Travaux Dirigés

Exercice 1 :

Déterminez la complexité des algorithmes (par rapport au nombre d'itérations effectuées).

1-

```
1 Procédure algo1 ( m : Entier ; n : Entier )
2 Variable i , j : Entier
3 Début
4       i <- 1
5       j <- 1
6       TantQue ( i <= m ) Et ( j <= n ) Faire
7               i <- i + 1
8               j <- j + 1
9       FinTantQue
10      Fin
```

2- Refaire la question 1 en remplaçant le 'Et' de la ligne 6 avec un 'Ou'.

3-

```
1 Procédure algo2 ( m : Entier ; n : Entier )
2 Variable i , j : Entier
3 Début
4       i <- 1
5       j <- 1
6       TantQue j <= n Faire
7               Si ( i <= m ) Alors
8                       i <- i + 1
9               Sinon
10                      j <- j + 1
11               FinSi
12      FinTantQue
```

4- Refaire la question 3 en ajoutant dans la ligne 10 après incrémentation de 'j', l'instruction ' i<- 1' (réinitialisation de i à 1).

Exercice 2 :

1- Montrez que $n^2 \in O(10^{-4}n^3)$.
2- Montrez que $25n^4 - 10n^3 + 22n^2 \in O(n^4)$.
3- Montrez que que $2^{n+10} \in O(2^n)$.
4- Donnez les relations d'inclusion entre les ensembles suivants : O(n log n), $O(2^n)$, O(log n), O(1), $O(n^2)$, $O(n^3)$ et O(n). Que pouvez-vous conclure en termes de complexité des algorithmes.

Exercice 3 :

Soient f, g : N → N.
1- Prouvez que si $f \in O(g)$ alors $(f + g) \in O(g)$.
2- Prouvez que O(f + g) = O(max(f, g)).
Soient f, g, S, T : N → N. On suppose $S \in O(f)$ et $T \in O(g)$.
3- Prouvez que $S\,T \in O(fg)$. Que pouvez-vous conclure en termes de complexités des algorithmes.
Soient f, g, S, T : N → N. On suppose $S \in O(f)$ et $T \in O(g)$.
4- Prouvez que si $f \in O(g)$ alors $(S + T) \in O(g)$. Que pouvez-vous conclure en termes de complexités des algorithmes.

Exercice 4 :

Considérer le tri de n éléments stocker dans un tableau A en trouvant le plus petit élément de A puis en l'échangeant avec l'élément A[1]. Puis trouve le deuxième plus petit et l'échange avec A[2], et ainsi de suite pour les n-1 éléments restant de A.
Ecrire un pseudocode pour cet algorithme.
Pourquoi devra-t-il s'exécuter pour seulement les premiers n-1 élément et non pour tous les éléments n de A.
Donner le pire temps et le meilleur temps d'exécution en notation O.

Exercice 5 :

Modifier l'algorithme de tri par insertion pour trier les éléments du tableau de façon descendante.

Exercice 6 :

Soit la forme récursive de T(n) suivante:

$$T(n) = \begin{cases} 2 & \text{if } n = 2, \\ 2T(n/2) + n & \text{if } n = 2^k, \text{for } k > 1 \end{cases}$$

Prouver que la forme générale pour tout n est $T(n) = n.\log_2(n)$

Exercice 7 :

Appliquer le tri par fusion sur le tableau représenté par les valeurs ci-dessous :
16 - 11 - 9 - 10 - 5 - 6 - 8 - 1 - 2 – 4
Combien de divisions et de fusions seront effectuées.

Exercice 8 : Tri à bulles

On veut trier un tableau d'entiers.
Le principe du tri bulle est de parcourir le tableau du début à la fin en déplaçant la plus grande valeur rencontrée au fur et à mesure que l'on avance. Les premières étapes du tri bulle sont illustrées sur l'exemple suivant (les cases entre crochets sont les cases qui sont comparées, elles sont permutées quand la plus grande valeur est à gauche) :

[2 4]3 5 0 1
2**[4 3]**5 0 1
2 3**[4 5]**0 1
2 3 4**[5 0]**1
2 3 4 0**[5 1]**
2 3 4 0 1 5

Lorsque l'on arrive au bout du tableau, on recommence depuis le début...
1. Réalisez à la main le deuxième parcours du tableau.
2. Combien de parcours du tableau doit-on effectuer pour être sûrs que le tableau est trié ? (prouvez que le tableau est bien trié à la fin)
3. Écrivez la fonction tri_bulle(tab) qui trie un tableau par la méthode du tri bulle.
4. Sur un tableau de n cases, combien faut-il effectuer de comparaisons pour trier le tableau ?