

TP n°2

La récursivité

1° - Une somme récursive

Exercice : La somme

Écrire un programme qui demande à l'utilisateur de rentrer un nombre, et qui affiche la somme de tous les nombres avant lui jusqu'à 0 plus celui-ci.

Cela permet de calculer :

$$1 + 2 + 3 + \dots + n$$

Exemple de résultat :

```
Saisir valeur de n : 5
La somme pour n=5 est de 15
```

1° - Réalisez l'algorithme et en déduisez la complexité.

2° - Avant de lancer votre algo sur la valeur 100. Trouvez la solution de tête.

(chut... si vous la trouvez, ne l'ébruitez pas... laissez les autres trouver par eux-même... ;)).

2° - La suite de Fibonacci

Exercice 1 : Nombre de Fibonacci avec récursivité

Ecrire un programme qui demande à l'utilisateur d'entrer un nombre n puis calcule et affiche le nombre de Fibonacci associé F_n :

- $F_0 = 0$
- $F_1 = 1$
- $F_n = F_{n-1} + F_{n-2}$

Les deux premiers éléments de la suite sont données soit F_0 et F_1 avec les valeurs de 0 et de 1. Les nombres suivants de la suite sont calculés en faisant la somme des deux précédents.

Rappel de la suite de Fibonacci :

n	0	1	2	3	4	5	6	7	8	9	10	11	12	...
F_n	0	1	1	2	3	5	8	13	21	34	55	89	144	...

1° - Réalisez l'algorithme avec l'appel à une méthode récursive.

Exemple :

```
Saisir un nombre : 10
Fibo à F10 = 55
```

2° - Testez votre programme avec des valeurs comme 10, 20, 30 ou 40 (pas vraiment plus...).

3° - Dessinez sur papier l'arbre d'appels à la méthode récursive pour un $n = 7$ par exemple. Déduire la complexité de cette version.

4° - Comprenez pourquoi le programme devient très lent ?

Exercice 2 : Nombre de Fibonacci, version itérative.

1° - Trouvez une nouvelle solution sans utiliser la récursivité.

2° - Déduisez la complexité de cette version et comparez-la à la précédente.

Faire la conclusion de tout ça.