# CIS 343

## Functional Programming Assignment

Complete the following functions, using MIT-Scheme. If a function exists which already does what is asked, do not use it. Rather, create your own. You may refer to <https://www.gnu.org/software/mit-scheme/documentation/mit-scheme-ref/> for built-in function documentation.

1. Define the two values, pi and e to at least 5 decimal places.
2. Write a function called circle-specs. This function will take a single parameter radius and will return a list that provides the circumference and area). So, (circle-specs 10) will return something along the lines of (62.8318 314.159). One caveat; you must use the let function to define the pi \* r part so it may be used in both calculations.
3. Define a function (logn n val) that calculates the log of val given base n. Use this to create a new function (log2 val) which provides the log base 2 of a value. A hint – MIT-Scheme has the log function that returns the natural (log base *e*) log, so you will need to convert the bases.
4. The map function takes a function and one or more lists and applies the function to each element in each (the lists must be the same length). Write a function that will take two lists and add them together, producing a new list. For example, if we had (1 2 3 4) and (5 6 7 8) we should end with (6 8 10 12). No need to define this function unless you wish.
5. Functional languages have reduce functions as well. Where map applies functions to lists, reduce applies a function to combine all elements of a list. For instance, we may wish to add each element of a list together, so (1 2 3 4) would result in 10. Define a new function called (dot-product vector1 vector2) that will take two arbitrary length vectors and calculate the dot product. For (1 2 3 4) (5 6 7 8) the return should be the scalar (single number) value 70. Use the reduce-right function as well as the function you created in step 3.
6. Write a function called fib that takes a single parameter and returns the Fibonacci number at that position. For instance, (fib 5) will return the value 5, and (fib 10) will return the value 55.
7. Create a function called (create-list start end) that creates a list of numbers from start to end. (create-list 1 10) will return the list (1 2 3 4 5 6 7 8 9 10).
8. Create a new function called fib-list. It will take one parameter and using the above functions will return a list of all Fibonacci numbers from 1 to the value of the parameter. Therefore, a call of (fib-list 10) will return (1 1 2 3 5 8 13 21 34 55). This is a very simple function if you compose it from the functions you have created and learned about above.
9. Write a function called nth, that is passed a list and a number and returns the nth element of the list. Lists are zero indexed.
10. Write a function called remainder that is given two numbers a and b and returns the remainder from dividing a by b. Complete the task by continually subtracting b from a.