Partiel

Exercice 1 (On peut utiliser un théorème vu en cours).

Vous devez choisir parmi ces trois algorithmes:

1. L'algorithme A résout les problèmes de taille n en les divisant en cinq sous-problèmes de taille $n/2$, il résout les sous-problèmes récursivement, et il combine les solutions en temps $O(n)$.
2. L'algorithme B résout les problèmes de taille n en résolvant récursivement deux sous-problèmes de taille $n-1$, il résout les sous-problèmes récursivement, et il combine les solutions en temps $O(1)$.
3. L'algorithme C résout les problèmes de taille n en les divisant en neuf sous-problèmes de taille $n/3$, il résout les sous-problèmes récursivement, et il combine les solutions en temps $O(n^2)$.

Quelle est la complexité de chacun de ces algorithmes? Lequel est le plus rapide?

Exercice 2 (L'utilisation d'un quelconque théorème entraine un zéro pour cet exercice).

Soit $f : \mathbb{R}_+ \setminus \{0\} \to \mathbb{R}$ une fonction telle que

$$f(xy) = f(x) + f(y)$$

1. Montrer que $f(1) = 0$.
2. Montrer que $f(x/y) = f(x) - f(y)$.
3. Donnez la définition de $\log_b(x)$, où $b \geq 2$ est un entier.
4. Montrer que $f(x) = \log_b(x)$ satisfait la condition $f(xy) = f(x) + f(y)$.
5. Montrer que $\log_a(x) = \log_a(b) \times \log_b(x)$.

Exercice 3 Donnez un algorithme de complexité $O(n^k)$, où k est un réel fixé strictement inférieur à 2, pour multiplier deux entiers de n digits (par-exemple $k = 1.58$).

Exercice 4 On considère des fonctions de $\mathbb{N} \setminus \{0\}$ dans lui-même

1. Donnez les définitions des notations asymptotiques $O(f(n))$, $\Omega(f(n))$ et $\Theta(f(n))$.
2. Montrer que $\sum_{i=1}^{i=k} a_i O\big(f_i(n)\big) = O\Big(\sum_{i=1}^{i=k} a_i f_i(n)\Big)$ où k,$a_1, \ldots, a_k$ sont des entiers et $f_1(n), \ldots, f_k(n)$ des fonctions.

Exercice 5 L'algorithme d'Al Kwarizmi pour multiplier deux entiers opère ainsi:
Diviser le premier entier par 2 en arrondissant à l'inférieur et multiplier le second par 2 jusqu'à ce que le premier vaille 1, puis additionner les valeurs prises par le second lorsque le premier avait une valeur impaire.
Par exemple:

premier	11	5	2	1
second	13	26	52	104

On obtient $11 \times 13 = 13 + 26 + 104 = 143$.

1. Donnez le code Java de cet algorithme.
2. Prouvez sa validité.