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

Syntax:

$$
\begin{aligned}
e \in \mathsf{Exp} &::= x \mid \text{æ} \mid lam \\
&\mid (\texttt{if}\ e\ e\ e) \\
&\mid (\texttt{let}\ (x\ e)\ e) \\
&\mid (\texttt{prim}\ op\ e\ e...) \\
&\mid (e\ e\ ...) \\
\text{æ} \in \mathsf{AExp} &= Clo\ +\ \mathsf{Bool}\ +\ \mathbb{Z}
\end{aligned}
$$

This doesnt feel right $\Uparrow$ Because Clo cant be constructed syntactically...

$$
\begin{aligned}
lam \in \mathsf{Lam} &::= (\lambda\ (x...)\ e) \\
x \in \mathsf{Var} & \quad \text{A set of identifiers} \\
\mathsf{Bool} &::= \texttt{\#t} \mid \texttt{\#f} \\
done &= \mathsf{AExp*} \\
todo &= \mathsf{Exp*}
\end{aligned}
$$

Semantics:

$$
\begin{aligned}
\varsigma \in \Sigma &= \mathsf{Exp} \times \mathsf{Env} \times \mathsf{Kont} \\
op \in Primitive & \quad \text{The set of primitves} \\
clo \in Clo &= lam \times \mathsf{Env} \\
\rho \in Env &= \mathsf{Var} \rightharpoonup Addr \\
\sigma \in \mathsf{Store} &= Addr \rightharpoonup \mathsf{AExp} \\
\kappa \in Kont &= \mathbf{mt} \mid \mathbf{appk}(done, todo, \rho, a) \\
&\mid \mathbf{ifk}(e, e, \rho, a) \\
&\mid \mathbf{letk}(x, e, \rho, a)
\end{aligned}
$$

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

Transition Function:

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

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

$\langle x, \rho, a \rangle$	$\langle æ, \rho, a \rangle$ where $æ = \sigma(\rho(x))$
$\langle lam, \rho, a \rangle$	$\langle \mathbf{clo}(lam, \rho), \rho, a \rangle$
$\langle (\texttt{if}\ e_c\ e_t\ e_f), \rho, a \rangle$	$\langle e_c, \rho, b \rangle$ $\sigma[b \mapsto \mathbf{ifk}(e_t, e_f, \rho, a)]$
$\langle (\texttt{let}\ (x\ e_x)\ e_b), \rho, a \rangle$	$\langle e_x, \rho, b \rangle$ $\sigma[b \mapsto \mathbf{letk}(x, e_b, \rho, a)]$
$\langle (\texttt{prim}\ op\ e_0\ es...), \rho, a \rangle$	$\langle e_0, \rho, b \rangle$ $\sigma[b \mapsto \mathbf{appk}([op], es, \rho, a)]$
$\langle (e_f\ es...), \rho, a \rangle$	$\langle e_f, \rho, b \rangle$ $\sigma[b \mapsto \mathbf{appk}([], e_s, \rho, a)]$
$\langle æ, \rho, a \rangle$ match on κ below	
mt	$\langle æ, \rho, a \rangle$
$\mathbf{ifk}(e_t, e_f, \rho', c)$ where $æ = \texttt{\#f}$	$\langle e_f, \rho', c \rangle$
$\mathbf{ifk}(e_t, e_f, \rho', c)$ where $æ \neq \texttt{\#f}$	$\langle e_t, \rho', c \rangle$
$\mathbf{letk}(x, e_b, \rho', c)$	$\langle e_b, \rho'[x \mapsto b], c \rangle$ $\sigma[b \mapsto v]$
$\mathbf{appk}(op :: æ_s, [], \rho', c)$	$\langle æ', \rho', c \rangle$ $æ' = op$ applied to $(æ_s +\!\!+ [æ])$
$\mathbf{appk}((\mathbf{clo}((\lambda\ (xs...)\ e_b), \rho'') :: æ_s),$ $[\,], \rho', c)$	$\langle e_b, \rho''[xs_0 \mapsto b_0 ... xs_i \mapsto b_i], c \rangle$ $æ_s = æ_s +\!\!+ [æ]$ $\sigma[b_0 \mapsto æ_{s0} ... b_i \mapsto æ_{si}]$
$\mathbf{appk}(done, e_h :: e_t, \rho', c)$	$\langle e_h, \rho', b \rangle$ $\sigma[b \mapsto \mathbf{appk}(done +\!\!+ [æ], e_t, \rho', c)]$

$\hat{\mathcal{A}}$

1 Formalization of an Abstract CESK* machine with basic Scheme features.