Université Mohamed Boudiaf – M'sila
Faculté des Mathématiques et de l'Informatique
Département d'Informatique

Concours d'accès au Doctorat LMD Informatique 2019-2020

Epreuve 1: Algorithmique Avancée et Complexité

Date : 02/11/2019, Durée : 1h 30m**,** 13.00 h -14.30 h

Exercice 1: <07 points>

Soit **T** un tableau contenant ***n*** entiers distincts.

1. Proposer un algorithme naïf pour trouver le plus grand et deuxième plus grand élément dans **T** ? Calculer la complexité de cet algorithme.
2. On veut trier le tableau **T** de la manière suivante : on commence par le début du tableau, on compare deux éléments consécutifs (***i***, ***i+1***): s'ils sont dans l'ordre on se déplace d'un cran vers la fin du tableau (incrémente) ou on s'arrête si la fin est atteinte; sinon, on les permute et on se déplace d'un cran vers le début du tableau (décrémente) ou si on est au début du tableau alors on se déplace d'un cran vers la fin (incrémente).
   a. Proposer une procédure effectuant ce tri ?
   b. Donner sa complexité au meilleur cas et au pire cas ? Justifier.

Exercice 2: <07 points>

L'algorithme suivant est celui d'une fonction qui prend en entrée un entier non nul et fournit en sorite un entier non nul.

```
int f(int n) {
if (n == 1) return 1 ;
if (n == 2) return 2 ;
for (int k = f(n - 1) ; k > 1 ; k = f(k - 1))
        if ((n % k) == 0) return f(n - 1) ;
return n ;
}
```

1) Donner les résultats respectifs retournés par cette fonction pour les valeurs 3 , 4 et 5 de n.
2) Construire l'arbre des appels récursifs pour n=5.
3) Expliquer ce que fait cet algorithme. Quel inconvénient présente-t-il ? Quelle solution proposez-vous ?
4) Donner une équation récurrente exprimant le nombre T(n) d'opérations modulo (%) qu'effectue cet algorithme en fonction de n.
5) Donner une version itérative de cet algorithme en $O(n\sqrt{n})$.

Exercice 3: <06 points>

On considère un tableau à une dimension contenant des lettres majuscules. On désire compter la fréquence de chacune des 26 lettres de l'alphabet.

- Ecrire deux procédures qui donnent en sortie un tableau de fréquence : l'une où le tableau est parcouru 26 fois, et l'autre (plus performante !) où le calcul est fait en un seul parcours. On pourra supposer que l'on dispose d'une fonction auxiliaire *position* (lettre) qui pour chaque lettre donne sa position dans l'alphabet.

Bon courage