

Homework 3

ENE4014 Programming Languages, Spring 2019

Woosuk Lee

due: 5/13(Wed), 24:00

Exercise 1 Consider the following programming language, called `miniML`, that features (recursive) procedures and explicit references.

Syntax The syntax is defined as follows:

$$\begin{array}{lcl}
 P & \rightarrow & E \\
 E & \rightarrow & n \\
 & | & x \\
 & | & E + E \mid E - E \mid E * E \mid E / E \\
 & | & E - E \\
 & | & \text{iszero } E \\
 & | & \text{if } E \text{ then } E \text{ else } E \\
 & | & \text{let } x = E \text{ in } E \\
 & | & \text{letrec } f(x) = E \text{ in } E \\
 & | & \text{proc } x E \\
 & | & E E \\
 & | & \text{ref } E \\
 & | & ! E \\
 & | & E := E \\
 & | & E; E \\
 & | & \text{begin } E \text{ end}
 \end{array}$$

Semantics The semantics is defined with the following domain:

$$\begin{array}{lcl}
 Val & = & \mathbb{Z} + Bool + Procedure + RecProcedure + Loc \\
 Procedure & = & Var \times E \times Env \\
 RecProcedure & = & Var \times Var \times E \times Env \\
 \rho \in Env & = & Var \rightarrow Val \\
 \sigma \in Mem & = & Loc \rightarrow Val
 \end{array}$$

and evaluation rules:

$$\begin{array}{c}
\frac{}{\rho, \sigma \vdash n \Rightarrow n, \sigma} \quad \frac{}{\rho, \sigma \vdash x \Rightarrow \rho(x), \sigma} \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow n_1, \sigma_1 \quad \rho, \sigma_1 \vdash E_2 \Rightarrow n_2, \sigma_2}{\rho, \sigma_0 \vdash E_1 \oplus E_2 \Rightarrow n_1 \oplus n_2, \sigma_2} \quad \oplus \in \{+, -, *, /\} \\
\\
\frac{\rho, \sigma_0 \vdash E \Rightarrow 0, \sigma_1}{\rho, \sigma_0 \vdash \text{iszero } E \Rightarrow \text{true}, \sigma_1} \quad \frac{\rho, \sigma_0 \vdash E \Rightarrow n, \sigma_1}{\rho, \sigma_0 \vdash \text{iszero } E \Rightarrow \text{false}, \sigma_1} \quad n \neq 0 \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow \text{true}, \sigma_1 \quad \rho, \sigma_1 \vdash E_2 \Rightarrow v, \sigma_2}{\rho, \sigma_0 \vdash \text{if } E_1 \text{ then } E_2 \text{ else } E_3 \Rightarrow v, \sigma_2} \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow \text{false}, \sigma_1 \quad \rho, \sigma_1 \vdash E_3 \Rightarrow v, \sigma_2}{\rho, \sigma_0 \vdash \text{if } E_1 \text{ then } E_2 \text{ else } E_3 \Rightarrow v, \sigma_2} \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow v_1, \sigma_1 \quad [x \mapsto v_1]\rho, \sigma_1 \vdash E_2 \Rightarrow v, \sigma_2}{\rho, \sigma_0 \vdash \text{let } x = E_1 \text{ in } E_2 \Rightarrow v, \sigma_2} \\
\\
\frac{[f \mapsto (f, x, E_1, \rho)]\rho, \sigma_0 \vdash E_2 \Rightarrow v, \sigma_1}{\rho, \sigma_0 \vdash \text{letrec } f(x) = E_1 \text{ in } E_2 \Rightarrow v, \sigma_1} \\
\\
\frac{}{\rho, \sigma \vdash \text{proc } x \ E \Rightarrow (x, E, \rho), \sigma} \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow (x, E, \rho'), \sigma_1 \quad \rho, \sigma_1 \vdash E_2 \Rightarrow v, \sigma_2 \quad [x \mapsto v]\rho', \sigma_2 \vdash E \Rightarrow v', \sigma_3}{\rho, \sigma_0 \vdash E_1 \ E_2 \Rightarrow v', \sigma_3} \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow (f, x, E, \rho'), \sigma_1 \quad \rho, \sigma_1 \vdash E_2 \Rightarrow v, \sigma_2 \quad [x \mapsto v, f \mapsto (f, x, E, \rho')]\rho', \sigma_2 \vdash E \Rightarrow v', \sigma_3}{\rho, \sigma_0 \vdash E_1 \ E_2 \Rightarrow v', \sigma_3} \\
\\
\frac{\rho, \sigma_0 \vdash E \Rightarrow v, \sigma_1}{\rho, \sigma_0 \vdash \text{ref } E \Rightarrow l, [l \mapsto v]\sigma_1} \quad l \notin \text{Dom}(\sigma_1) \\
\\
\frac{\rho, \sigma_0 \vdash E \Rightarrow l, \sigma_1}{\rho, \sigma_0 \vdash ! E \Rightarrow \sigma_1(l), \sigma_1} \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow l, \sigma_1 \quad \rho, \sigma_1 \vdash E_2 \Rightarrow v, \sigma_2}{\rho, \sigma_0 \vdash E_1 := E_2 \Rightarrow v, [l \mapsto v]\sigma_2} \\
\\
\frac{\rho, \sigma_0 \vdash E_1 \Rightarrow v_1, \sigma_1 \quad \rho, \sigma_1 \vdash E_2 \Rightarrow v_2, \sigma_2}{\rho, \sigma_0 \vdash E_1; E_2 \Rightarrow v_2, \sigma_2} \\
\\
\frac{\rho, \sigma_0 \vdash E \Rightarrow v, \sigma_1}{\rho, \sigma_0 \vdash \text{begin } E \text{ end} \Rightarrow v, \sigma_1}
\end{array}$$

Implement an interpreter of `miniML`. Raise an exception `UndefinedSemantics` whenever the semantics is undefined. Skeleton code will be provided (before you start, see `README.md`).