COMP 210, Spring 2002, Homework 8
Due Friday, March 22, 2002 at the start of class

Before you start the homework, you should remind yourself of our General Advice, Advice on Homeworks, and Grading Guidelines. All are available from the class web site (http://www.owlnet.rice.edu/~comp210).

For each program that you develop, be sure to write a contract, purpose, and header – unless the program is a lambda function. You do not need to write out templates for the programs in this homework assignment.

1. (1 point) Write down the contracts for **filter**, **map**, and **foldr**.
2. (2 points) Given the following definition for a point

   ```
   ;; a point is
   ;;    (make-point x y)
   ;; where x and y are numbers
   (define-struct point (x y)
   ```

   Use the Scheme functions **filter**, **length**, and **sqrt** to create a function **within-1** that consumes a list of point and produces a list of point. The list that it produces should contain exactly those points that are within a distance of 1 from the point (0,0).

   [That is, the square root of $(x-0)^2 + (y-0)^2$ is less than or equal to one.]

   Use **local** to hide any helper functions that you write.
3. (2 points) Write the following abstract function

   ```
   ;; somep: (alpha→boolean)  list of alpha → boolean
   ;; Purpose: takes a predicate function p and a list.  Returns true if
   ;;         p is true for some element of the list
   (define (somep  p  alist) …)
   ```

   Use local to avoid passing any invariant parameters.
4. (5 points) Consider the following two programs.

   ```
   (define (double-last alon)
     (cond [(empty? (rest alon)) (* 2 (first alon))]
           [else (double-last (rest alon))]))


   ;; The function max is built-in in DrScheme
   (define (max-of-list alon)
     (cond [(empty? (rest alon)) (first alon)]
           [else (max (first alon) (max-of-list (rest alon)))]))
   ```

   a. (3 points) Write an abstract program that captures both of these definitions.
   b. (2 point) Provide new definitions of double-last and max-of-list in terms of your abstract program.