Feuille d'exercices no. 1

Introduction

Exercice 1. [*Tri par dénombrement*]
Soit un tableau T dont les éléments sont des entiers compris entre 1 et une valeur donnée (connue) max. On peut trier T en comptant tout d'abord le nombre de 0, le nombre de 1, le nombre de 2, ..., le nombre de max en entrée. Ensuite, il suffit de remettre dans le tableau le bon nombre de copies de chaque valeur.

1. Ecrire cet algorithme de tri.
2. Calculer sa complexité.
3. Discuter cette complexité par rapport à la borne théorique du tri par comparaison de n éléments, qui est de ... mais de combien est-elle ? !

Exercice 2. [1] [*Où l'on calcule la complexité d'un algorithme récursif sans branchement*]
On place 1 000 euros sur un livret qui rapporte 0,5 % par mois d'intérêts. Chaque fin de mois, on y verse la somme de 30 euros. Ce livret est bloqué pour 5 ans ce qui signifie que, sur cette période, il est donc impossible de retirer de l'argent.

1. Vérifier qu'à la fin du premier mois, la somme présente sur le livret est égale à 1 035 euros.
2. Donner un algorithme itératif qui permet d'afficher en sortie finale la somme présente sur ce livret au bout d'une année. Calculer sa complexité.
3. Donner un algorithme récursif pour répondre à la même question. Pour cela, écrire une fonction récursive *CalculeSuite*(S, i) qui, pour une somme S présente sur le livret à la fin du i-ème mois, calcule la somme finale présente sur le livret au bout d'une année.
4. En notant $C(i)$ le nombre total d'opérations nécessaires pour *CalculeSuite*(S, i) (noter que ce nombre d'opérations ne dépend pas de S), écrire une fonction récursive qui exprime $C(i)$ en fonction de $C(i+1)$.
5. Obtenir une forme explicite (c'est à dire non-récursive) de C et en déduire la complexité de votre algorithme récursif.

1. Réécrit à partir d'un exercice de bac général, 2010

Pour aller plus loin ...

Exercice 3. [*Où l'on calcule la complexité d'un algorithme récursif avec branchement*]

1. Ecrire un algorithme récursif de recherche dichotomique d'un élément dans une liste d'entiers triés
2. En notant $C(n)$ le nombre de comparaisons effectuées pour chercher l'élément dans une liste de n éléments, écrire une inégalité qui permette de majorer la valeur de $C(n)$ en fonction de $C(\frac{n}{2})$.
3. En résolvant cette inégalité, trouver la valeur de $C(n)$ (On supposera ici que n est une puissance de 2).
4. Pouvez-vous en déduire la complexité (c'est à dire l'ordre de grandeur) de l'algorithme ?