$$
\begin{array}{rcll}
x & ::= & \text{ALPHANUMERIC} \mid \star\text{ALPHANUMERIC} & \textit{variables} \\[2pt]
\overset{\star}{\ell} & ::= & \ell \mid * & \textit{general labels} \\[2pt]
\ell & & & \textit{labels} \\[2pt]
T & ::= & [t, \ldots] & \textit{stack} \\[2pt]
t & ::= & \langle \eta, \overset{\star}{\ell}, S \rangle & \textit{stack frames} \\[2pt]
S & ::= & [s, \ldots] & \textit{programs} \\[2pt]
s & ::= & \ell : \overset{\star}{\ell} : d & \textit{clauses} \\[2pt]
d & ::= & x = e \mid \texttt{return } x \mid \texttt{goto } \ell \mid \texttt{goto } \ell \texttt{ if not } x & \textit{directives} \\
  &   & \mid \texttt{raise } x \mid \texttt{catch } x \mid \texttt{pass} & \\[2pt]
B & ::= & \{x \mapsto m, \ldots\} & \textit{bindings} \\[2pt]
H & ::= & \{m \mapsto v, \ldots\} & \textit{heap} \\[2pt]
v & ::= & \mathbb{Z} \mid \mathbb{S} \mid [m, \ldots] \mid (m, \ldots) \mid B \mid F \mid M \mid * & \textit{values} \\[2pt]
e & ::= & \mathbb{Z} \mid \mathbb{S} \mid \texttt{None} \mid x \mid \texttt{def } x(x, \ldots) = \{S\} \mid x(x, \ldots) \mid x.x \mid [x, \ldots] \mid (x, \ldots) & \textit{expressions} \\[2pt]
Y & ::= & [y, \ldots] & \textit{microcode stack} \\[2pt]
Z & ::= & [z, \ldots] & \textit{microcode literal stack} \\[2pt]
y & ::= & \textsc{Store} \mid \textsc{Wrap} \mid \textsc{Bind} \mid \textsc{LookUp} \mid \textsc{List } n \mid \textsc{Tuple } n & \textit{microcode instructions} \\
  &   & \mid \textsc{Advance} \mid \textsc{Pop} \mid \textsc{Push } S \mid \textsc{Raise} \mid \textsc{Goto } \ell \mid \textsc{Gotoifn } \ell & \\
  &   & \mid \textsc{Call } n \mid \textsc{GetCall } n \mid \textsc{Convert } n \mid \textsc{Retrieve} & \\
  &   & \mid \textsc{AllocNameError} \mid \textsc{AllocTypeError} \mid \textsc{AllocAttrError} & \\[2pt]
z & ::= & x \mid m \mid v & \textit{microcode literals} \\[2pt]
\overset{\star}{m} & ::= & m \mid \eta \mid * & \textit{general memory locations} \\[2pt]
\eta, m & ::= & \text{<ADDRESS>} & \textit{memory locations} \\[2pt]
F & ::= & \langle \eta, \texttt{def } (x, \ldots) \to S \rangle \mid \mathfrak{F} \mid \mathfrak{M} & \textit{general functions} \\[2pt]
M & ::= & \langle m, F \rangle & \textit{general methods} \\[2pt]
\mathfrak{F}, \mathfrak{M} & ::= & \mathfrak{CallFunc} \mid \mathfrak{Type} & \textit{magic functions} \\[2pt]
n & & & \textit{integers}
\end{array}
$$

Figure 1: Expression Grammar

Definition 0.1. *Initialization*

$$
\begin{aligned}
H_{\text{INIT}} &= \{\eta_{\text{INIT}} \mapsto B_{\text{INIT}}, m_{None} \mapsto *, m_{AttrError} \mapsto \{\ \}, m_{FunType} \mapsto \{\ \}\} \\
B_{\text{INIT}} &= \{\textsc{AttributeError} \mapsto m_{AttrError}, \textsc{FunctionType} \mapsto m_{FunType}\} \\
t_{\text{INIT}} &= \langle \ell_{\text{INIT}}, S_{\text{INIT}} \rangle \\
T_{\text{INIT}} &= [t_{\text{INIT}}] \\
\textsc{GetAttribute} &= \langle \eta,\ \textit{def } (o, a) \to \$1 = o.a; \rangle \\
\textsc{GetCall} &= \langle \eta,\ \textit{def } (f) \to \rangle
\end{aligned}
$$

