

A Linguagem Imperativa WHILE

$$\begin{array}{ll}
 \text{Sintaxe abstracta} &
 \begin{array}{l}
 \mathbf{Aexp} \ni a ::= n \mid x \mid a_1 + a_2 \mid a_1 * a_2 \mid a_1 - a_2 \\
 \mathbf{Bexp} \ni b ::= \mathbf{true} \mid \mathbf{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b \mid b_1 \wedge b_2 \mid b_1 \vee b_2 \\
 \mathbf{Stm} \ni C ::= x := a \mid \mathbf{skip} \mid C_1 ; C_2 \mid \mathbf{if } b \mathbf{ then } C_1 \mathbf{ else } C_2 \mid \mathbf{while } b \mathbf{ do } C
 \end{array}
 \end{array}$$

Semântica Natural (*big-step*)

$$\begin{array}{c}
 [\text{skip}_{\text{ns}}] \quad \overline{\langle \mathbf{skip}, s \rangle \rightarrow s} \quad [\text{if}_{\text{ns}}^{\text{tt}}] \quad \frac{\langle C_1, s \rangle \rightarrow s'}{\langle \mathbf{if } b \mathbf{ then } C_1 \mathbf{ else } C_2, s \rangle \rightarrow s'} \text{ se } \mathcal{B}[b]s = \mathbf{tt} \\
 [\text{ass}_{\text{ns}}] \quad \overline{\langle x := a, s \rangle \rightarrow s [x \mapsto \mathcal{A}[a]s]} \quad [\text{if}_{\text{ns}}^{\text{ff}}] \quad \frac{\langle C_2, s \rangle \rightarrow s'}{\langle \mathbf{if } b \mathbf{ then } C_1 \mathbf{ else } C_2, s \rangle \rightarrow s'} \text{ se } \mathcal{B}[b]s = \mathbf{ff} \\
 [\text{comp}_{\text{ns}}] \quad \frac{\langle C_1, s \rangle \rightarrow s' \quad \langle C_2, s' \rangle \rightarrow s''}{\langle C_1 ; C_2, s \rangle \rightarrow s''} \quad [\text{while}_{\text{ns}}^{\text{tt}}] \quad \frac{\langle C, s \rangle \rightarrow s' \quad \langle \mathbf{while } b \mathbf{ do } C, s' \rangle \rightarrow s''}{\langle \mathbf{while } b \mathbf{ do } C, s \rangle \rightarrow s''} \text{ se } \mathcal{B}[b]s = \mathbf{tt} \\
 [\text{while}_{\text{ns}}^{\text{ff}}] \quad \overline{\langle \mathbf{while } b \mathbf{ do } C, s \rangle \rightarrow s} \text{ se } \mathcal{B}[b]s = \mathbf{ff} \\
 \begin{array}{l}
 \mathcal{S}_{\text{ns}} : \mathbf{Stm} \rightarrow (\mathbf{State} \hookrightarrow \mathbf{State}) \\
 \mathcal{S}_{\text{ns}}[C]s = \begin{cases} s' & , \text{ se } \langle C, s \rangle \rightarrow s' \\ \underline{\text{undef}} & , \text{ nos outros casos} \end{cases}
 \end{array}
 \end{array}$$

Semântica Operacional Estrutural (*small-step*)

$$\begin{array}{c}
 [\text{skip}_{\text{sos}}] \quad \overline{\langle \mathbf{skip}, s \rangle \Rightarrow s} \quad [\text{if}_{\text{sos}}^{\text{tt}}] \quad \overline{\langle \mathbf{if } b \mathbf{ then } C_1 \mathbf{ else } C_2, s \rangle \Rightarrow \langle C_1, s \rangle} \text{ se } \mathcal{B}[b]s = \mathbf{tt} \\
 [\text{ass}_{\text{sos}}] \quad \overline{\langle x := a, s \rangle \Rightarrow s [x \mapsto \mathcal{A}[a]s]} \quad [\text{if}_{\text{sos}}^{\text{ff}}] \quad \overline{\langle \mathbf{if } b \mathbf{ then } C_1 \mathbf{ else } C_2, s \rangle \Rightarrow \langle C_2, s \rangle} \text{ se } \mathcal{B}[b]s = \mathbf{ff} \\
 [\text{comp}_{\text{sos}}^1] \quad \frac{\langle C_1, s \rangle \Rightarrow \langle C'_1, s' \rangle}{\langle C_1 ; C_2, s \rangle \Rightarrow \langle C'_1 ; C_2, s' \rangle} \quad [\text{comp}_{\text{sos}}^2] \quad \frac{\langle C_1, s \rangle \Rightarrow s'}{\langle C_1 ; C_2, s \rangle \Rightarrow \langle C_2, s' \rangle} \\
 [\text{while}_{\text{sos}}] \quad \overline{\langle \mathbf{while } b \mathbf{ do } C, s \rangle \Rightarrow \langle \mathbf{if } b \mathbf{ then } (C ; \mathbf{while } b \mathbf{ do } C) \mathbf{ else } \mathbf{skip}, s \rangle}
 \end{array}$$

