A Linguagem Imperativa WHILE

Sintaxe abstracta

$$\begin{array}{lcccl} \textbf{Aexp} & \ni & a & ::= & n \mid x \mid a_1 + a_2 \mid a_1 * a_2 \mid a_1 - a_2 \\ \textbf{Bexp} & \ni & b & ::= & \texttt{true} \mid \texttt{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid \neg b \mid b_1 \wedge b_2 \mid b_1 \vee b_2 \\ \textbf{Stm} & \ni & S & ::= & x := a \mid \texttt{skip} \mid S_1 \,;\, S_2 \mid \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2 \mid \texttt{while}\ b\ \texttt{do}\ S \end{array}$$

Semântica Natural

$$[\text{skip}_{\text{ns}}] \quad \frac{}{\langle \texttt{skip}, s\rangle \to s} \qquad [\text{if}^{\text{tt}}_{\text{ns}}] \quad \frac{\langle S_1, s\rangle \to s'}{\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2, s\rangle \to s'} \text{ se } \mathcal{B}[\![b]\!]\, s = \mathbf{tt}$$

$$[\text{ass}_{\text{ns}}] \quad \frac{}{\langle x := a, s\rangle \to s\,[x \mapsto \mathcal{A}[\![a]\!]\, s]} \qquad [\text{if}^{\text{ff}}_{\text{ns}}] \quad \frac{\langle S_2, s\rangle \to s'}{\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2, s\rangle \to s'} \text{ se } \mathcal{B}[\![b]\!]\, s = \mathbf{ff}$$

$$[\text{comp}_{\text{ns}}] \quad \frac{\langle S_1, s\rangle \to s' \qquad \langle S_2, s'\rangle \to s''}{\langle S_1 \,;\, S_2, s\rangle \to s''} \qquad [\text{while}^{\text{tt}}_{\text{ns}}] \quad \frac{\langle S, s\rangle \to s' \qquad \langle \texttt{while}\ b\ \texttt{do}\ S, s'\rangle \to s''}{\langle \texttt{while}\ b\ \texttt{do}\ S, s\rangle \to s''} \text{ se } \mathcal{B}[\![b]\!]\, s = \mathbf{tt}$$

$$[\text{while}^{\text{ff}}_{\text{ns}}] \quad \frac{}{\langle \texttt{while}\ b\ \texttt{do}\ S, s\rangle \to s} \text{ se } \mathcal{B}[\![b]\!]\, s = \mathbf{ff}$$

$$\mathcal{S}_{\text{ns}} : \mathbf{Stm} \to (\mathbf{State} \hookrightarrow \mathbf{State})$$
$$\mathcal{S}_{\text{ns}}[\![S]\!]\, s = \begin{cases} s' & \text{, se } \langle S, s\rangle \to s' \\ \underline{\text{undef}} & \text{, nos outros casos} \end{cases}$$

Semântica Operacional Estrutural

$$[\text{skip}_{\text{sos}}] \quad \frac{}{\langle \texttt{skip}, s\rangle \Rightarrow s} \qquad [\text{if}^{\text{tt}}_{\text{sos}}] \quad \frac{}{\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2, s\rangle \Rightarrow \langle S_1, s\rangle} \text{ se } \mathcal{B}[\![b]\!]\, s = \mathbf{tt}$$

$$[\text{ass}_{\text{sos}}] \quad \frac{}{\langle x := a, s\rangle \Rightarrow s\,[x \mapsto \mathcal{A}[\![a]\!]\, s]} \qquad [\text{if}^{\text{ff}}_{\text{sos}}] \quad \frac{}{\langle \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2, s\rangle \Rightarrow \langle S_2, s\rangle} \text{ se } \mathcal{B}[\![b]\!]\, s = \mathbf{ff}$$

$$[\text{comp}^1_{\text{sos}}] \quad \frac{\langle S_1, s\rangle \Rightarrow \langle S_1', s'\rangle}{\langle S_1 \,;\, S_2, s\rangle \Rightarrow \langle S_1' \,;\, S_2, s'\rangle} \qquad [\text{comp}^2_{\text{sos}}] \quad \frac{\langle S_1, s\rangle \Rightarrow s'}{\langle S_1 \,;\, S_2, s\rangle \Rightarrow \langle S_2, s'\rangle}$$

$$[\text{while}_{\text{sos}}] \quad \frac{}{\langle \texttt{while}\ b\ \texttt{do}\ S, s\rangle \Rightarrow \langle \texttt{if}\ b\ \texttt{then}\ (S \,;\, \texttt{while}\ b\ \texttt{do}\ S)\ \texttt{else}\ \texttt{skip}, s\rangle}$$

