Homework 4
Due Friday, 4/29/11.

Please turn in your programs in the dropbox on Blackboard and hand in written assignments and listings before the beginning of class on the due date.

1. (15 points) **Adding Closures to the Lisp interpreter**
   Add closures to the Lisp interpreter (modify your version or my sample solution to HW 1) and show that this makes the interpreter use static instead of dynamic scoping by re-running the example from HW 1. Evaluate the following expression using the original interpreter and your modified version:

   \[
   \left((\text{lambda} \ (x) \ (\text{let} \ ((\text{makeInc} \ (\text{lambda} \ (\text{inc}) \ (\text{lambda} \ (x) \ (+ \ x \ \text{inc})))))) \right)\\
   \left((\text{let} \ ((\text{inc1} \ (\text{makeInc} \ 1)) \ (\text{inc5} \ (\text{makeInc} \ 5))) \right)\\
   (+ \ (\text{inc1} \ x) \ (\text{inc5} \ x)))) \right) 3)
   \]

   What happens in each case? Why does the original interpreter behave as it does, and why is the behavior different for the modified interpreter?

2. (20 points) **Let expressions**
   Suppose we extend the grammar for typed PCF that we saw in class to include “let” expressions:

   \[e ::= \text{let} \ (x: T) = e_1 \ in \ e_2 \ \text{end}\]

   Thus we could write

   \[\text{let} \ (x: \text{Int}) = 17 \ in \ \text{succ} (\text{succ} \ x) \ \text{end}\]

   (a) Please write down the type-checking rule for the let expression in the same form as in the lecture. That is, the bottom line (below the long horizontal line) should read

   \[E \vdash \text{let} \ (x:T) = e_1 \ in \ e_2 \ \text{end} : U\]

   Write down the complete rule including all of the hypotheses necessary to prove the typing of the let expression.

   (b) Please write down the evaluation rule for the new let expression. As in the lecture, the bottom line should read:

   \[\text{(let} \ (x:T) = e_1 \ in \ e_2 \ \text{end}, \ \text{env}) \Rightarrow v\]

   Again, write down the entire rule, including the hypotheses that generate that conclusion.

   (c) Add let expressions to TypeCheck.hs, updating the lexer and parser as needed.

3. (20 points) **Activation Records for Inline Blocks**
   Please do problem 7.1 from Mitchell, page 191.
4. (10 points) **Static and Dynamic Scope**
   Please do problem 7.8 from Mitchell, page 196.

5. (10 points) **Eval and Scope**
   Please do problem 7.10 from Mitchell, page 197. You only need to do parts b and c, but please read part a so that the rest makes sense.

6. (20 points) **Lambda Calculus and Scope**
   Please do problem 7.11 from Mitchell, page 198.