$$\begin{array}{l}
 \mathcal{S}_{\text{sos}} : \mathbf{Stm} \rightarrow (\mathbf{State} \hookrightarrow \mathbf{State}) \\
 \mathcal{S}_{\text{sos}}[C]s = \begin{cases} s' & , \text{ se } \langle C, s \rangle \Rightarrow^* s' \\ \underline{\text{undef}} & , \text{ nos outros casos} \end{cases}
 \end{array}$$

A maquina abstracta **AM**

As **instruções** de **AM** são dadas pela sintaxe abstracta:

$$\begin{aligned} \text{inst} &::= \text{PUSH-}n \mid \text{ADD} \mid \text{MULT} \mid \text{SUB} \\ &\quad \mid \text{TRUE} \mid \text{FALSE} \mid \text{EQ} \mid \text{LE} \mid \text{AND} \mid \text{NEG} \\ &\quad \mid \text{FETCH-}x \mid \text{STORE-}x \\ &\quad \mid \text{NOOP} \mid \text{BRANCH}(c, c) \mid \text{LOOP}(c, c) \\ c &::= \varepsilon \mid \text{inst}:c \end{aligned}$$

Code é a categoria sintática das sequências de instruções.

$$\mathbf{Stack} = (\mathbf{Z} \cup \mathbf{T})^*$$

$$\mathbf{State} = \mathbf{Var} \rightarrow \mathbf{Z}$$

$$\langle c, e, s \rangle \in \mathbf{Code} \times \mathbf{Stack} \times \mathbf{State}$$

Funções de tradução:

$$\mathcal{CA}: \mathbf{Aexp} \rightarrow \mathbf{Code}$$

$$\mathcal{CB}: \mathbf{Bexp} \rightarrow \mathbf{Code}$$

$$\mathcal{CS}: \mathbf{Stm} \rightarrow \mathbf{Code}$$

Semântica operacional de **AM**

$$\begin{aligned} \langle \text{PUSH-}n:c, e, s \rangle &\triangleright \langle c, \mathcal{N}[n]:e, s \rangle \\ \langle \text{ADD}:c, z_1:z_2:e, s \rangle &\triangleright \langle c, (z_1+z_2):e, s \rangle \quad \text{if } z_1, z_2 \in \mathbf{Z} \\ \langle \text{MULT}:c, z_1:z_2:e, s \rangle &\triangleright \langle c, (z_1 \star z_2):e, s \rangle \quad \text{if } z_1, z_2 \in \mathbf{Z} \\ \langle \text{SUB}:c, z_1:z_2:e, s \rangle &\triangleright \langle c, (z_1 - z_2):e, s \rangle \quad \text{if } z_1, z_2 \in \mathbf{Z} \\ \langle \text{TRUE}:c, e, s \rangle &\triangleright \langle c, \text{tt}:e, s \rangle \\ \langle \text{FALSE}:c, e, s \rangle &\triangleright \langle c, \text{ff}:e, s \rangle \\ \langle \text{EQ}:c, z_1:z_2:e, s \rangle &\triangleright \langle c, (z_1 = z_2):e, s \rangle \quad \text{if } z_1, z_2 \in \mathbf{Z} \\ \langle \text{LE}:c, z_1:z_2:e, s \rangle &\triangleright \langle c, (z_1 \leq z_2):e, s \rangle \quad \text{if } z_1, z_2 \in \mathbf{Z} \\ \langle \text{AND}:c, t_1:t_2:e, s \rangle &\triangleright \\ &\quad \left\{ \begin{array}{ll} \langle c, \text{tt} : e, s \rangle & \text{if } t_1 = \text{tt} \text{ and } t_2 = \text{tt} \\ \langle c, \text{ff} : e, s \rangle & \text{if } t_1 = \text{ff} \text{ or } t_2 = \text{ff}, t_1, t_2 \in \mathbf{T} \end{array} \right. \\ \langle \text{NEG}:c, t:e, s \rangle &\triangleright \left\{ \begin{array}{ll} \langle c, \text{ff} : e, s \rangle & \text{if } t = \text{tt} \\ \langle c, \text{tt} : e, s \rangle & \text{if } t = \text{ff} \end{array} \right. \\ \langle \text{FETCH-}x:c, e, s \rangle &\triangleright \langle c, (s\ x):e, s \rangle \\ \langle \text{STORE-}x:c, z:e, s \rangle &\triangleright \langle c, e, s[x \mapsto z] \rangle \quad \text{if } z \in \mathbf{Z} \\ \langle \text{NOOP}:c, e, s \rangle &\triangleright \langle c, e, s \rangle \\ \langle \text{BRANCH}(c_1, c_2):c, t:e, s \rangle &\triangleright \left\{ \begin{array}{ll} \langle c_1 : c, e, s \rangle & \text{if } t = \text{tt} \\ \langle c_2 : c, e, s \rangle & \text{if } t = \text{ff} \end{array} \right. \\ \langle \text{LOOP}(c_1, c_2):c, e, s \rangle &\triangleright \langle c_1:\text{BRANCH}(c_2:\text{LOOP}(c_1, c_2), \text{NOOP}):c, e, s \rangle \end{aligned}$$

Semântica Axiomática

[skip_P]

$$\overline{\{\phi\} \text{skip} \{\phi\}}$$

[ass_P]

$$\overline{\{\psi[e/x]\} x := e \{\psi\}}$$

[comp_P]

$$\frac{\{\phi\} C_1 \{\theta\} \quad \{\theta\} C_2 \{\psi\}}{\{\phi\} C_1 ; C_2 \{\psi\}}$$

[if_P]

$$\frac{\{\phi \wedge b\} C_t \{\psi\} \quad \{\phi \wedge \neg b\} C_f \{\psi\}}{\{\phi\} \text{if } b \text{ then } C_t \text{ else } C_f \{\psi\}}$$

[while_P]

$$\frac{\{\theta \wedge b\} C \{\theta\}}{\{\theta\} \text{while } b \text{ do } C \{\theta \wedge \neg b\}}$$

[cons_P]

$$\frac{\{\phi\} C \{\psi\}}{\{\phi'\} C \{\psi'\}} \text{ se } \phi' \Rightarrow \phi \text{ e } \psi \Rightarrow \psi'$$

Semântica Denotacional

$$\mathcal{S}_{ds} : \mathbf{Stm} \rightarrow (\mathbf{State} \hookrightarrow \mathbf{State})$$

$$\mathcal{S}_{ds}[x := a] s = s[x \mapsto \mathcal{A}[a] s]$$

$$\mathcal{S}_{ds}[\text{skip}] = \text{id}$$

$$\mathcal{S}_{ds}[C_1 ; C_2] = \mathcal{S}_{ds}[C_2] \circ \mathcal{S}_{ds}[C_1]$$

$$\mathcal{S}_{ds}[\text{if } b \text{ then } C_1 \text{ else } C_2] = \text{cond}(\mathcal{B}[b], \mathcal{S}_{ds}[C_1], \mathcal{S}_{ds}[C_2])$$

$$\mathcal{S}_{ds}[\text{while } b \text{ do } C] = \text{FIX } F$$

$$\text{onde } F g = \text{cond}(\mathcal{B}[b], g \circ \mathcal{S}_{ds}[C], \text{id})$$

$$\text{cond}(p, g_1, g_2) s = \begin{cases} g_1 s & \text{se } p s = \text{tt} \\ g_2 s & \text{se } p s = \text{ff} \end{cases}$$

