Problem Set 2

Due: 6 October 1995.

Problem 1. Here is the scheme body *maxtest* from the file `examples.scm` in the course directory containing the substitution model code.

```
(define (max2d f n)
  (if (zero? n)
      0
      (max (max1d (lambda (i) (f n i)) (- n 1))
           (max1d (lambda (j) (f j n)) n)
           (max2d f (- n 1)))))
(define (max1d h n)
  (define (iter max-so-far k)
    (if (zero? k)
        max-so-far
        (iter
         (max max-so-far (h k))
         (- k 1))))
  (iter 0 n))
(define (m n)
  (lambda (i j)
    (let ((2n (* 2 n)))
      (- 2n (+ i j)))))
(max2d (m 4) 4)
```

1(a) Add suffixes to the binding and bound occurrences of the variables in *maxtest* so it becomes an equivalent body satisfying the Variable Convention.

1(b) Check your answer by matching it against the result of applying the `enforce` procedure (in the file `sm-scope.scm`) to *maxtest*. Is your answer identical to the `enforce` result? If not, explain why it is correct anyway.

1(c) Double-check your answer by applying the procedure `clean-suffixes` to it. Does the result match the original *maxtest*?

1(d) Rename the bound variables in *maxtest* so that you use the smallest number of distinct identifiers.

1(e) How could you use the `clean-suffixes` procedure to aid in working out the previous subproblem and similar examples?

Problem 2.

2(a) Give an informal equational proof in the style of lecture on 9/27 that application of Church numerals corresponds to exponentiation. In other words, prove that the procedure `church-exp` from Handout 7, "Church Numerals and Y Operator" works as claimed.

2(b) Similarly prove that the composition procedure is associative:

`(compose (compose f g) h)` = `(compose f (compose g h))`.

2(c) Complete the definition

```
(define (church-add m n) <???>)
```

from Handout 7, and give an equation-style proof that your definition is correct.

Problem 3. Let A be any one of the Scheme term-rewriting rules P, C, I, L, G. We use the notation $M \xrightarrow[\mathrm{A}]{} N$ to indicate that scheme body M rewrites to become body N by a *single* application of rule A, and $M \xrightarrow[\mathrm{A}*]{} N$ to indicate any number, including zero, applications of rule A.

3(a) Show that if $M \xrightarrow[\mathrm{A}]{} M_1$ and also $M \xrightarrow[\mathrm{G}]{} M_2$, then there is an M_3 such that $M_1 \xrightarrow[\mathrm{G}*]{} M_3$ $M_2 \xrightarrow[\mathrm{A}]{} M_3$.

3(b) We can now conclude that it doesn't matter when garbage collections are performed in the course of rewriting a Scheme expression into a value. Give a precise statement of this fact (along the lines of the Uniqueness Theorem of the handout "A Term Rewriting Model for Scheme"), and outline its proof.

3(c) It follows from the previous subproblem that we get the "same" result as any given rewriting model evaluation, if we postpone all garbage collection steps until the end. Give an example showing that it is not possible to perform all the garbage collections at the beginning and still finish with the same value.