CS345 Fall 2012 - Homework Assignment # 2

Due Friday, September 21, 2012 by 5:00 pm. Use turnin to submit your homework (i.e., turnin –submit [MW use swati, TTh use benself] cs345_hw2 hw2.zip)

Put each solution in a separate file hw2.1.lisp, hw2.2.lisp, hw2.3.lisp, hw2.4.lisp, hw2.5.lisp, ec.lisp respectively and then put them all into a zip file.

1. [2 Points] In the answer to Problem 59 in DeLong's book, Howard gives the functions for primitive recursive exponentiation. Implement these functions in lisp in a manner similar to what's on page 11 of the PLAI1 class notes for primitive recursive addition (i.e., f_5). In the let statement for the solution show that $2^{10} = 1024$. (By the way, you should see that f_9 is multiplication).

2. [2 Points] On pages 158 and 159 in DeLong's book, Howard gives the functions for primitive recursive equality. Implement these functions in lisp in a manner similar to problem 1 above. In the let statement for the solutions build a list that shows that 11 = 11, 11 != 12, and 12 != 11. I.e., the let should return '(0 1 1).

3. [2 Points] In a manner similar to problem 2, implement the functions for primitive recursive "less than" and in the let statement for the solution build a list that shows that 11 < 12, 11 !< 11, and 12 ! < 11. I.e., the let should return '(0 1 1). Hint – see DeLong problem number 60.

4. [2 Points] Using just lambda, if, car, cdr, and cons, add a foldl function to the list of functions defined in the let statement an page 32 of the PLAI1 class notes. You can think of the foldl function as doing the following to the list '(1 2 3 4 5 6) if it is applied with the "+" function and v = 0:
   (((((1 + 2) + 3) + 4) + 5) + 6) + v = 21

I.e., (foldl + 0 List) evaluates to 21. Hint - study and understand the head function, foldl is somewhat similar to it.

5. [2 Points] Using just lambda, if, car, cdr, and cons, add a foldr function to the list of functions defined in the let statement an page 32 of the PLAI1 class notes. You can think of the foldr function as doing the following to the list '(1 2 3 4 5 6) if it is applied with the "+" function and v = 0:
   v + (1 + (2 + (3 + (4 + (5 + 6)))))

I.e., (foldr + 0 List) evaluates to 21.

Extra Credit [1 Point] Implement the map function in terms of foldr and show that (map (lambda (x) (+ x 5)) List)) evaluates to '(6 7 8 9 10 11).