Uma Linguagem Funcional

I. Sintaxe Abstracta:

$\langle exp \rangle ::= \langle var \rangle \mid \lambda \langle var \rangle. \langle exp \rangle \mid \langle exp \rangle \langle exp \rangle$	(cálculo básico)
0 1 2 ... True False	(naturais, booleanos)
$\neg \langle exp \rangle \mid -\langle exp \rangle \mid \langle exp \rangle \text{ bop } \langle exp \rangle$	(operadores)
if $\langle exp \rangle$ then $\langle exp \rangle$ else $\langle exp \rangle$	(condicional)
$\langle \langle exp \rangle, \dots, \langle exp \rangle \rangle \mid \langle exp \rangle. \langle tag \rangle$	(tuplos)
$@\langle tag \rangle \langle exp \rangle \mid \text{sumcase } \langle exp \rangle \text{ of } (\langle exp \rangle, \dots, \langle exp \rangle)$	(alternativas)
$\lambda \langle pat \rangle. \langle exp \rangle \mid \text{let } \langle pat \rangle \equiv \langle exp \rangle, \dots, \langle pat \rangle \equiv \langle exp \rangle \text{ in } \langle exp \rangle$	(definições)
nil $\langle exp \rangle :: \langle exp \rangle \mid \text{listcase } \langle exp \rangle \text{ of } (\langle exp \rangle, \langle exp \rangle)$	(listas)
letrec $\langle var \rangle \equiv \lambda \langle pat \rangle. \langle exp \rangle, \dots, \langle var \rangle \equiv \lambda \langle pat \rangle. \langle exp \rangle \text{ in } \langle exp \rangle$	(def. recursivas, na ling. CBV)
rec $\langle exp \rangle$	(op. ponto fixo, na ling. CBN)
letrec $\langle pat \rangle \equiv \langle exp \rangle, \dots, \langle pat \rangle \equiv \langle exp \rangle \text{ in } \langle exp \rangle$	(def. recursivas, na ling. CBN)

$$\text{bop} \in \{+, -, *, \text{div}, \text{mod}, =, \neq, <, >, \leq, \geq, \vee, \wedge\}$$

$$\langle tag \rangle ::= 1 \mid 2 \mid \dots$$

$$\langle pat \rangle ::= \langle var \rangle \mid \langle \langle pat \rangle, \dots, \langle pat \rangle \rangle$$

II. Tipos:

Sintaxe abstracta

$$\begin{aligned}\langle type \rangle &::= \mathbf{Int} \mid \mathbf{Bool} \mid \langle type \rangle \rightarrow \langle type \rangle \\ &\quad \mid \mathbf{Prod}(\langle type \rangle, \dots, \langle type \rangle) \mid \mathbf{Sum}(\langle type \rangle, \dots, \langle type \rangle) \mid \mathbf{List} \langle type \rangle\end{aligned}$$

$$\begin{aligned}\langle type \rangle \times \dots \times \langle type \rangle &\doteq \mathbf{Prod}(\langle type \rangle, \dots, \langle type \rangle) \\ \mathbf{Unit} &\doteq \mathbf{Prod}() \\ \langle type \rangle + \dots + \langle type \rangle &\doteq \mathbf{Sum}(\langle type \rangle, \dots, \langle type \rangle)\end{aligned}$$

Regras

$$\frac{}{\Gamma, x : \tau \vdash x : \tau} \quad \frac{\Gamma \vdash e_1 : \sigma \rightarrow \tau \quad \Gamma \vdash e_2 : \sigma}{\Gamma \vdash e_1 e_2 : \tau} \quad \frac{\Gamma, p : \sigma \vdash e : \tau}{\Gamma \vdash \lambda p. e : \sigma \rightarrow \tau}$$

$$\frac{}{\Gamma \vdash \mathbf{True} : \mathbf{Bool}} \quad \frac{}{\Gamma \vdash \mathbf{False} : \mathbf{Bool}} \quad \frac{}{\Gamma \vdash n : \mathbf{Int}} \text{ para cada } n \in \mathbb{N}$$