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

A maquina abstracta **AM**

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

$$
\begin{array}{lcl}
inst & ::= & \textsc{push-}n \mid \textsc{add} \mid \textsc{mult} \mid \textsc{sub} \\
 & \mid & \textsc{true} \mid \textsc{false} \mid \textsc{eq} \mid \textsc{le} \mid \textsc{and} \mid \textsc{neg} \\
 & \mid & \textsc{fetch-}x \mid \textsc{store-}x \\
 & \mid & \textsc{noop} \mid \textsc{branch}(c,\ c) \mid \textsc{loop}(c,\ c) \\
c & ::= & \varepsilon \mid inst{:}c
\end{array}
$$

Code é a categoria sintática das sequências de instruções.

$$\mathbf{Stack} = (\mathbf{Z} \cup \mathbf{T})^\star \qquad \mathbf{State} = \mathbf{Var} \to \mathbf{Z}$$

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

Funções de tradução:

$$\mathcal{CA}\colon \mathbf{Aexp} \to \mathbf{Code}$$
$$\mathcal{CB}\colon \mathbf{Bexp} \to \mathbf{Code}$$
$$\mathcal{CS}\colon \mathbf{Stm} \to \mathbf{Code}$$

Semântica operacional de **AM**

$$
\begin{array}{lcll}
\langle \textsc{push-}n{:}c,\ e,\ s\rangle & \rhd & \langle c,\ \mathcal{N}[\![n]\!]{:}e,\ s\rangle & \\
\langle \textsc{add}{:}c,\ z_1{:}z_2{:}e,\ s\rangle & \rhd & \langle c,\ (z_1{+}z_2){:}e,\ s\rangle & \text{if } z_1,\ z_2{\in}\mathbf{Z} \\
\langle \textsc{mult}{:}c,\ z_1{:}z_2{:}e,\ s\rangle & \rhd & \langle c,\ (z_1{\star}z_2){:}e,\ s\rangle & \text{if } z_1,\ z_2{\in}\mathbf{Z} \\
\langle \textsc{sub}{:}c,\ z_1{:}z_2{:}e,\ s\rangle & \rhd & \langle c,\ (z_1{-}z_2){:}e,\ s\rangle & \text{if } z_1,\ z_2{\in}\mathbf{Z} \\
\langle \textsc{true}{:}c,\ e,\ s\rangle & \rhd & \langle c,\ \mathbf{tt}{:}e,\ s\rangle & \\
\langle \textsc{false}{:}c,\ e,\ s\rangle & \rhd & \langle c,\ \mathbf{ff}{:}e,\ s\rangle & \\
\langle \textsc{eq}{:}c,\ z_1{:}z_2{:}e,\ s\rangle & \rhd & \langle c,\ (z_1{=}z_2){:}e,\ s\rangle & \text{if } z_1,\ z_2{\in}\mathbf{Z} \\
\langle \textsc{le}{:}c,\ z_1{:}z_2{:}e,\ s\rangle & \rhd & \langle c,\ (z_1{\leq}z_2){:}e,\ s\rangle & \text{if } z_1,\ z_2{\in}\mathbf{Z} \\
\langle \textsc{and}{:}c,\ t_1{:}t_2{:}e,\ s\rangle & \rhd & & \\
 & \multicolumn{3}{l}{\begin{cases} \langle c, \mathbf{tt} : e, s\rangle & \text{if } t_1{=}\mathbf{tt} \text{ and } t_2{=}\mathbf{tt} \\ \langle c, \mathbf{ff} : e, s\rangle & \text{if } t_1{=}\mathbf{ff} \text{ or } t_2{=}\mathbf{ff},\ t_1,\ t_2{\in}\mathbf{T} \end{cases}} \\
\langle \textsc{neg}{:}c,\ t{:}e,\ s\rangle & \rhd & \begin{cases} \langle c, \mathbf{ff} : e, s\rangle & \text{if } t{=}\mathbf{tt} \\ \langle c, \mathbf{tt} : e, s\rangle & \text{if } t{=}\mathbf{ff} \end{cases} & \\
\langle \textsc{fetch-}x{:}c,\ e,\ s\rangle & \rhd & \langle c,\ (s\ x){:}e,\ s\rangle & \\
\langle \textsc{store-}x{:}c,\ z{:}e,\ s\rangle & \rhd & \langle c,\ e,\ s[x{\mapsto}z]\rangle & \text{if } z{\in}\mathbf{Z} \\
\langle \textsc{noop}{:}c,\ e,\ s\rangle & \rhd & \langle c,\ e,\ s\rangle & \\
\langle \textsc{branch}(c_1,\ c_2){:}c,\ t{:}e,\ s\rangle & \rhd & \begin{cases} \langle c_1 : c, e, s\rangle & \text{if } t{=}\mathbf{tt} \\ \langle c_2 : c, e, s\rangle & \text{if } t{=}\mathbf{ff} \end{cases} & \\
\langle \textsc{loop}(c_1,\ c_2){:}c,\ e,\ s\rangle & \rhd & & \\
\multicolumn{4}{c}{\langle c_1{:}\textsc{branch}(c_2{:}\textsc{loop}(c_1,\ c_2),\ \textsc{noop}){:}c,\ e,\ s\rangle}
\end{array}
$$

