– First Step: Define the semantic domains
– Second Step: Define Big Step semantics

1 Semantic Domain - *Python*

Expressions' transitions rules:

$$s, b \vdash exp \Downarrow s', b', v$$

Commands' transition rule:

$$s, b \vdash C \Downarrow s', b'$$

Bindings:

$$(identifier \mapsto cell) \cup (identifier \mapsto closure).$$

Store:

$$cell \mapsto value$$

$$value = string\_literal \cup num\_literal \cup bool\_literal \cup list$$

Regras de atribuição

Atribuição de Elemento não lista e que não exista nos bindings:

$$\textit{(list-attrib-1)}\frac{s, b \vdash \mathit{exp} \Downarrow s', b', v}{s, b \vdash \mathit{identifier} = \mathit{exp} \Downarrow s', b'[\mathit{identifier} \mapsto v]} identifier \notin dom(b), v \notin list$$

Atribuição de Elemento não lista e que exista nos bindings:

$$\textit{(list-attrib-2)}\frac{s, b \vdash \mathit{exp} \Downarrow s', b', v}{s, b \vdash \mathit{identifier} = \mathit{exp} \Downarrow s', b'[b(\mathit{identifier}) \mapsto v]} identifier \in dom(b), v \notin list$$

Atribuição de Elemento lista e que não exista nos bindings:

$$\textit{(list-attrib-3)}\frac{s, b \vdash \mathit{exp} \Downarrow s', b', v \qquad k := alloc(s', v)}{s, b \vdash \mathit{identifier} = \mathit{exp} \Downarrow s'[k \mapsto v], b'[\mathit{identifier} \mapsto v]} identifier \notin dom(b), v \in list$$

Atribuição de Elemento lisãa e que exista nos bindings:

$$\textit{(list-attrib-4)}\frac{s, b \vdash \mathit{exp} \Downarrow s', b', v}{s, b \vdash \mathit{identifier} = \mathit{exp} \Downarrow s'[b'(\mathit{identifier}) \mapsto v], b'} identifier \in dom(b), v \in list$$

Atribuição em listas:

$$\textit{(list-attrib)}\frac{\begin{array}{cc} s, b \vdash \mathit{exp1} \Downarrow s', b', vK & s', b' \vdash \mathit{exp2} \Downarrow s'', b'', vI \\ s'', b'' \vdash \mathit{exp3} \Downarrow s''', b''', vV & update(s'''(vK), vI, vV) = l \end{array}}{s, b \vdash \mathit{exp1}[\mathit{exp2}] = \mathit{exp3} \Downarrow s'''[k \mapsto l], b'''}$$

Regras de desvio condicional:

$$\textit{(if-true)}\frac{s, b \vdash \mathit{exp} \Downarrow s', b', v' \qquad s', b' \vdash \mathit{C1} \Downarrow s'', b''}{s, b \vdash \mathit{if}\, \mathit{exp} : \mathit{C1}\, \mathit{else} : \mathit{C2} \Downarrow s'', b''} v' = True$$

$$\textit{(if-false)}\frac{s, b \vdash \mathit{exp} \Downarrow s', b', v' \qquad s', b' \vdash \mathit{C2} \Downarrow s'', b''}{s, b \vdash \mathit{if}\, \mathit{exp} : \mathit{C1}\, \mathit{else} : \mathit{C2} \Downarrow s'', b''} v' = False$$

For loop
Transition rules for for clause:

$$s, b, [], i \vdash C \Downarrow s, b$$

$$s, b[i \mapsto v] \vdash C \Downarrow s', b'$$

$$s', b', l, i \vdash C \Downarrow s'', b''$$

$$\textit{(for-loop)}\frac{s, b \vdash \mathit{exp} \Downarrow s', b', v \qquad s, b[i \mapsto v] \vdash C \Downarrow s', b' \qquad s', b', l, i \vdash C \Downarrow s'', b''}{s', b' \vdash \mathit{for}\, \mathit{identifier}\, \mathit{in}\, \mathit{exp} : C \Downarrow s'', b''} v \in list$$

$$\textit{(string-add)}\ \frac{s, b \vdash \textit{exp1} \Downarrow s', b', v' \qquad s', b' \vdash \textit{exp2} \Downarrow s'', b'', v''}{s'', b'' \vdash \textit{exp1} + \textit{exp2} \Downarrow s''', b''', v'''}\ v''' = v'.v''$$

$$\textit{(access-list)}\ \frac{s, b \vdash \textit{exp1} \Downarrow s', b', vL \qquad s', b' \vdash \textit{exp2} \Downarrow s'', b'', vI}{s, b \vdash \textit{exp1}[\textit{exp2}] \Downarrow s''', b''', v}\ vL \in list$$

$$\frac{indegree(w) = 0 \qquad \overline{a} = Eval_\rho(\overline{E}) \qquad A'_c = A_c - \{w \mapsto v \mid v \in V\} \qquad A'_d = A_d - \{w \mapsto v \mid v \in V\} \qquad G = \langle V[w : S!\overline{E},\ w' : S?\overline{X}], A_c[w \mapsto w'], A_d\rangle \qquad G' = \langle V[w' : S?\overline{X}], A'_c, A'_d\rangle}{\langle G, \rho, I, O, M, T_E, N_{MT}\rangle \ \rhd\ \langle G', \rho, I, O[\langle S, w', \overline{a}\rangle], M[\langle w : S!\overline{E},\ w', now(), N_{MT}\rangle], T_E, N_{MT}\rangle}$$