

Université de Bejaia.  
Faculté des Sciences Exactes.  
Département Informatique

**Travaux Dirigés**  
**Programmation Avancée**

**Master 1 en Informatique**  
**Options RN : RS & SIA**

*Prof. BOUALLOUCHE Louiza*

# Série n°1 de TD PA

**Exercice 1.** Soit la procédure suivante.

**Procédure** Produit (**Var** C : Matrice ; A, B : Matrice ; n, m, p : entier);

**Var** i, j, k: entier;

**Début**

```
Pour i allant de 1 à n Faire  
  Pour j allant de 1 à m Faire  
    C[i,j] ← 0 ;  
    Pour k allant de 1 à p Faire  
      C[i,j] ← C[i,j] + A[i,k] × B[k,j] ;  
    Fin pour  
  Fin pour  
Fin pour
```

**Fin**

1. Que calcule t-elle?
2. Calculer sa complexité.

**Exercice 2.** On considère une matrice triangulaire inférieure *MAT* de *NXN* éléments.

1. Ecrire une fonction qui retourne la somme des éléments de *MAT*.
2. Evaluer la complexité de la fonction puis donner son ordre de grandeur en  $O(\cdot)$ .
3. Donner sa complexité en nombre d'additions.

**Exercice 3.** On donne en entrée un réel  $x$  et un entier  $n$ .

1. Ecrire une fonction efficace qui calcule  $Som = 1+x + 2x^2 + \dots + nx^n$
2. Donner la complexité de la fonction en nombre de multiplications et un ordre de grandeur en  $O(\cdot)$ .
3. Peut on faire mieux? si oui comment?

**Exercice 4.**

1. Ecrire une fonction itérative qui retourne la factorielle de  $N$  ( $N!$ )
2. Donner la complexité de la fonction en nombre de multiplications  $O(\cdot)$  Peut on faire mieux?
3. Ecrire une fonction itérative efficace qui retourne  $C_N^P$

$$C_N^P = \frac{N!}{P!(N-P)!}$$

4. Donner la complexité de cette fonction en nombre de multiplications et un ordre de grandeur en  $O(\cdot)$ .

### Exercice 5.

- 1- Elaborer un algorithme de recherche du maximum dans un ensemble à  $n$  éléments (on dispose en tout et pour tout d'une fonction de comparaison).
- 2- Quelle est la complexité de votre algorithme en nombre de comparaisons ?
- 3- Montrer qu'il est optimal.

### Exercice 6.

- 1- Proposer un algorithme simple de recherche du deuxième plus grand élément.
- 2- Quel est sa complexité en nombre de comparaisons ?
- 3- Réécrire votre algorithme de recherche du maximum sous forme d'un tournoi. Une figure explicative sera largement suffisante. Donner sa complexité.
- 4- Dans combien de comparaisons, le deuxième plus grand élément de l'ensemble a-t-il été trouvé être le plus petit des deux éléments comparés ?
- 5- Proposer un nouvel algorithme de recherche du deuxième plus grand élément.
- 6- Quelle est sa complexité en nombre de comparaisons ?

**NB.** On suppose que l'ensemble considéré dans cet exercice ne contient pas deux fois la même valeur.

### Exercice 7.

- 1- Proposez un algorithme naïf de recherche du maximum et du minimum d'un ensemble de  $n$  éléments.
- 2- Quelle est sa complexité en nombre de comparaisons ?
- 3- Proposer un algorithme plus efficace.
- 4- Quelle est sa complexité en nombre de comparaisons ?

**NB.** On suppose que l'ensemble considéré dans cet exercice ne contient pas deux fois la même valeur.

**Exercice 8.** On dispose d'une liste linéaire chaînée, une liste doublement chaînée et un tableau de taille  $n$  pour chaque structure. Selon que les structures soient triées ou pas, compléter le tableau suivant par l'ordre de grandeur de chaque opération.

	LLC non triée	LLC triée	LDC non triée	LDC triée	Tableau non trié	Tableau trié
Recherche (L,x)						
Insertion (L,x)						
Suppression (L,x)						
Succession (L,x)						
Prédécesseur (L,x)						
Minimum (L)						
Maximum (L)						

**LLC** : Liste Linéaire Chaînée ; **LDC** : Liste Doublement Chaînée. **x** : Pointeur ou indice. Le tri est dans l'ordre croissant.