TUTORIAL N°1: SUB-PROGRAMS
Function and procedures

Exercise 1: Write a function ***"power"*** that calculates a^b = a x a x a x …x a (b times); a and b are positive integers.

Exercise 2: Knowing that a **prime** number is a number that does not accept any divider except one and itself.

1. Write a function **"prime"** that indicates whether an integer N is prime or not.
2. Construct the algorithm that gives us the first N prime numbers.
3. Write a procedure **NearPrem(N)** which returns two integers: the first is the prime number closest to N and greater than N. The second is the prime number closest to N and less than N.

Exercise 3: Write a procedure ***Exchange1*** that reverses the content of two integer passed as arguments

Exercise 4: Write the following procedures

1. Procedure ***Exchange2*** with 2 reals a and b which optionally inverts a and b, so that the output state is a <= b. Write a main Algorithm.
2. Procedure ***Exchange3*** with 3 reals a, b and c which calls Exchange2. The output state is a <= b<= c. Write the main Algorithm.

Exercise 5: from an integer N, we would like to obtain two other numbers N1 and N2. The first (N1) will be made up of the even numbers of N and the second (N2) of the odd numbers

Examples :	N = 25461327	N1 = 2462	N2 = 5137
	N = 42613786	N1= 42686	N2 = 137
	N = 240682	N1 = 240682	N2 = 0

1. Write an algorithm that returns the numbers N1 and N2 from a number N

N.B: solution must include at least one function and one procedure.

Exercise 6: Let A be a time in seconds..

1. Write a ***procedure*** that converts A to hours, minutes and seconds.

Exercise 7: Given two numbers N1 and N2.

1. Write a ***function*** that allows you to merge the digits of the two integers N1 and N2 alternately.

Examples: N1 = 381 N2 = 946 N= 398416

N.B: The solution must have at least two functions.

Exercise 8: (Examen rattrapage Init.Algo S1 2015/2016)

1. Write a procedure **Insert_Chiff (c, p, N)** which allows you to insert a digit **c** into the number **N** at position **p** (the result is N itself).

   Example: Insert_Chiff(5, 3, N) with N = 14721 gives the result N = 147521.

Exercise 9: Write a function to convert a number N written in a base x (x <= 10) to base10.

Exemple : (7643)8 = (4003)10

1. Generalize the previous solution to convert a number written in a base x to a number written in a base y (x<= 10, y <= 10)

Exercise 10 : (Examen rattrapage Init.Algo S1 2013/2014)

1. Write a function **SumDiv(n)** which returns the sum of the proper divisors (except the number itself) of an integer N.
2. Write an algorithm that displays whether an integer X is abundant, perfect or deficient using the **SumDiv(n)** function.

When the sum of the proper divisors of a number is greater, the number is said to be abundant. When the sum is equal to it, the number is said to be perfect. When the sum is less, it is said to be deficient.

Exercise 10: Two strictly positive numbers **m** and **n** are called friend numbers if and only if the sum of the divisors of **m** except itself is equal to **n** and the sum of the divisors of **n** except itself is equal to **m**.

Example: *220 and 284 are two friend numbers.*

1. Write an algorithm that displays whether two given natural numbers **m** and **n** are friends or not.

N.B: The solution must include at least one function.

Exercise 12:

1. Write a function to check whether a word is palindrome or not.

   Example: palindrome words: RADAR, LEVEL, ROTATOR, ….

2. Write an algorithm to display all palindrome words in a given text

Exercise 13:

1. Write an algorithm that allows you to enter two non-empty words Word1 and Word1 then determine if Word1 is an anagram of Word1.

An anagram is a word obtained by transposing the letters of another word.

Exemple : silent is an anagram listen.

N.B : The solution must include at least one function.

Exercise 14:

Let **T** be a one-dimensional array of **n** integer values (with n <1000)

1. Write a function ***Occurrences (T, n, Val)*** which returns the number of occurrences (repetitions) of the value **Val** in the array **T**.
2. Using the previous function, Write an algorithm which returns the 1st non-repeating element in the array T

Example: The first non-repeating element is **8**

2	**2**	**3**	**8**	**6**	**6**	**7**	**3**	**3**	**2**	**9**	**9**

Exercise 15: (Examen PSD S2 2013/2014)

Let T be a one-dimensional array of **N** integer values (with n <1000)

1. Write a function **Tab_Rech (T, N, val)** which allows you to search for an integer value **val** in the array **T** and returns its index in the array T if it exists (returns the index of the first occurrence) otherwise, returns a negative value.
2. Write a procedure **Tab_without_double (T1, N, T2)**, using the previous function, which allows you to construct an array **T2** which contains only one occurrence of each number in **T1** of **N** elements.