1 Modules and Paramertized Abstractions

exercise 1.1. To get some practice working with modules, do problem 13.1 on pp. 71 of the Schmidt text.

exercise 1.2. Read Chapter 3 up to section 3.7 on page 90.

exercise 1.3. Complete the interpreter (code provided) to support procedures parametrized by expressions under call-by-name (lazy) semantics. The syntax, typing rules and semantic equations for this extension are found in Schmidt. You will need to extend meaning_of_declaration to properly handle procedure declarations with lazily evaluated $\tau$exp parameters that will be evaluated lazily.

The equations to implement are on page 80.

\[
\llbracket \pi \vdash \text{proc } I_1(I_2 : \tau_{\text{exp}}) = C : \{I_1 : \tau_{\text{exp}} \rightarrow \text{comm}\} \text{dec} \rrbracket e s = (\{I_1 : p\}, s)
\]

where $p f s' = \llbracket \pi \cup \{I_2 : \tau_{\text{exp}}\} : C : \text{comm} \rrbracket (e \cup \{I_2 : f\}) s'$.  

This rule becomes two rules when we specialize $\tau$ to be $\text{int}$ or $\text{bool}$ as follows:

\[
\llbracket \pi \vdash \text{proc } I_1(I_2 : \text{int}_{\text{exp}}) = C : \{I_1 : \text{int}_{\text{exp}} \rightarrow \text{comm}\} \text{dec} \rrbracket e s = (\{I_1 : p\}, s)
\]

where $p f s' = \llbracket \pi \cup \{I_2 : \text{int}_{\text{exp}}\} : C : \text{comm} \rrbracket (e \cup \{I_2 : f\}) s'$.  

\[
\llbracket \pi \vdash \text{proc } I_1(I_2 : \text{bool}_{\text{exp}}) = C : \{I_1 : \text{bool}_{\text{exp}} \rightarrow \text{comm}\} \text{dec} \rrbracket e s = (\{I_1 : p\}, s)
\]

where $p f s' = \llbracket \pi \cup \{I_2 : \text{bool}_{\text{exp}}\} : C : \text{comm} \rrbracket (e \cup \{I_2 : f\}) s'$.  

For the first rule, you update the proc_int environment and for the second you must update the proc_bool environment.

Note that for type assignments, $\pi_1 \vdash \pi_2$ is implemented by the following ML code: (bar_union_ta pi1 pi2) and for environments $e_1 \vdash e_2$ is implemented by the call (union_env e1 e2).

For completeness I include the semantics of the call here as well (though it is implemented in the code provided).

\[
\llbracket \pi \vdash \text{call } I_1(E) : \text{comm} \rrbracket e s = p f s
\]

where $\langle I_1, p \rangle \in e$ and $f s' = \llbracket \pi \vdash E : \tau_{\text{exp}} \rrbracket e s'$.  

1Note that the line above the semantic equations on page 80 should say

\[
p \in \llbracket \tau_{\text{exp}} \rrbracket \rightarrow \llbracket \text{comm} \rrbracket
\]

The inclusion of the Env argument to $p$ is incorrect. You can see this in the definition of the type env in the code.