Semântica Axiomática

$$[\text{skip}_\text{p}] \quad \frac{}{\{\phi\}\ \texttt{skip}\ \{\phi\}} \qquad [\text{ass}_\text{p}] \quad \frac{}{\{\psi[e/x]\}\ x := e\ \{\psi\}}$$

$$[\text{comp}_\text{p}] \quad \frac{\{\phi\}\ S_1\ \{\theta\} \qquad \{\theta\}\ S_2\ \{\psi\}}{\{\phi\}\ S_1\ ;\ S_2\ \{\psi\}} \qquad [\text{if}_\text{p}] \quad \frac{\{\phi \wedge b\}\ S_t\ \{\psi\} \qquad \{\phi \wedge \neg b\}\ S_f\ \{\psi\}}{\{\phi\}\ \texttt{if}\ b\ \texttt{then}\ S_t\ \texttt{else}\ S_f\ \{\psi\}}$$

$$[\text{while}_\text{p}] \quad \frac{\{\theta \wedge b\}\ S\ \{\theta\}}{\{\theta\}\ \texttt{while}\ b\ \texttt{do}\ S\ \{\theta \wedge \neg b\}} \qquad [\text{cons}_\text{p}] \quad \frac{\{\phi\}\ S\ \{\psi\}}{\{\phi'\}\ S\ \{\psi'\}} \ \text{ se } \ \phi' \Rightarrow \phi \ \text{ e } \ \psi \Rightarrow \psi'$$

Semântica Denotacional

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

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

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

$\mathcal{S}_{\text{ds}}[\![S_1\ ;\ S_2]\!] = \mathcal{S}_{\text{ds}}[\![S_2]\!] \circ \mathcal{S}_{\text{ds}}[\![S_1]\!]$

$\mathcal{S}_{\text{ds}}[\![\texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2]\!] = \text{cond}(\mathcal{B}[\![b]\!], \mathcal{S}_{\text{ds}}[\![S_1]\!], \mathcal{S}_{\text{ds}}[\![S_2]\!])$

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

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

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

Uma Linguagem Funcional

I. Sintaxe Abstracta:

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

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

$$\langle tag \rangle \quad ::= \quad 1 \mid 2 \mid \ldots$$

$$\langle pat \rangle \quad ::= \quad \langle var \rangle \ \mid \ \langle \langle pat \rangle, \ldots, \langle pat \rangle \rangle$$

II. Tipos:

Sintaxe abstracta

$$\begin{array}{rcl} \langle type \rangle & ::= & \mathbf{Int} \mid \mathbf{Bool} \mid \langle type \rangle \to \langle type \rangle \\ & \mid & \mathbf{Prod}(\langle type \rangle, \ldots, \langle type \rangle) \mid \mathbf{Sum}(\langle type \rangle, \ldots, \langle type \rangle) \mid \mathbf{List}\ \langle type \rangle \end{array}$$

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

Regras

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

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

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

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

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

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

$$\frac{\Gamma \vdash e : \theta_k}{\Gamma \vdash @k\, e : \theta_1 + \ldots + \theta_n}\ \text{para } k \in \{1, \ldots, n\} \qquad \frac{\Gamma \vdash e : \theta_1 + \ldots + \theta_n \qquad \Gamma \vdash e_1 : \theta_1 \to \theta \ \cdots\ \Gamma \vdash e_n : \theta_n \to \theta}{\Gamma \vdash \mathsf{sumcase}\ e\ \mathsf{of}\ (e_1, \ldots, e_n) : \theta}$$