("if $1 == None then return getattribute(o.__class__,a) else return $1" – TC)
(todo: add builtin mappings $m \mapsto F$ – TC)

STORE v
$$\frac{m \notin H \qquad H' = H[m \mapsto v]}{Z \,||\,[v, \text{STORE}] \,||\, Y, T, H \longrightarrow^1 Z \,||\,[m] \,||\, Y, T, H'}$$

WRAP m
$$\frac{v = \text{GETOBJ}(H, m)}{Z \,||\,[m, \text{WRAP}] \,||\, Y, T, H \longrightarrow^1 Z \,||\,[v] \,||\, Y, T, H}$$

BIND m TO x
$$\frac{B = H[\eta] \qquad B' = B[x \mapsto m] \qquad H' = H[\eta \mapsto B']}{Z \,||\,[m, x, \text{BIND}] \,||\, Y, T, H \longrightarrow^1 Z \,||\, Y, T, H'}$$

ADVANCE
$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} : d \qquad \ell \overset{s}{\blacktriangleleft} \overset{\star''}{\ell}}{Z \,||\,[\text{ADVANCE}] \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Z \,||\, Y, [\langle \eta, \overset{\star''}{\ell}, S \rangle] \,||\, T, H}$$

POP
$$\frac{}{Z \,||\,[\text{POP}] \,||\, Y, t \,||\, T, H \longrightarrow^1 Z \,||\, Y, T, H}$$

PUSH S
$$\frac{H' = H[\eta' \mapsto \{\star x_{\text{parent}} \mapsto \eta\}], \eta' \notin H \qquad S = [\ell : \overset{\star'}{\ell} : d, \dots]}{Z \,||\,[\eta, \text{PUSH } S] \,||\, Y, T, H \longrightarrow^1 P', Z \,||\, Y, [\langle \eta', \ell, S \rangle] \,||\, T, H'}$$

LOOK UP x (BOUND)
$$\frac{\text{LOOKUP}(H, \eta, x) = m}{Z \,||\,[x, \text{LOOKUP}] \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Z \,||\,[m] \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

LOOK UP x (NAMEERROR)
$$\frac{\text{LOOKUP}(H, \eta, x) = *}{Z \,||\,[x, \text{LOOKUP}] \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 [\text{ALLOCNAMEERROR}, \text{RAISE}], [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

MAKE LIST
$$\frac{v = [m_1, \dots, m_n]}{Z \,||\,[m_1, \dots, m_n, \text{LIST } n] \,||\, Y, T, H \longrightarrow^1 Z \,||\,[v] \,||\, Y, T, H}$$

MAKE TUPLE
$$\frac{v = (m_1, \dots, m_n)}{Z \,||\,[m_1, \dots, m_n, \text{TUPLE } n] \,||\, Y, T, H \longrightarrow^1 Z \,||\,[v] \,||\, Y, T, H}$$

Figure 2: Microcommands

Raise (no exception label)

$$\frac{S(\ell) = \ell : * : d}{Z \,||[\textsc{Raise}] \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Z \,||[\textsc{Pop}, \textsc{Raise}] \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

Raise (caught)

$$\frac{S(\ell) = \ell : \ell_0 : d \qquad S(\ell_0) = \ell_0 : \ell_1 : \texttt{catch}\ x \qquad Y' = [x, \textsc{Bind}, \textsc{Advance}]}{Z \,||[\textsc{Raise}] \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Z \,||\, Y' \,||\, Y, [\langle \eta, \ell_0, S \rangle] \,||\, T, H}$$

Goto ℓ

$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} : d}{Z \,||[\textsc{Goto}\ \ell] \,||\, Y, [\langle \eta, \ell', S \rangle] \,||\, T, H \longrightarrow^1 Z \,||\, Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

Gotoifn ℓ (success)

$$\frac{H[m] = \textsc{False} \qquad S(\ell) = \ell : \overset{\star'}{\ell} : d}{Z \,||[m, \textsc{Gotoifn}\ \ell] \,||\, Y, T, H \longrightarrow^1 Z \,||[\textsc{Goto}] \,||\, Y, T, H}$$

