0 Formalization of a CESK* machine with basic Scheme features.

$$
\begin{aligned}
\varsigma \in \Sigma &= \mathsf{Exp} \times \mathsf{Env} \times \mathsf{Kont} \\
op \in \mathit{Primitive} &\quad \text{The set of primitves} \\
\mathit{clo} \in \mathit{Clo} &= (\lambda\ (x...)\ e) \times \mathsf{Env} \\
\mathsf{Bool} &::= \texttt{\#t} \mid \texttt{\#f} \\
v \in \mathsf{AExp} &= \mathit{Clo} \ + \ \mathsf{Bool} \ + \ \mathbb{Z} \\
x \in \mathsf{Var} &\quad \text{A set of identifiers} \\
e \in \mathsf{Exp} &::= x \mid v \mid (\lambda\ (x...)\ e) \\
&\quad \mid (\texttt{if}\ e\ e\ e) \\
&\quad \mid (\texttt{let}\ (x\ e)\ e) \\
&\quad \mid (\texttt{prim}\ \mathit{op}\ e\ e...) \\
&\quad \mid (e\ e\ ...) \\
a, b, c \in \mathit{Addr} &\quad \text{A set of addresses} \\
\rho \in \mathsf{Env} &= \mathsf{Var} \rightharpoonup \mathit{Addr} \\
\sigma \in \mathsf{Store} &= \mathit{Addr} \rightharpoonup \mathsf{AExp} \\
\mathit{done} &= \mathsf{Val}* \\
\mathit{todo} &= \mathsf{Exp}* \\
\kappa \in \mathsf{Kont} &= \mathbf{mt} \mid \mathbf{appk}(\mathit{done}, \mathit{todo}, \rho, a) \\
&\quad \mid \mathbf{ifk}(e, e, \rho, a) \\
&\quad \mid \mathbf{letk}(x, e, \rho, a)
\end{aligned}
$$

$$
\begin{aligned}
\mathit{alloc} &: \Sigma \to \mathit{Addr} \\
\mathit{alloc}(\varsigma) &= \text{an unallocated address}
\end{aligned}
$$

Our transition function is of type

$$(\Sigma \times \mathsf{Store}) \to (\Sigma \times \mathsf{Store})$$

$(\varsigma \times \sigma) \to (\varsigma' \times \sigma)$, where $\kappa = \sigma(a), b = alloc(\varsigma)$
proceed by matching on ς

$\langle x, \rho, a\rangle$	$\langle v, \rho, a\rangle$ where $v = \sigma(\rho(x))$
$\langle(\lambda\ (x...)\ e), \rho, a\rangle$	$\langle \mathbf{clo}((\lambda\ (x...)\ e), \rho), \rho, a\rangle$
$\langle(\texttt{if}\ e_c\ e_t\ e_f), \rho, a\rangle$	$\langle e_c, \rho, b\rangle$ $\sigma[b \mapsto \mathbf{ifk}(e_t, e_f, \rho, a)]$
$\langle(\texttt{let}\ (x\ e_x)\ e_b), \rho, a\rangle$	$\langle e_x, \rho, b\rangle$ $\sigma[b \mapsto \mathbf{letk}(x, e_b, \rho, a)]$
$\langle(\texttt{prim}\ op\ e_0\ es...), \rho, a\rangle$ trick for 0 arg prims? What would I make ctrl? Also should I merge primk and appk	$\langle e_0, \rho, b\rangle$ $\sigma[b \mapsto \mathbf{primk}(op, [], es, \rho, a)]$
$\langle(e_f\ es...), \rho, a\rangle$	$\langle e_f, \rho, b\rangle$ $\sigma[b \mapsto \mathbf{appk}([], e_s, \rho, a)]$
$\langle v, \rho, a\rangle$ match on κ below	
mt	ς
$\mathbf{ifk}(e_t, e_f, \rho', c)$ where $v = \texttt{\#f}$	$\langle e_f, \rho', c\rangle$
$\mathbf{ifk}(e_t, e_f, \rho', c)$ where $v \neq \texttt{\#f}$	$\langle e_t, \rho', c\rangle$
$\mathbf{letk}(x, e_b, \rho', c)$	$\langle e_b, \rho'[x \mapsto b], c\rangle$ $\sigma[b \mapsto v]$
$\mathbf{primk}(op, done, [\,], \rho', c)$	$\langle v', \rho', c\rangle$ $v' = op$ applied to $(done +\!\!+ [v])$
$\mathbf{primk}(op, done, (h :: t), \rho', c)$	$\langle h, \rho', b\rangle$ $\sigma[b \mapsto \mathbf{primk}(op, done +\!\!+ [v], t, \rho', c)]$
$\mathbf{appk}((\mathbf{clo}((\lambda\ (xs...)\ e_b), \rho'') :: vs),$ $[\,], \rho', c)$	$\langle e_b, \rho''[xs_0 \mapsto b_0 ... xs_i \mapsto b_i], c\rangle$ $vs = vs +\!\!+ [v]$ $\sigma[b_0 \mapsto vs_0 ... b_i \mapsto vs_i]$
$\mathbf{appk}(done, h :: t, \rho', c)$	$\langle h, \rho', b\rangle$ $\sigma[b \mapsto \mathbf{appk}(done +\!\!+ [v], t, \rho', c)]$

1 Formalization of an aCESK* machine with basic Scheme features.