$$\frac{}{\Gamma \vdash \mathsf{nil} : \mathbf{List}\ \theta} \qquad \frac{\Gamma \vdash e_1 : \theta \qquad \Gamma \vdash e_2 : \mathbf{List}\ \theta}{\Gamma \vdash (e_1 :: e_2) : \mathbf{List}\ \theta} \qquad \frac{\Gamma \vdash e : \mathbf{List}\ \theta \qquad \Gamma \vdash e_1 : \theta' \qquad \Gamma \vdash e_2 : \theta \to \mathbf{List}\ \theta \to \theta'}{\Gamma \vdash \mathsf{listcase}\ e\ \mathsf{of}\ (e_1, e_2) : \theta'}$$

$$\frac{\Gamma, p_1 : \theta_1, \ldots, p_n : \theta_n \vdash e : \theta}{\Gamma, \langle p_1, \ldots, p_n \rangle : \theta_1 \times \ldots \times \theta_n \vdash e : \theta} \qquad \frac{\Gamma \vdash e_1 : \theta_1 \ \ldots\ \Gamma \vdash e_n : \theta_n \qquad \Gamma, p_1 : \theta_1, \ldots, p_n : \theta_n \vdash e : \theta}{\Gamma \vdash \mathsf{let}\ p_1 \equiv e_1, \ldots, p_n \equiv e_n\ \mathsf{in}\ e : \theta}$$

$$\frac{\Gamma \vdash e : \theta \to \theta}{\Gamma \vdash \mathsf{rec}\ e : \theta} \qquad \frac{\Gamma' \vdash e_1 : \theta_1 \ \ldots\ \Gamma' \vdash e_n : \theta_n \qquad \Gamma' \vdash e : \theta}{\Gamma \vdash \mathsf{letrec}\ p_1 \equiv e_1, \ldots, p_n \equiv e_n\ \mathsf{in}\ e : \theta}\ \text{com } \Gamma' = \Gamma, p_1 : \theta_1, \ldots, p_n : \theta_n$$

III. Semântica de avaliação "call-by-value":

Formas canónicas

$$\begin{array}{rcl} \langle cfm \rangle & ::= & \lambda\langle var\rangle.\langle exp\rangle \mid \ldots \mid -2 \mid -1 \mid 0 \mid 1 \mid 2 \mid \ldots \mid \mathsf{True} \mid \mathsf{False} \\ & \mid & \langle\langle cfm\rangle, \ldots, \langle cfm\rangle\rangle \mid @\langle tag\rangle\langle cfm\rangle \mid \mathsf{nil} \mid \langle cfm\rangle :: \langle cfm\rangle \end{array}$$

Regras de avaliação

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

$$\frac{e_1 \Rightarrow \lfloor i_1 \rfloor \qquad e_2 \Rightarrow \lfloor i_2 \rfloor}{e_1 \ \mathbf{bop}\ e_2 \Rightarrow \lfloor i_1 \ \mathbf{bop}\ i_2 \rfloor} \text{ com } \mathbf{bop} \in \{+, -, *, =, \neq, <, >, \leq, \geq\} \text{ ou } \mathbf{bop} \in \{\mathsf{div}, \mathsf{mod}\} \text{ e } i_2 \neq 0$$

$$\frac{e \Rightarrow \lfloor i \rfloor}{-e \Rightarrow \lfloor -i \rfloor} \qquad \frac{e \Rightarrow \lfloor b \rfloor}{\neg e \Rightarrow \lfloor \neg b \rfloor} \qquad \frac{e_1 \Rightarrow \lfloor b_1 \rfloor \qquad e_2 \Rightarrow \lfloor b_2 \rfloor}{e_1 \ \mathbf{bop}\ e_2 \Rightarrow \lfloor b_1 \ \mathbf{bop}\ b_2 \rfloor} \text{ com } \mathbf{bop} \in \{\vee, \wedge\}$$

$$\frac{e_1 \Rightarrow \mathsf{True} \qquad e_2 \Rightarrow z}{\mathsf{if}\ e_1\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3 \Rightarrow z} \qquad \frac{e_1 \Rightarrow \mathsf{False} \qquad e_3 \Rightarrow z}{\mathsf{if}\ e_1\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3 \Rightarrow z}$$

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

$$\frac{e \Rightarrow z}{@k\,e \Rightarrow @k\,z} \qquad \frac{e \Rightarrow @k\,z \qquad e_k z \Rightarrow z'}{\mathsf{sumcase}\ e\ \mathsf{of}\ (e_1, \ldots, e_n) \Rightarrow z'}$$

