULE $\dfrac{\texttt{letrec } F{:}Id\ \ X{:}Id = E \texttt{ in } E'}{\texttt{let } F = (\lambda \texttt{\$x}.((\lambda F.\lambda X.E)\ \ (\lambda \texttt{\$y}.(\texttt{\$x \$x \$y}))))\ \ (\lambda \texttt{\$x}.((\lambda F.\lambda X.E)\ \ (\lambda \texttt{\$y}.(\texttt{\$x \$x \$y})))) \texttt{ in } E'}$ [macro]

END MODULE