Throughout this document, \( \rho \) is an environment mapping variable names to values. The term \( \rho(x) \) is the value assigned to the variable \( x \). It is assumed that the environment is represented as an ordered list of variable name/value pairs, and that it is searched from the front with the first match returned. Similarly, \( \sigma \) is an environment storing the definitions of functions. The term \( \sigma(f) \) is the definition of the function \( f \), represented as a pair of the list of formal parameters and the expression forming the body of the function.

**Evaluation of literals and variables:**

In these three rules, \( n \) is an integer literal, \( r \) is a floating point literal, and \( x \) is an identifier.

\[
\begin{align*}
(n, \rho, \sigma) & \downarrow (\text{Int } n) \\
(r, \rho, \sigma) & \downarrow (\text{Real } r) \\
(x, \rho, \sigma) & \downarrow \rho(x)
\end{align*}
\]

**Evaluation of built-in operators:**

In the following four rules, \( \bullet \) on the left of an arrow is any one of \( +, - \), or \( \ast \); on the right of an arrow it is the corresponding one of \( +, - \), or \( \ast \).

\[
\begin{align*}
(e_1, \rho, \sigma) & \downarrow (\text{Int } i_1) \quad (e_2, \rho, \sigma) \downarrow (\text{Int } i_2) \\
((\bullet e_1 e_2), \rho, \sigma) & \downarrow (\text{Int } (i_1 \bullet i_2))
\end{align*}
\]

\[
\begin{align*}
(e_1, \rho, \sigma) & \downarrow (\text{Real } r_1) \quad (e_2, \rho, \sigma) \downarrow (\text{Real } r_2) \\
((\bullet e_1 e_2), \rho, \sigma) & \downarrow (\text{Real } (r_1 \bullet r_2))
\end{align*}
\]

\[
\begin{align*}
(e_1, \rho, \sigma) & \downarrow (\text{Real } r_1) \quad (e_2, \rho, \sigma) \downarrow (\text{Int } i_2) \\
((\bullet e_1 e_2), \rho, \sigma) & \downarrow (\text{Real } (r_1 \bullet (i_2 \text{ as real})))
\end{align*}
\]

\[
\begin{align*}
(e_1, \rho, \sigma) & \downarrow (\text{Real } r_1) \quad (e_2, \rho, \sigma) \downarrow (\text{Real } r_2) \\
((\bullet e_1 e_2), \rho, \sigma) & \downarrow (\text{Real } ((i_1 \text{ as real})/i_2 \text{ as real}))
\end{align*}
\]

\[
\begin{align*}
(e_1, \rho, \sigma) & \downarrow (\text{Int } i_1) \quad (e_2, \rho, \sigma) \downarrow (\text{Real } r_2) \\
((\bullet e_1 e_2), \rho, \sigma) & \downarrow (\text{Real } (r_1/r_2))
\end{align*}
\]

\[
\begin{align*}
(e_1, \rho, \sigma) & \downarrow (\text{Real } r_1) \quad (e_2, \rho, \sigma) \downarrow (\text{Int } i_2) \\
((\bullet e_1 e_2), \rho, \sigma) & \downarrow (\text{Real } (r_1/(i_2 \text{ as real})))
\end{align*}
\]

\[
\begin{align*}
(e_1, \rho, \sigma) & \downarrow (\text{Real } r_1) \quad (e_2, \rho, \sigma) \downarrow (\text{Real } r_2) \\
((\bullet e_1 e_2), \rho, \sigma) & \downarrow (\text{Real } ((i_1 \text{ as real})/r_2))
\end{align*}
\]
Evaluation of user-defined operators:

In the following rule, \( @ \) is the list append function, and \( \text{zip} \) is the function which, given two lists of the same length, returns a list of pairs of corresponding elements.

\[
(e_1, \rho, \sigma) \downarrow v_1 \quad \cdots \quad (e_n, \rho, \sigma) \downarrow v_n \quad (2^n(\sigma(f)), \text{zip}(1^n(\sigma(f)), \lbrack v_1, \ldots, v_n \rbrack) @, \rho, \sigma) \downarrow v \\
(f \ e_1 \cdots e_n), \rho, \sigma) \downarrow v 
\]

Note that if there is a user-defined operator with the same name as one of the built-in operators, the user definition takes precedence.

Evaluation of Commands:

While expressions (in the context of variable and function definitions) evaluate to values, commands evaluate to new contexts.

\[
((\text{exit}), \rho, \sigma) \downarrow \bot \quad \text{(The system shuts down.)} \\
((\text{bindings}), \rho, \sigma) \downarrow (\rho, \sigma) \quad \text{(The current bindings are displayed.)} \\
(\text{exp}, \rho, \sigma) \downarrow v \\
((\text{define var exp}), \rho, \sigma) \downarrow (((\text{var}, v) :: \rho), \sigma) \\
((\text{defun ((opr :: vars)) exp}), \rho, \sigma) \downarrow (\rho, (\text{opr}, \text{vars}, \text{exp}) :: \sigma))
\]

Composition of Commands:

\[
(c_1, \rho, \sigma) \downarrow (\rho', \sigma') \quad (c_2, \rho', \sigma') \downarrow (\rho'', \sigma'') \\
((c_1 ; c_2), \rho, \sigma) \downarrow (\rho'', \sigma''))
\]