

# Algorithmique 1

## TD 1

### Exercice 1.1

On considère la fonction suivante :

```
fonction mystere(ref T : tableau{1..N} d'entier , val N : entier):vide;
var i, j : entier;
debut
  i = 1;
  j = N;
  tant que i < j faire
    si T[i] == 0 alors
      i = i+1
    sinon
      echanger(T[i], T[j]);
      j = j-1
    fsi
  ftq
fin;
```

1. Que fait-elle en supposant que le tableau T ne contient que des 0 et des 1 ?
2. Modifier l'algorithme de façon à l'appliquer à des tableaux contenant au plus trois valeurs différentes : par exemple 0, 1 et 2 (rappel : chaque élément de T doit être examiné au maximum une fois).  
Peut-on l'étendre aisément à K valeurs différentes ?
3. Donner une version récursive de l'algorithme général.
4. Ecrire une fonction séparant un tableau d'entiers T[1..N] contenant une valeur X en 2 zones : T[1..I-1] contenant les entrées strictement inférieures à X, T[I..N] les entrées supérieures ou égales à X, de façon que X se trouve en position I à la fin du traitement.  
Etudier l'occupation en mémoire.  
Donner le principe d'un algorithme de tri utilisant cette dernière fonction.

### Exercice 1.2

**Suite de Fibonacci.** On rappelle que la suite de Fibonacci, 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ... s'obtient de la façon suivante : à partir du rang 3, chaque terme est la somme des deux précédents.

Pour calculer le  $n$ -ième terme de la suite donner une fonction :

- récursive
- récursive terminale
- itérative

Calculer la complexité du calcul des nombres de Fibonacci par les algorithmes récursif et itératif précédents. Il est utile de savoir que les nombres de Fibonacci sont étroitement liés au *nombre d'or*  $\phi$  et à son conjugué  $\bar{\phi}$ , qui sont donnés par les formules suivantes :

$$\phi = \frac{1 + \sqrt{5}}{2} = 1,61803\dots$$

et

$$\bar{\phi} = \frac{1 - \sqrt{5}}{2} = -0,61803\dots$$

Plus précisément, si on appelle  $F(n)$  le  $n$ -ième nombre de Fibonacci ( $n \geq 0$ ), on a

$$F(n) = \frac{\phi^n - \bar{\phi}^n}{\sqrt{5}}$$

(ce qu'on peut prouver par récurrence).

### Exercice 1.3

Ecrire un algorithme calculant le  $n$ -ième nombre de Fibonacci en temps  $\log_2(n)$ .

Indications :

1. Construire le vecteur  $F\vec{I}\vec{B}O(n) = \begin{pmatrix} F(n) \\ F(n-1) \end{pmatrix}$  et exprimer  $F\vec{I}\vec{B}O(n)$  en fonction de  $F\vec{I}\vec{B}O(n-1)$  à l'aide d'une matrice de dimension  $2 \times 2$  à déterminer.
2. Calculer les puissances d'une matrice en utilisant la méthode alexandrine.