Homework 2

Due Friday, 9/19/08

Please turn in your homework solutions at the beginning of class.

1. (10 points) **Parsing**

   Please do problem 4.1 from Mitchell, page 83.

2. (10 points) **Parsing**

   Please do problem 4.2 from Mitchell, page 83.

3. (5 points) **Lambda Calculus Reduction**

   Please do problem 4.3 from Mitchell, page 83.

4. (10 points) **Symbolic Reduction**

   Please do problem 4.4 from Mitchell, page 83.

5. (10 points) **Lambda Reduction with Sugar**

   Here is a "sugared" lambda-expression using `let` declarations:

   $$\begin{array}{l} \texttt{let}\ \mathit{compose} = \lambda f.\, \lambda g.\, \lambda x.\, f(g\,x)\ \texttt{in} \\ \quad \texttt{let}\ h = \lambda x.\, x + x\ \texttt{in} \\ \qquad ((\mathit{compose}\, h)\, h)\, 3 \end{array}$$

   The "de-sugared" lambda-expression, obtained by replacing each $\texttt{let}\ z = U\ \texttt{in}\ V$ by $(\lambda z.\, V)\, U$ is

   $$\begin{array}{l} (\lambda \mathit{compose}. \\ \quad (\lambda h.\, ((\mathit{compose}\ h)\ h)\ 3)\ \ (\lambda x.\, x + x)) \\ \quad (\lambda f.\, \lambda g.\, \lambda x.\, f(g\ x)) \end{array}$$

   This is written using the same variable names as the `let`-form in order to make it easier to read the expression.

   Simplify the desugared lambda expression using reduction. Write one or two sentences explaining why the simplified expression is the answer you expected.

6. (20 points) **Defining Terms in Lambda Calculus**

   In class we defined Church numerals and booleans, and showed how to define more complex functions in the pure lambda calculus. Please show how to define the following functions in the pure lambda calculus.

   (a) (5 points)

   Define a function *Minus* such that Minus m n = m - n if m > n and 0 otherwise. Do not use recursion, but instead define it directly.

   (b) (5 points)

   Define a function *LessThan* such that LessThan m n = true iff m < n.

(c) (5 points)
Define the recursive fibonacci function *fib* such that fib 1 = 1, fib 2 = 2, and fib n = fib (n-1) + fib (n-2) for n > 2.

(d) (5 points)
Use your definition of fib from above to calculate fib 2. Do it step by step, showing all of your work. You may assume that fib 1 = 1 and fib 0 = 1 and that Plus 1 1 = 2 without showing all of the reduction steps. All other steps in the reduction should be shown.

7. (10 points) **Fixed Points**

We showed in class that every function f definable in the lambda calculus has a fixed point, i.e., there is a term a such that f(a) = a. In class we defined the function Succ as the successor function. I.e., Succ n = n+1. Needless to say, we don't expect the successor function to have a fixed point. Compute the fixed point of Succ. What can you say about it? In particular, why doesn't this contradict our expectations that the successor function on the natural numbers does not have a fixed point.