Functional Programming

Notes about this assignment:

a. Implement your program in DrRacket. Click on "Language" and choose "Choose Language". Then select "Other Languages" and then in "Legacy Languages", select "R5RS" which is the Scheme language.

b. Write all your solutions in a single file with the extension ".scm" and submit it on Canvas.

1. Implement the following functions using the Scheme programming languages:

   (a) [15 points] `inc_n`: a higher-order function that takes an integer n as a parameter and returns an n-th increment function, which increments its parameter by n. Thus, in Scheme syntax, `((inc_n 3) 2)` and `((inc_n -2) 3)` return `5` and `1`, respectively.

   (b) [15 points] `len`: a tail-recursive function that takes a list as a parameters and returns its length. For example, `(len '(2 1))` returns `2`.

   (c) [15 points] `maxmin`: a function that computes and returns the maximum and minimum of a list of integers. For example, `(maxmin '(4 2 -1 10))` should return `(10 -1)`.

   (d) [15 points] `mem`: a Boolean function that takes two parameters (the first one has any data type but the second one will be a list), and returns true/false if the first data is/is not found in the list. For example, `(mem '(1) '(1 4 -2))` returns `#f`.

   (e) [10 points] `ins`: a function that takes two parameters (similar to `mem`), and inserts the data in the list if it is *not* already there. For example, `(ins 5 '(2 10 -3))` returns `(5 2 10 -3)`.
   Hint: use `mem` *in your function.*

   (f) [15 points] `numT`: a function that takes two parameters, a Boolean function and a list, calls the Boolean function for each element in the list and returns the number of times that the function returned true. For example, `(numT number? '(1 -5 -4 (2 1) 7))` returns `4`.

   (g) [15 points] `moreT`: a function that takes three parameters, a Boolean function and two lists, and outputs which list returns more *true* values. In other words, it runs the Boolean function for each value in each list, and counts the number of times that each list of values returned true. It should return 1 (or 2) if the first (or second) list returns more true values. If both lists return true the same number of times, your function should return 0.
   For example, `(moreT negative? '(8 -4 3 8) '(7 -3 -2 1 -5))` should return `2`. As another example, `(moreT even? '(8 -4 3 8) '(6 3 2 1 -4))` should return `0`.
   Hint: use `numT` *in your function.*