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

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

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

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

$$\begin{array}{rcl} \mathsf{let}\ p_1 \equiv e_1, \ldots, p_n \equiv e_n\ \mathsf{in}\ e & \doteq & (\lambda p_1 \ldots . p_n.e)\,e_1 \cdots e_n \\ \lambda\langle p_1, \ldots, p_n\rangle.e & \doteq & \lambda v.\mathsf{let}\ p_1 \equiv v.1, \ldots, p_n \equiv v.n\ \mathsf{in}\ e \\ \mathsf{nil} & \doteq & @1\,\langle\rangle \\ e :: e' & \doteq & @2\,\langle e, e'\rangle \\ \mathsf{listcase}\ e\ \mathsf{of}\ (e_1, e_2) & \doteq & \mathsf{sumcase}\ e\ \mathsf{of}\ (\lambda\langle\rangle.e_1, \lambda\langle x, y\rangle.e_2\,x\,y) \end{array}$$

IV. Semântica de avaliação "call-by-name":

Formas canónicas

$$\begin{array}{rcl} \langle cfm \rangle & ::= & \lambda\langle var\rangle.\langle exp\rangle \mid \ldots \mid -2 \mid -1 \mid 0 \mid 1 \mid 2 \mid \ldots \mid \mathsf{True} \mid \mathsf{False} \\ & \mid & \langle\langle exp\rangle, \ldots, \langle exp\rangle\rangle \mid @\langle tag\rangle\langle exp\rangle \mid \mathsf{nil} \mid \langle exp\rangle :: \langle exp\rangle \end{array}$$

Regras de avaliação

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

$$\frac{e_1 \Rightarrow \lfloor i_1 \rfloor \qquad e_2 \Rightarrow \lfloor i_2 \rfloor}{e_1 \ \mathbf{bop}\ e_2 \Rightarrow \lfloor i_1 \ \mathbf{bop}\ i_2 \rfloor} \text{ para } \mathbf{bop} \in \{+, -, *, =, \neq, <, >, \leq, \geq\} \text{ ou } \mathbf{bop} \in \{\mathsf{div}, \mathsf{mod}\} \text{ e } i_2 \neq 0$$

$$\frac{e \Rightarrow \lfloor i \rfloor}{-e \Rightarrow \lfloor -i \rfloor} \qquad \frac{e_1 \Rightarrow \mathsf{True} \qquad e_2 \Rightarrow z}{\mathsf{if}\ e_1\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3 \Rightarrow z} \qquad \frac{e_1 \Rightarrow \mathsf{False} \qquad e_3 \Rightarrow z}{\mathsf{if}\ e_1\ \mathsf{then}\ e_2\ \mathsf{else}\ e_3 \Rightarrow z}$$

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

$$\frac{e \Rightarrow @k\, e' \qquad e_k\, e' \Rightarrow z}{\mathsf{sumcase}\ e\ \mathsf{of}\ (e_1, \ldots, e_n) \Rightarrow z}$$

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

$$\frac{e\,(\mathsf{rec}\, e) \Rightarrow z}{\mathsf{rec}\, e \Rightarrow z}$$

Açúcar sintático:

$$\begin{array}{rcl} \neg e & \doteq & \mathsf{if}\ e\ \mathsf{then}\ \mathsf{False}\ \mathsf{else}\ \mathsf{True} \\ e \ \vee\ e' & \doteq & \mathsf{if}\ e\ \mathsf{then}\ \mathsf{True}\ \mathsf{else}\ e' \\ e \ \wedge\ e' & \doteq & \mathsf{if}\ e\ \mathsf{then}\ e'\ \mathsf{else}\ \mathsf{False} \end{array}$$

$$\begin{array}{rcl} \mathsf{let}\ p_1 \equiv e_1, \ldots, p_n \equiv e_n\ \mathsf{in}\ e & \doteq & (\lambda p_1 \ldots p_n.e)\, e_1 \cdots e_n \\ \lambda\langle p_1, \ldots, p_n\rangle.e & \doteq & \lambda v.\, \mathsf{let}\ p_1 \equiv v.1, \ldots, p_n \equiv v.n\ \mathsf{in}\ e \end{array}$$

$$\mathsf{letrec}\ p_1 \equiv e_1\ \mathsf{in}\ e \ \doteq\ \mathsf{let}\ p_1 \equiv \mathsf{rec}\,(\lambda p_1.\, e_1)\ \mathsf{in}\ e$$

$$\begin{array}{l} \mathsf{letrec}\ p_1 \equiv e_1, \ldots, p_n \equiv e_n\ \mathsf{in}\ e \\ \quad \doteq\ \ \mathsf{let}\ \langle p_1, \ldots, p_n\rangle \equiv \mathsf{rec}\,(\lambda\langle p_1, \ldots, p_n\rangle.\, \langle e_1, \ldots, e_n\rangle)\ \mathsf{in}\ e \end{array}$$