UNIVERSITE IBN TOFAIL
Faculté des sciences
Département d'Informatique
Kenitra

Année: 2018/2019
Filières : SMI
Semestre : 3

Algorithmique II
Examen de rattrapage
Durée : 1h 30mn

Exercice 1 : (Sur 6 points)

On considère un tableau d'entiers T[1..n]. Ecrire une procédure *PlusLongSousTabCroissant (T : Entier[1..n])* qui détermine le premier sous-tableau de T de la plus grande taille constitué d'une suite croissante d'entiers. La procédure affichera les indices du début et de fin de ce sous-tableau de T. Autrement dit, si i et j sont respectivement les indices du début et de fin du sous-tableau trouvé alors : T[i]<=T[i+1]<=…<=T[j]. De plus le tableau T ne contient pas de suite croissante de taille strictement supérieur à (j - i+1).

Exercice 2 : (Sur 7 points)

On considère A[1..m, 1..m] un tableau d'entiers de dimension (m,m) . Soit la fonction Mystere(A : Entier[1..m, 1..m]) donnée par

```
Fonction Mystere( A : Entier[1..m, 1..m]) : Booleen
Var     i, j, k : Entier
Début
        Pour (i ← 1 à m-1) Faire
                Pour  (j←i+1  à m) Faire
                        k ← 1
                        Tant que (k<=m Et A[i, k] = A[j, k]) Faire
                                Si (k = m) Alors
                                        Retourner VRAI
                                Sinon
                                        k ←k + 1
                                Fin Si
                        Fin Tant que
                Fin Pour
        Fin Pour
        Retourner FAUX
Fin
```

1. **Quel est le but de la fonction Mystere ?**
2. **Calculer C1(m) la complexité temporelle dans les pires des cas de la fonction Mystere.**
3. **Calculer C2(m) la complexité temporelle dans les meilleurs des cas de la fonction Mystere.**

Exercice 3 : (Sur 7 points)

Le trie stupide est l'algorithme de trie le plus simple à comprendre et à programmer. Il fonctionne de la manière suivante :
Soit Tab[1..n] un tableau d'entiers à trier par ordre croissant en utilisant cet algorithme. On procède de la manière suivante :

a. **Parcourir le tableau Tab du début jusqu'à rencontrer deux éléments successives qui ne sont pas dans l'ordre.**
b. **Permuter les deux éléments trouvés.**
c. **Si les deux éléments trouvés ne sont pas à la fin du tableau, aller vers l'étape 1.**
d. **Sinon arrêter le processus, le tableau est trié**

1. **Ecrire en pseudo code l'algorithme ci-dessus**
2. **L'ordre de grandeur de la complexité temporelle <u>dans les meilleurs des cas</u> de cet algorithme est $\theta(n)$. Comment sont les éléments du tableau Tab dans ce cas-là ?**
3. **L'ordre de grandeur de la complexité temporelle <u>dans les pires des cas</u> de cet algorithme est $\theta(n^3)$. Comment sont les éléments du tableau Tab dans ce cas-là ?**