MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Department of Electrical Engineering and Computer Science
6.001 Structure and Interpretation of Computer Programs
Spring, 2007

Recitation 5, Friday February 23

List (+ Recursion + Orders of Growth) Problems Dr. Kimberle Koile

Fill in the code for these recursive procedures. Assume recursive processes (not iterative).

1. This procedure returns the length (i.e., number of elements) in a list.

   (define (length lst)

   time = Θ( )

   space = Θ( )

   n is

)

2. This procedure returns the nth element of a list, where the first element index is 0.
   (define (list-ref lst n)

   time = Θ( )

   space = Θ( )

   n is

)

3. This procedure returns #t if obj is an element of a list; #f if it is not.
   (Hint: Use the procedure `equal?`.)

   (define (member? obj lst)

   time = Θ( )

   space = Θ( )

   n is

)

4. The procedure returns a new list that has exactly one instance of each element in the original list.
   (Hint: Use the procedure `member?`.) e.g., (remove-duplicates (list 1 2 1 2 3 4)) => (1 2 3 4)

   (define (remove-duplicates lst)

   time = Θ( )

   space = Θ( )

   n is

)