$$\frac{\Gamma \vdash e : \mathbf{Bool}}{\Gamma \vdash \neg e : \mathbf{Bool}} \quad \frac{\Gamma \vdash e_1 : \mathbf{Bool} \quad \Gamma \vdash e_2 : \mathbf{Bool}}{\Gamma \vdash e_1 \mathbf{bop} e_2 : \mathbf{Bool}} \text{ para } \mathbf{bop} \in \{\vee, \wedge\}$$

$$\frac{\Gamma \vdash e : \mathbf{Int}}{\Gamma \vdash \neg e : \mathbf{Int}} \quad \frac{\Gamma \vdash e_1 : \mathbf{Int} \quad \Gamma \vdash e_2 : \mathbf{Int}}{\Gamma \vdash e_1 \mathbf{bop} e_2 : \mathbf{Int}} \text{ para } \mathbf{bop} \in \{+, -, *, \text{div}, \text{mod}\}$$

$$\frac{\Gamma \vdash e_1 : \mathbf{Int} \quad \Gamma \vdash e_2 : \mathbf{Int}}{\Gamma \vdash e_1 \mathbf{bop} e_2 : \mathbf{Bool}} \text{ para } \mathbf{bop} \in \{=, \neq, <, >, \leq, \geq\} \quad \frac{\Gamma \vdash e_1 : \mathbf{Bool} \quad \Gamma \vdash e_2 : \theta \quad \Gamma \vdash e_3 : \theta}{\Gamma \vdash \text{if } e_1 \text{ then } e_2 \text{ else } e_3 : \theta}$$

$$\frac{\Gamma \vdash e_1 : \theta_1 \dots \Gamma \vdash e_n : \theta_n}{\Gamma \vdash \langle e_1, \dots, e_n \rangle : \theta_1 \times \dots \times \theta_n} \quad \frac{\Gamma \vdash e : \theta_1 \times \dots \times \theta_n}{\Gamma \vdash e.k : \theta_k} \text{ para } k \in \{1, \dots, n\}$$

$$\frac{\Gamma \vdash e : \theta_k}{\Gamma \vdash @k e : \theta_1 + \dots + \theta_n} \text{ para } k \in \{1, \dots, n\} \quad \frac{\Gamma \vdash e : \theta_1 + \dots + \theta_n \quad \Gamma \vdash e_1 : \theta_1 \rightarrow \theta \dots \Gamma \vdash e_n : \theta_n \rightarrow \theta}{\Gamma \vdash \text{sumcase } e \text{ of } (e_1, \dots, e_n) : \theta}$$

$$\frac{}{\Gamma \vdash \text{nil} : \mathbf{List} \theta} \quad \frac{\Gamma \vdash e_1 : \theta \quad \Gamma \vdash e_2 : \mathbf{List} \theta}{\Gamma \vdash (e_1 :: e_2) : \mathbf{List} \theta} \quad \frac{\Gamma \vdash e : \mathbf{List} \theta \quad \Gamma \vdash e_1 : \theta' \quad \Gamma \vdash e_2 : \theta \rightarrow \mathbf{List} \theta \rightarrow \theta'}{\Gamma \vdash \text{listcase } e \text{ of } (e_1, e_2) : \theta'}$$

$$\frac{\Gamma, p_1 : \theta_1, \dots, p_n : \theta_n \vdash e : \theta}{\Gamma, \langle p_1, \dots, p_n \rangle : \theta_1 \times \dots \times \theta_n \vdash e : \theta} \quad \frac{\Gamma \vdash e_1 : \theta_1 \dots \Gamma \vdash e_n : \theta_n \quad \Gamma, p_1 : \theta_1, \dots, p_n : \theta_n \vdash e : \theta}{\Gamma \vdash \text{let } p_1 \equiv e_1, \dots, p_n \equiv e_n \text{ in } e : \theta}$$

$$\frac{\Gamma \vdash e : \theta \rightarrow \theta}{\Gamma \vdash \text{rec } e : \theta} \quad \frac{\Gamma' \vdash e_1 : \theta_1 \dots \Gamma' \vdash e_n : \theta_n \quad \Gamma' \vdash e : \theta}{\Gamma \vdash \text{letrec } p_1 \equiv e_1, \dots, p_n \equiv e_n \text{ in } e : \theta} \text{ com } \Gamma' = \Gamma, p_1 : \theta_1, \dots, p_n : \theta_n$$

