Semantics for MP2

The following rules describe the semantics for the language you are interpreting.

Expressions:

For `IntExp` and `BoolExp`:

$$\frac{}{(t, env) \Downarrow \texttt{IntVal}\,t}\; t \text{ an integer constant}$$

$$\frac{}{(t, env) \Downarrow \texttt{BoolVal}\,t}\; t \text{ an boolean constant}$$

For `IntOpExp`:

$$\frac{(e_1, env) \Downarrow v_1 \quad (e_2, env) \Downarrow v_2}{(e_1 \oplus e_2, env) \Downarrow (\texttt{IntVal}\,v_1 \oplus v_2)}\; \oplus \text{ an integer operation besides division}$$

$$\frac{(e_1, env) \Downarrow v_1 \quad (e_2, env) \Downarrow \texttt{IntVal 0}}{(e_1 \,/\, e_2, env) \Downarrow (\texttt{ExnVal "Division by 0"})}$$

$$\frac{(e_1, env) \Downarrow v_1 \quad (e_2, env) \Downarrow v_2}{(e_1 \,/\, e_2, env) \Downarrow (\texttt{IntVal}\,v_1 \,/\, v_2)}\; v_2 \neq \texttt{IntVal 0}$$

$$\frac{}{(e_1 \oplus e_2, env) \Downarrow \texttt{ExnVal "No matching operator"}}\; \oplus \text{ not a valid integer operation}$$

For `BoolOpExp`:

$$\frac{(e_1, env) \Downarrow v_1 \quad (e_2, env) \Downarrow v_2}{(e_1 \oplus e_2, env) \Downarrow (\texttt{BoolVal}\,v_1 \oplus v_2)}\; \oplus \text{ a boolean operation}$$

$$\frac{}{(e_1 \oplus e_2, env) \Downarrow \texttt{ExnVal "No matching operator"}}\; \oplus \text{ not a valid boolean operation}$$

For `CompOpExp`:

$$\frac{(e_1, env) \Downarrow v_1 \quad (e_2, env) \Downarrow v_2}{(e_1 \oplus e_2, env) \Downarrow (\texttt{BoolVal}\,v_1 \oplus v_2)}\; \oplus \text{ an integer comparison operation}$$

$$\frac{}{(e_1 \oplus e_2, env) \Downarrow \texttt{ExnVal "No matching operator"}}\; \oplus \text{ not a valid integer comparison operation}$$

For `IfExp`:

$$\frac{(e_1, env) \Downarrow \texttt{true} \quad (e_2, env) \Downarrow v_2}{(\texttt{if } e_1 \texttt{ then } e_2 \texttt{ else } e_3, env) \Downarrow v_2}$$

$$\frac{(e_1, env) \Downarrow \texttt{false} \quad (e_3, env) \Downarrow v_3}{(\texttt{if } e_1 \texttt{ then } e_2 \texttt{ else } e_3, env) \Downarrow v_3}$$

$$\frac{(e_1, env) \Downarrow v}{(\texttt{if } e_1 \texttt{ then } e_2 \texttt{ else } e_3, env) \Downarrow \texttt{ExnVal "Condition is not a Bool"}}\; v \text{ is not a BoolVal}$$

For `VarExp`:

$$\frac{}{(\text{x}, env) \Downarrow v} \; x \in env, v = env(x)$$

$$\frac{}{(\text{x}, env) \Downarrow \text{ExnVal "No match in env"}} \; x \notin env$$

For `FunExp`:

$$\frac{}{(\text{fn}[x_1, ..., x_n] \; body \; \text{end;}, env) \Downarrow \langle [x_1, ..., x_n], body, env \rangle}$$

For `AppExp`:

$$\frac{(f, env) \Downarrow \langle x_1, ..., x_n, body, env' \rangle \quad (e_1, env) \Downarrow v_1 ... (e_n, env) \Downarrow v_n \quad (body, env' + \{x_1 \mapsto v_1, ..., x_n \mapsto v_n\}) \Downarrow v_f}{(\text{apply} \; f(e_1, ..., e_n) \text{;}, env) \Downarrow v_f} \; n \geq 0$$

$$\frac{(f, env) \Downarrow v}{(\text{apply} \; f(e_1, ..., e_n) \text{;}, env) \Downarrow \text{ExnVal "Apply to non-closure"}} \; n \geq 0, v \text{ is not a CloVal}$$

For `LetExp`:

$$\frac{(e_1, env) \Downarrow v_1 \quad ... \quad (e_n, env) \Downarrow v_n \quad (e_f, env + \{x_1 \mapsto v_1, ..., x_n \mapsto v_n\}) \Downarrow v_f}{(\text{let}[x_1 := e_1, ..., x_n := e_n] e_f, env) \Downarrow v_f} \; n \geq 0$$

Statements:

For `SeqStmt`:

$$\frac{(S_1, penv, env) \Downarrow (p_1, penv', env') \quad (S_2, penv', env') \Downarrow (p_2, penv'', env'')}{(S_1; S_2, penv, env) \Downarrow (p_1 \; \text{++} \; p_2, penv'', env'')}$$

For `IfStmt`:

$$\frac{(e_1, env) \Downarrow \text{true} \quad (S_2, penv, env) \Downarrow (p_2, penv', env')}{(\text{if } e_1 \text{ then } S_2 \text{ else } S_3, penv, env) \Downarrow (p_2, penv', env')}$$

$$\frac{(e_1, env) \Downarrow \text{false} \quad (S_3, penv, env) \Downarrow (p_3, penv', env')}{(\text{if } e_1 \text{ then } S_2 \text{ else } S_3, penv, env) \Downarrow (p_3, penv', env')}$$

$$\frac{(e_1, env) \Downarrow v}{(\text{if } e_1 \text{ then } S_2 \text{ else } S_3, penv, env) \Downarrow (\text{"exn: Condition is not a Bool"}, penv, env)} \; v \text{ is not a BoolVal}$$

For `SetStmt`:

$$\frac{(e, env) \Downarrow v_1}{(\text{x := } e, penv, env) \Downarrow (\text{""}, penv, env + \{\text{x} \mapsto v_1\})}$$

For `ProcedureStmt`:

$$\frac{}{(\text{procedure } f(ps) \; body; \; \text{endproc;}, penv, env) \Downarrow (\text{""}, penv + \{f \mapsto (\text{ProcedureStmt } f \; ps \; body)\}, env)}$$

For `CallStmt`:

$$\frac{A \quad (body, penv, env' = env + \{x_1 \mapsto v_1, \; ... \; , x_n \mapsto v_n\}) \Downarrow (p_f, penv', env'')}{(\text{call } f(e_1, \; ... \; , e_n), penv, env) \Downarrow (p_f, penv', env'')} \; n \geq 0, f \in penv$$

where
$$
\begin{aligned}
A = \quad & (e_1, env) \Downarrow v_1 \quad ... \quad (e_n, env) \Downarrow v_n \\
penv(f) = \quad & (\text{ProcedureStmt } f \; ps \; body) \\
ps = \quad & x_1 \; ... \; x_n
\end{aligned}
$$

$$\frac{}{(\text{call } f(e_1, ..., e_n), penv, env) \Downarrow (\text{"Procedure } f \text{ undefined"}, penv, env)} \; f \notin penv$$