Homework 5

Please email your solutions to Rongdi Huang (rh1424@nyu.edu). Solutions to programming exercises must be submitted electronically as plain text files. No exotic formats, please!

The deadline for Homework 5 is October 10.

Problem 1  λ-Calculus (6 Points)
Using the definitions from the lecture, compute the normal forms of the following λ-calculus terms. Show each β-reduction.

(a) iszero(mult⌜0⌜2⌝)
(b) exp⌜2⌜2⌝
(c) Y exp⌜0⌝

Problem 2  Programming in Scheme (14 Points)
For the following problems, make sure your code runs under drracket. drracket is available on the CIMS machines and can be downloaded from http://racket-lang.org. Also, many Linux distributions provide appropriate packages.

(a) PLP, pp. 538-539: 10.6 b. The name of the function should be min. (4 Points)
(b) PLP, p. 539: 10.7 b. The name of the function should be filter. (5 Points)
(c) PLP, p. 539: 10.8. The name of the function should be permutations. (5 Points)

Problem 3  λ-Lists (5 Bonus Points)
Note: this exercise is optional.
Lists are an important data structure in Scheme programs. Scheme therefore provides an inbuilt list data type. However, from a theoretical point the list type is redundant because, just like numbers, lists can be encoded directly in the untyped λ-calculus. A list can be represented in the λ-calculus by its fold function. For example, the list ’(x y z) becomes a function that takes two arguments c and n and returns c x (c y (c z n)). What is the representation of the empty list ()? Write a λ-calculus term cons that takes an element h and a list t (that is a fold function) and returns a similar representation of the list formed by prepending h to t. Write isnil and head functions, each taking a list parameter. Finally, write a tail function for this representation of lists (this is quite a bit harder and requires a trick analogous to the one used to define pred on Church numerals). Transfer your λ-terms into function definitions in Scheme and write a function evallist that translates the fold representation of a list to the corresponding inbuilt representation in Scheme.