III. Semântica de avaliação “call-by-value”:

Formas canónicas

$$\langle cfm \rangle ::= \lambda \langle var \rangle. \langle exp \rangle | \dots | -2 | -1 | 0 | 1 | 2 | \dots | \text{True} | \text{False} \\ | \langle \langle cfm \rangle, \dots, \langle cfm \rangle \rangle | @\langle tag \rangle \langle cfm \rangle | \text{nil} | \langle cfm \rangle :: \langle cfm \rangle$$

Regras de avaliação

$$\frac{}{z \Rightarrow z} \quad \frac{e \Rightarrow \lambda v. \hat{e} \quad e' \Rightarrow z' \quad \hat{e}[z'/v] \Rightarrow z}{e e' \Rightarrow z}$$

$$\frac{e_1 \Rightarrow [i_1] \quad e_2 \Rightarrow [i_2]}{e_1 \text{ bop } e_2 \Rightarrow [i_1 \text{ bop } i_2]} \text{ com } \text{bop} \in \{+, -, *, =, \neq, <, >, \leq, \geq\} \text{ ou } \text{bop} \in \{\text{div}, \text{mod}\} \text{ e } i_2 \neq 0$$

$$\frac{e \Rightarrow [i]}{-e \Rightarrow [-i]} \quad \frac{e \Rightarrow [b]}{\neg e \Rightarrow [\neg b]} \quad \frac{e_1 \Rightarrow [b_1] \quad e_2 \Rightarrow [b_2]}{e_1 \text{ bop } e_2 \Rightarrow [b_1 \text{ bop } b_2]} \text{ com } \text{bop} \in \{\vee, \wedge\}$$

$$\frac{e_1 \Rightarrow \text{True} \quad e_2 \Rightarrow z}{\text{if } e_1 \text{ then } e_2 \text{ else } e_3 \Rightarrow z} \quad \frac{e_1 \Rightarrow \text{False} \quad e_3 \Rightarrow z}{\text{if } e_1 \text{ then } e_2 \text{ else } e_3 \Rightarrow z}$$

$$\frac{e_1 \Rightarrow z_1 \dots e_n \Rightarrow z_n}{\langle e_1, \dots, e_n \rangle \Rightarrow \langle z_1, \dots, z_n \rangle} \quad \frac{e \Rightarrow \langle z_1, \dots, z_n \rangle \quad k \in \{1, \dots, n\}}{e.k \Rightarrow z_k}$$

$$\frac{e \Rightarrow z}{@k e \Rightarrow @k z} \quad \frac{e \Rightarrow @k z \quad e_k z \Rightarrow z'}{\text{sumcase } e \text{ of } (e_1, \dots, e_n) \Rightarrow z'}$$

$$\frac{e \Rightarrow z \quad e' \Rightarrow z'}{e :: e' \Rightarrow z :: z'} \quad \frac{e \Rightarrow \text{nil} \quad e_1 \Rightarrow z}{\text{listcase } e \text{ of } (e_1, e_2) \Rightarrow z} \quad \frac{e \Rightarrow z :: z' \quad e_2 z z' \Rightarrow z''}{\text{listcase } e \text{ of } (e_1, e_2) \Rightarrow z''}$$

$$\frac{(\lambda v_1 \dots v_n. e) (\lambda u_1. e_1^*) \dots (\lambda u_n. e_n^*) \Rightarrow z}{\text{letrec } v_1 \equiv \lambda u_1. e_1, \dots, v_n \equiv \lambda u_n. e_n \text{ in } e \Rightarrow z}$$

com $e_i^* = \text{letrec } v_1 \equiv \lambda u_1. e_1, \dots, v_n \equiv \lambda u_n. e_n \text{ in } e_i$

