Programming Exercise 03
Scheme Programming 1

OBJECTIVE

To be able to write a program following the functional programming paradigm using Scheme.

INSTRUCTIONS

1. You are to work on this activity individually or by pair.
2. You are to write a code for a program that follows the details in the PROGRAM SPECIFICATION section. There should only be one rkt file for this PE. This single rkt file should contain all the procedures/functions you will create.
3. You are to develop your program following the functional programming approach. No global identifiers are allowed.
4. You only need to submit the source code file you have created. Name your source code file using your surname followed by the PE # similar to this example: Rizal_PE03 (for individual) or Bonifacio_Rizal_PE03 (for pair/group).

PROGRAM SPECIFICATION

Write the required named procedures/functions being described.

A. (**Factorial *n***) - returns the factorial of ***n***

B. (**T-Ice *n***) – displays the values from 1 to ***n*** in the following manner:
   - Display T if the current number (from 1 to *n*) is divisible by 2
   - Display ICE if the current number (from 1 to *n*) is divisible by 3
   - Display T-ICE if the current number (from 1 to *n*) is divisible by both 2 and 3
   - Otherwise, display the current number (from 1 to *n*) itself

   Example: a call to the procedure for the value *n* = 8 should display:
   1 T ICE T 5 T-ICE 7 T

C. Sum of Primes: whenever the function (Sumprimes ***n***) is called, it displays the sum of all the prime numbers from 1 to ***n***. Note: you can assume positive integer values for ***n***.

D. Define a function **perform-op** that accepts three arguments ***m***, ***n***, and ***opt*** (***m*** and ***n*** are two numbers or integers, opt is either a symbol or a character). Given a valid operator (+, -, /, *, %) and two integers as operands, the function returns the result of the operation.

E. Define a function **calc-distance** that accepts four integers (***x1***, ***x2***, ***y1***, ***y2***) and returns the distance between the points (***x1***, ***y1***) and (***x2***, ***y2***). Use the built-in functions **sqrt** (square-root of a number) and **abs** (absolute value of a number). *Note that output similar to **#i2.8**… is okay since it may be a result of the **sqrt** procedure*.

F. Define a function **count-factors** that accepts two integers ***m*** and ***n***, and returns the number of factor ***m*** in ***n***. If no factor ***m*** exists in ***n***, display "**None**".

   To describe: n = 48, m = 4: output is 2 since 48 = 4*4*3

Note: For items A to E, you can define some helper functions to achieve the purpose of each. For item F, there should only be a single function – no helper function(s) should be defined. Also, you are not allowed to define and use identifiers.