4TPM205U: Array Algorithms : Sheet 1

Iterate a given interval with the loop `for`

TP

In all the functions of the sheet, we implement the iterations utilizing the loop `for`.
Subsequently, each time a function calculates a result, it has to return it.

Exercice 1 – Let `a` and `b` be two integers, write a function `affichePairs(a,b)` which prints all the even numbers in the *closed* interval $[a, b]$.
Can you write the function using maximum one `if` instruction before the loop ? (maximum implies that we might as well be able to do without using any. ...)

Exercice 2 – Let `n` be a natural number, write a function `puissance(x,n)` which calculates x^n.

Exercice 3 – Write and execute the function `fibonacci(n)` defined in the sheet 1 of TD.
The function call `fibonacci(9)` must return 55.

Exercice 4 – Write a function `afficheTableMult(n)` which prints a multiplication table of size n.
For example if `n` is 6, it should display :

```
 1   2   3   4   5   6
 2   4   6   8  10  12
 3   6   9  12  15  18
 4   8  12  16  20  24
 5  10  15  20  25  30
 6  12  18  24  30  36
```

Exercice 5 – Write and execute the function `sommeFactorielles(n)` defined in Sheet 1 of TD.
Test both the versions.

Exercice 6 – How do we calculate an approximation of $\sqrt{2}$? A simple method is to use the following sequence u_n :

$$\begin{cases} u_0 & = 2 \\ u_{n+1} & = \frac{u_n}{2} + \frac{1}{u_n} \end{cases}$$

With this method, the error between $\sqrt{2}$ and u_n reduces every iteration of the calculation.

1. Write a function `suite(n)` which calculates and returns the term u_n.

2. Use a loop to display the first 10 terms of the sequence. An approximated value of $\sqrt{2}$ is 1.414213562373095048[...]. It should be noted that the sequence converges towards this value, but without being able to reach it : the number of digits in the computer's value are limited !
3. More generally, to calculate $\sqrt{x}$, we can use the sequence

$$\begin{cases} u_0 & = x \\ u_{n+1} & = \frac{1}{2}(u_n + \frac{x}{u_n}) \end{cases}$$

where u_0 is a first rough estimate of $\sqrt{x}$.
Write a function `sqrt(x)` which returns an approximation of $\sqrt{x}$ by calculating u_{10} while printing the intermediate values u_i. Test with $x = \{2, 3, 4, 10\}$ and $n = 10$.

Exercice 7 – More difficult exercice, to be attempted in the end.

We are looking for a number of 9 non-zero digits, such that :
- the number formed by the first digit is divisible par 1
- the number formed by the first two digits is divisible by 2
- the number formed by the first three digits is divisible by 3
- the number formed by the first four digits is divisible by 4
- the number formed by the first five digits is divisible by 5
- the number formed by the first six digits is divisible by 6
- the number formed by the first seven digits is divisible by 7
- the number formed by the first eight digits is divisible by 8
- the number formed by the first nine digits is divisible by 9

First digit is the left most digit of the number.

There are many such numbers and the least of them is 123252561.

Without using any other iteration except the *for* loop (and trying to minimize the number of iterations), can we write a function (iterative !) that displays (and saves) all the numbers with the property described above ?
(Suggestion : start by trying to write a function which looks for a number with three digits satisfying the first three conditions.)