Gotoifn ℓ (failure)

$$\frac{H[m] = \textsc{True}}{Z \,||[m, \textsc{Gotoifn}\ \ell] \,||\, Y, T, H \longrightarrow^1 Z \,||[\textsc{Advance}] \,||\, Y, T, H}$$

Call function m

$$\frac{\begin{array}{c}v = \langle \eta,\ \texttt{def}\ (x_1, \ldots, x_n) \to S \rangle \\ Y' = [\eta, \textsc{Push}\ S, m_1, x_1, \textsc{Bind}, \ldots, m_n, x_n, \textsc{Bind}]\end{array}}{Z \,||[v, m_1, \ldots, m_n, \textsc{Call}\ n] \,||\, Y, T, H \longrightarrow^1 Z \,||\, Y' \,||\, Y, T, H}$$

Call function (wrong args)

$$\frac{v = \langle \eta,\ \texttt{def}\ (x_1, \ldots, x_q) \to S \rangle q \neq n}{Z \,||[v, m_1, \ldots, m_n, \textsc{Call}\ n] \,||\, Y, T, H \longrightarrow^1 [\textsc{AllocTypeError}, \textsc{Raise}], T, H}$$

Get Call m

$$\frac{v = H[m_0][\star x_{\texttt{value}}],\ v \text{ is of form } F \text{ or } M \qquad Y' = [v, m_1, \ldots, m_n]}{Z \,||[m_0, \ldots, m_n, \textsc{GetCall}\ n] \,||\, Y, T, H \longrightarrow^1 Z \,||\, Y' \,||\, Y, T, H}$$

Get Call (TypeError)

$$\frac{v = H[m_0][\star x_{\texttt{value}}],\ v \text{ is not of form } F \text{ or } M}{Z \,||[m_0, \ldots, m_n, \textsc{GetCall}\ n] \,||\, Y, T, H \longrightarrow^1 [\textsc{AllocTypeError}, \textsc{Raise}], T, H}$$

Figure 3: Microcommands (cont.)

CONVERT FUNCTION v

$$\frac{v = F}{Z \,||\,[v, m_1, \ldots, m_n, \text{CONVERT } n] \,||\, Y, T, H \longrightarrow^1 Z \,||\,[v, m_1, \ldots, m_n, \text{CALL } n] \,||\, Y, T, H}$$

CONVERT METHOD v

$$\frac{v = \langle m_0, F\rangle \qquad v' = F}{Z \,||\,[v, m_1, \ldots, m_n, \text{CONVERT } n] \,||\, Y, T, H \longrightarrow^1 Z \,||\,[v', m_0, m_1, \ldots, m_n, \text{CALL } n+1] \,||\, Y, T, H}$$

RETRIEVE x

$$\frac{v = H[m][x] \qquad Y' = [v, \text{STORE}]}{Z \,||\,[m, x, \text{RETRIEVE}] \,||\, Y, T, H \longrightarrow^1 Z \,||\, Y' \,||\, Y, T, H}$$

RETRIEVE x (ATTRIBUTEERROR)

$$\frac{* = H[m][x]}{Z \,||\,[m, x, \text{RETRIEVE}] \,||\, Y, T, H \longrightarrow^1 [\text{ALLOCATTRERROR}, \text{RAISE}], T, H}$$

Figure 4: Microcommands (cont.)

Figure 5: Magic Functions

NONE ASSIGNMENT

$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} : x = \texttt{None} \qquad Y = [m_{\texttt{None}}, x, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

LITERAL ASSIGNMENT

$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} : x = e, \; e \text{ is of form } \mathbb{Z}, \mathbb{S} \qquad Y = [v, \textsc{Store}, \textsc{Wrap}, \textsc{Store}, x, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

(TODO: make literal category (ints, str, bool, None) – TC)

NAME ASSIGNMENT

$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} : x_1 = x_2 \qquad Y = [x_2, \textsc{Lookup}, x_1, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

LIST ASSIGNMENT

$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} : x = [x_1, \ldots, x_n] \qquad Y = [(x_1, \textsc{LookUp}), \ldots, (x_n, \textsc{LookUp}), \textsc{List } n, \textsc{Store}, \textsc{Wrap}, \textsc{Store}, x, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

