Tutorial 4 : Functions

Exercise 1:

1. Provide the Algorithm for a function called *digits_sum* that reads an integer and returns the sum of its digits.
2. Provide the Algorithm for a program that uses *digits_sum* to calculate the sum of the digits of a number entered by the user. The sum, in turn, will be converted to the sum of its digits. And so on, until the sum is only one digit long.
   Example : 75 12 3.
3. Write the correspondent C code.

Exercise 2:

1. Give the algorithm for a function *power* that calculates and returns the nth power (positive integer) of a real number p. n and p are provided as parameters.
2. Modify the function *power* to account for a negative n integer.

Exercise 3:

The natural logarithm of the number (1 + x) can be simulated by the following series:

$$Ln(1+x) = x - x^2/2 + x^3/3 - x^4/4 + x^5/5 - \ldots$$

Give the algorithm for a function that calculates the series to a certain rank N.

Exercise 4:

A factorion is a positive integer that is equal to the sum of the factorials of its digits.
Examples:

145 = 1! + 4! + 5!
40585 = 4! + 0! + 5! + 8! + 5!

1. Provide the algorithm for a function that calculates the factorial of an integer given as a parameter.
2. Provide the algorithm for a function that checks if a positive integer given as a parameter is a factorion.

Exercise 5:

Give the algorithm for a procedure that determines and displays whether a positive integer given as a parameter is an abundant, perfect, or deficient number.

A number is abundant when the sum of its divisors is greater than twice that number.
A number is perfect when the sum of its divisors is equal to twice that number.
A number is deficient when the sum of its divisors is less than twice that number.

Example :

6 is a perfect number : 1 + 2 + 3 + 6 = 2*6
8 is a deficient number : 1 + 2 + 4 + 8 < 2*8
12 is an abundant number : 1 + 2 + 3 + 4 + 6 +12 > 2*12