Açúcar sintático: definições, padrões, e listas (definição alternativa)

$$\text{let } p_1 \equiv e_1, \dots, p_n \equiv e_n \text{ in } e \doteq (\lambda p_1 \dots p_n. e) e_1 \dots e_n$$

$$\lambda \langle p_1, \dots, p_n \rangle. e \doteq \lambda v. \text{let } p_1 \equiv v.1, \dots, p_n \equiv v.n \text{ in } e$$

$$\begin{array}{lcl} \text{nil} & \doteq & @1 \langle \rangle \\ e :: e' & \doteq & @2 \langle e, e' \rangle \\ \text{listcase } e \text{ of } (e_1, e_2) & \doteq & \text{sumcase } e \text{ of } (\lambda \langle \rangle. e_1, \lambda \langle x, y \rangle. e_2 x y) \end{array}$$

IV. Semântica de avaliação “call-by-name”:

Formas canónicas

$$\langle cfm \rangle ::= \lambda \langle var \rangle. \langle exp \rangle | \dots | -2 | -1 | 0 | 1 | 2 | \dots | \text{True} | \text{False} \\ | \langle \langle exp \rangle, \dots, \langle exp \rangle \rangle | @\langle tag \rangle \langle exp \rangle | \text{nil} | \langle exp \rangle :: \langle exp \rangle$$

Regras de avaliação

$$\frac{}{z \Rightarrow z} \quad \frac{e \Rightarrow \lambda v. \widehat{e} \quad \widehat{e}[e'/v] \Rightarrow z}{e e' \Rightarrow z}$$

$$\frac{e_1 \Rightarrow [i_1] \quad e_2 \Rightarrow [i_2]}{e_1 \text{ bop } e_2 \Rightarrow [i_1 \text{ bop } i_2]} \text{ para } \text{bop} \in \{+, -, *, =, \neq, <, >, \leq, \geq\} \text{ ou } \text{bop} \in \{\text{div}, \text{mod}\} \text{ e } i_2 \neq 0$$

$$\frac{e \Rightarrow [i]}{-e \Rightarrow [-i]} \quad \frac{e_1 \Rightarrow \text{True} \quad e_2 \Rightarrow z}{\text{if } e_1 \text{ then } e_2 \text{ else } e_3 \Rightarrow z} \quad \frac{e_1 \Rightarrow \text{False} \quad e_3 \Rightarrow z}{\text{if } e_1 \text{ then } e_2 \text{ else } e_3 \Rightarrow z}$$

$$\frac{e \Rightarrow \langle e_1, \dots, e_n \rangle \quad e_k \Rightarrow z}{e.k \Rightarrow z} \text{ para } k \in \{1, \dots, n\}$$

$$\frac{e \Rightarrow @_k e' \quad e_k e' \Rightarrow z}{\text{sumcase } e \text{ of } (e_1, \dots, e_n) \Rightarrow z}$$

$$\frac{e \Rightarrow \text{nil} \quad e_1 \Rightarrow z}{\text{listcase } e \text{ of } (e_1, e_2) \Rightarrow z} \quad \frac{e \Rightarrow e' :: e'' \quad e_2 e' e'' \Rightarrow z}{\text{listcase } e \text{ of } (e_1, e_2) \Rightarrow z}$$

$$\frac{e(\text{rec } e) \Rightarrow z}{\text{rec } e \Rightarrow z}$$

Açúcar sintático:

$$\begin{aligned} \neg e &\doteq \text{if } e \text{ then False else True} \\ e \vee e' &\doteq \text{if } e \text{ then True else } e' \\ e \wedge e' &\doteq \text{if } e \text{ then } e' \text{ else False} \\ \text{let } p_1 \equiv e_1, \dots, p_n \equiv e_n \text{ in } e &\doteq (\lambda p_1. \dots. p_n. e) e_1 \dots e_n \\ \lambda \langle p_1, \dots, p_n \rangle. e &\doteq \lambda v. \text{let } p_1 \equiv v. 1, \dots, p_n \equiv v. n \text{ in } e \\ \text{letrec } p_1 \equiv e_1 \text{ in } e &\doteq \text{let } p_1 \equiv \text{rec } (\lambda p_1. e_1) \text{ in } e \\ \text{letrec } p_1 \equiv e_1, \dots, p_n \equiv e_n \text{ in } e &\doteq \text{let } \langle p_1, \dots, p_n \rangle \equiv \text{rec } (\lambda \langle p_1, \dots, p_n \rangle. \langle e_1, \dots, e_n \rangle) \text{ in } e \end{aligned}$$