(Parentheses in Y group instructions together for convenience of reading. – TC)

TUPLE ASSIGNMENT

$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} : x = [x_1, \ldots, x_n] \qquad Y = [(x_1, \textsc{LookUp}), \ldots, (x_n, \textsc{LookUp}), \textsc{Tuple } n, \textsc{Store}, \textsc{Wrap}, \textsc{Store}, x, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

FUNCTIONDEF ASSIGNMENT

$$\frac{S(\ell) = \ell : \ell' : x = \texttt{def } (x_1, \ldots, x_n) = \{S'\} \qquad v = \langle \eta, \texttt{def } (x_1, \ldots, x_n) \rightarrow S' \rangle \qquad Y = [v, \textsc{Store}, \textsc{Wrap}, \textsc{Store}, x, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

ATTRIBUTE ASSIGNMENT

$$\frac{S(\ell) = \ell : \ell' : x = x_1.x_2 \qquad Y = [x_1, \textsc{LookUp}, x_2, \textsc{Retrieve}, \textsc{Wrap}, \textsc{Store}, x, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

CALL ASSIGNMENT

$$\frac{S(\ell) = \ell : \ell' : x = x_0(x_1, \ldots, x_n) \qquad Y = [x_0, \textsc{LookUp}, \ldots, x_n, \textsc{LookUp}, \textsc{GetCall } n, \textsc{Convert } n]}{[\,], [\langle \eta, \ell, S \rangle] \,||\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,||\, T, H}$$

Figure 6: Operational Semantics: Assignment

PASS
$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} \, : \, \texttt{pass} \qquad Y = [\textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,\|\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,\|\, T, H}$$

RETURN
$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} \, : \, \texttt{return } x \qquad T = [\langle \eta', \ell'', S' \rangle] \,\|\, T'}{S(\ell'') = \ell'' : \overset{\star'''}{\ell} \, : x' = e \qquad Y = [x, \textsc{Lookup}, \textsc{Pop}, x', \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, \ell, S \rangle] \,\|\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,\|\, T, H}$$

GOTO
$$\frac{S(\ell) = \ell : \overset{\star'}{\ell} \, : \, \texttt{goto } \ell'' \qquad Y = [\textsc{Goto } \ell'']}{[\,], [\langle \eta, \ell, S \rangle] \,\|\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,\|\, T, H}$$

GOTOIFNOT
$$\frac{S(\ell) = \ell : \ell' : \, \texttt{goto } \ell'' \texttt{ if not } x \qquad Y = [x, \textsc{Lookup}, \textsc{Gotoifn } \ell'']}{[\,], [\langle \eta, \ell, S \rangle] \,\|\, T, H \longrightarrow^1 Y, [\langle \eta, \ell, S \rangle] \,\|\, T, H}$$

END OF FUNCTION
$$\frac{t = \langle \eta', \ell, S' \rangle \qquad S(\ell) = \ell : \overset{\star'}{\ell} \, : x = e \qquad Y = [\textsc{Pop}, m_{\texttt{None}}, x, \textsc{Bind}, \textsc{Advance}]}{[\,], [\langle \eta, *, S \rangle, t] \,\|\, T, H \longrightarrow^1 Y, [\langle \eta, *, S \rangle, t] \,\|\, T, H}$$

*(m_{None} is a memory location reserved for **None**. – TC)*

END OF PROGRAM
$$\frac{T = [\langle \eta, *, S \rangle] \qquad Y = [\textsc{Pop}]}{[\,], T, H \longrightarrow^1 Y, T, H}$$

Figure 7: Operational Semantics: Flow

Definition 0.2.
$$\textsc{Lookup}(H, \eta, x) = \textit{(todo – TC)}$$

Definition 0.3.
$$H[m] = v, B_{obj} = \{\star x_{value} \mapsto v, \_\_getattribute\_\_ \mapsto \textsc{Getattribute}\}$$

$$\textsc{GetObj}(H, m) = \begin{cases} B, & \textit{if } v = B \\ B = B_{obj}[\_\_class\_\_ \mapsto \star\textsc{funType}], & \textit{if } v = F \end{cases} \qquad (1)$$

(getobj takes (H,m,memory location of getattribute) – TC)

Figure 8: Helper Functions