$[\![v]\!]\eta\kappa = \kappa(\eta(v))$

$[\![e\ e']\!]\eta\kappa = [\![e]\!]\eta\,(\lambda f.\ [\![e']\!]\eta\,(\lambda z.\ f\,z\,\kappa))_{fun}$

Here $(-)_\theta$ is similar to $(-)_{\theta*}$ that we looked at before, but it doesn't deal with $\perp$ and errors.

That is, given $f \in V_\theta \to V_*$,

$$f_\theta \in V \to V_*$$

$$f_\theta(a) = \begin{cases} \langle 1, \text{typeerr} \rangle & \text{if } \neg(\exists i, b \text{ s.t } b \in V_\theta \wedge a = \psi(\langle i, b\rangle)) \\ f(b) & \text{if } \exists i, b \text{ s.t } b \in V_\theta \wedge a = \psi(\langle i, b\rangle) \end{cases}$$

$[\![\lambda v.e]\!]\eta\kappa = \kappa\left(\psi\langle 2, \lambda a.\lambda\kappa'.\ [\![e]\!][\eta \mid v:a]\kappa'\rangle\right)$

$[\![n]\!]\eta\kappa = \kappa\,(\psi\langle 0, n\rangle)$

$[\![-e]\!]\eta\kappa = [\![e]\!]\eta\,(\lambda\bar{z}.\ \kappa\,(\psi\langle 0, -z\rangle))_{int}$

$[\![e_0 + e_1]\!]\eta\kappa = [\![e_0]\!]\eta\,(\lambda\bar{z}.\ [\![e_1]\!]\eta\,(\lambda\bar{z}'.\ \kappa\,(\psi\langle 0, \bar{z}+\bar{z}'\rangle))_{int})_{int}$

$[\![\text{if } e \text{ then } e' \text{ else } e'']\!]\eta\kappa$
$= [\![e]\!]\eta\,(\lambda b.\ \text{if } b \text{ then } [\![e']\!]\eta\kappa \text{ else } [\![e'']\!]\eta\kappa)_{bool}$

$[\![\text{true}]\!]\eta\kappa = \kappa\,(\psi\langle 1, tt\rangle)$

$[\![\langle e_0, e_1\rangle]\!]\eta\kappa = [\![e_0]\!]\eta\,(\lambda a_0.\ [\![e_1]\!]\eta\,(\lambda a_1.\ \kappa\,(\psi\langle 3, \langle a_0, a_1\rangle\rangle)))$

$[\![e.k]\!]\eta\kappa = [\![e]\!]\eta\,(\lambda t.\ \text{if } k \in \text{dom}\,t \text{ then } \kappa(t_k) \text{ else typerr})_{tuple}$

$[\![\text{letrec } v = \lambda u.e \text{ in } e']\!]\eta\kappa = [\![e']\!][\eta \mid v : Y_F]\kappa$

$F \in [V_{fun} \xrightarrow{}_c V_{fun}]$

$F(f_0)(a)(\kappa') = [\![e]\!][\eta \mid u:a \mid v:f_0]\kappa'$

We omit a few definitions. You can find them in p254-255 of the textbook.

⑤ Note that whenever we interpret an expression that includes more than one immediate subexpression, such as $e_0 + e_1$, we construct a new continuation for the subexpressions that will not be evaluated next, such as $\left(\lambda\bar{z}.\ [\![e_1]\!]\eta\ (\lambda\bar{z}'.\ \kappa(\psi\langle 0, \bar{z}+\bar{z}'\rangle))_{int}\right)_{int}$.

Intuitively, this means that the semantics is very explicit about evaluation order.

⑥ This semantics can be expressed as syntactic transformation call cps transformation. Let $[\![-]\!]_d$ be the direct semantics that we studied in the previous chapter. Consider an expression e and a fresh variable v_{cont}. Then, this transformation has the following property:

$$[\![e]\!]\eta\kappa \text{ "=" } [\![\text{cps}(e, v_{cont})]\!]_d[\eta \mid v_{cont} : \kappa]$$

(= equal when no errors)

As mentioned before, this cps transformation is often used by a compiler as a preprocessing step.