ALGORITHME II

Contrôle Finale + Rattrapage

2016 - 2017

SMIA

STUDIES

UNIVERSITE MOHAMMED V

Faculté des Sciences – RABAT

Département d'Informatique

Année 2016- 2017

SMI3 – ALGO.II

Evaluation. Durée 1h30

(N.B. La complexité doit être justifiée pour chaque question)

EXERCICE 1. (5 pts)

Deux entiers sont congrus modulo n s'ils ont le même reste dans la division euclidienne par n.

Soit T un tableau de n entiers. Ecrire un algorithme pour regrouper tous les entiers de T qui sont congrus à 0 modulo 3 au début du tableau T, suivis des éléments congrus à 1 et les entiers qui sont congrus à 2 modulo 3 seront placés à la fin du tableau T. L'algorithme doit utiliser une seule boucle. Quelle est sa complexité ?

EXERCICE 2. (10 pts : 2,3,2,3)

Soit T un tableau à n entiers et soit x un entier.

1) Ecrire un algorithme récursif qui retourne la position de la dernière occurrence de x dans le tableau T, si l'élément x est dans T. Il retourne 0 sinon. Quelle est sa complexité ?
2) Ecrire un algorithme récursif utilisant le principe « diviser pour programmer » pour chercher si x est dans le tableau T. Il retourne 'vrai' si x est dans T, 'faux' sinon. Quelle est sa complexité ?
3) Ecrire un algorithme itératif qui cherche s'il existe deux entiers, dans T, dont la somme est x. Il retourne deux indices i et j si x = T[i] + [j] ou (0,0) dans le cas contraire. Quelle est sa complexité ? (Ce problème peut être spécifié par : $\exists i \, \exists j \, (1 \leq i < j \leq n) \wedge (x = T[i]+T[j])$).
4) On suppose que le tableau T est trié dans l'ordre croissant et soit l'algorithme $Rech\_dicho(T, inf, \mathrm{sup}, \mathrm{e})$ (vu en cours) qui fait une recherche dichotomique de l'élément e dans T[inf..sup] et qui retourne un indice k si e = T[k], 0 sinon. Réécrire l'algorithme précédent (algo. écrit en 3)) en faisant des appels à l'algorithme $Rech\_dicho$. Quelle est sa complexité ?

EXERCICE 3. (5 pts)

On considère l'algorithme suivant, m et n sont des entiers strictement positifs tels que $n \leq 2m$:

```
Données : m, n

Résultat : a

début

    k := 0 ;  a := n ;

    Tantque  a ≤ m  faire

            k := k + 1 ;

            a := 2 * a ;

    fin tantque

fin.
```

1) Prouver l'algorithme ci- dessus.
2) Déduire, de la post-condition, que le nombre d'itérations est $E(\log_2 \frac{m}{n} + 1)$.

UNIVERSITE MOHAMMED V **Année 2016- 2017**

Faculté des Sciences – RABAT **SMI3 – ALGO.II**

Département d'Informatique **Rattrapage. Durée 1h30**

EXERCICE II.

Soit T un tableau de n entiers.

1) Ecrire un algorithme récursif qui calcule la somme modulo 3 des éléments de T.

(La valeur retournée par l'algorithme est 0, 1 ou 2 selon le reste de la division de la somme par 3)

2) Utiliser la méthode 'Diviser Pour Résoudre' pour calculer la somme modulo 3 des éléments de T.

EXERCICE III.

1) Ecrire un algorithme itératif qui remplit un tableau T de la façon suivante : un 1, suivi de deux 2, suivi de trois 3, etc. jusqu'à un entier n fixé à l'avance. Par exemple, pour n = 4, le tableau T sera : [1,2,2,3,3,3,4,4,4,4]. Calculer le nombre d'éléments de T en fonction de n . Quelle est la complexite de l'algorithme ?

⚠ 4pts 2) Donner un algorithme récursif, équivalent au précédant, avec un en-tête : **remplissageR(T, k, n)** où k est l'indice de début d'un bloc d'un même entier, dans le tableau T.

EXERCICE IV.

Prouver l'algorithme suivant :

Donnée n ;

Résultat f ;

début

i := 0 ; f := 1 ;

tantque i < n faire

i := i + 1 ;

*f := f * i ;*

ftantque

fin.