Université de Bejaia.

Faculté des Sciences Exactes.

Département Informatique

**Travaux Dirigés**

**Programmation Avancée**

**Master 1 en Informatique**

**Options RN: RS et SIA**

***Prof. BOUALLOUCHE Louiza***

**Série n°2**

**Exercice 1.**

Soit la suite récurrente donnée par la relation suivante

 1 *Si* *n = 0*

 X(n) =

 *3 \* X(n-1) Sinon*

1. Ecrire une fonction récursive qui retourne *X(n)* pour un *n* donné ?
2. Quel est le type de récursivité de cette fonction ?
3. Montrer que cette fonction se termine.
4. Montrer que cette fonction est valide.
5. Soit à calculer la complexité T(n) de cette fonction en nombre multiplications.
	1. Donner la relation de récurrence entre T(n) et T(n-1) ;
	2. La résoudre pour obtenir T(n).

**Exercice 2.**

Une suite de Fibonacci est définie comme suit :

 *1 Si n = 0 ou n = 1*

 *Fib(n) = Fib(n-1) +Fib(n-2) Sinon*

1. Ecrivez une fonction récursive calculant *Fib (n).*
2. Montrez que la complexité (en nombre d’additions) de cette fonction est en Ω(2n/2).
3. Ecrivez une fonction récursive qui calcule, pour *n > 0*, le couple (*Fib(n),* *Fib*(*n-1)*)*.*
4. Utilisez la fonction précédente pour écrire une nouvelle fonction calculant Fib*(n).*
5. Quelle est la complexité en nombre d’additions de cette fonction?

**Exercice 3.**

1. Montrez comment multiplier deux polynômes linéaires $ax+b et cx+d $ à l’aide de trois multiplications seulement.
2. Donnez deux algorithmes **« *Diviser pour Régner*»** permettant de multiplier deux polynômes de degré au plus *n* et de complexité Θ$( n^{log\_{2 }3}$).
	1. Le premier algorithme devra couper les coefficients du polynôme d’entrée en deux moitiés, l’une supérieure et l’autre inférieure.
	2. Le second algorithme devra séparer les coefficients du polynôme d’entrée selon la parité de leur indices.
3. Montrez que deux entiers à *n*  bits peuvent être multipliés en Θ$( n^{log\_{2 }3}$ ) étapes.

**Exercice 4.**

On considère l’algorithme de Karashuba qui multiplie deux nombres de *n* chiffres en un nombre de multiplications beaucoup plus petit que celui de la multiplication ordinaire.

1. Montrez que cet algorithme se termine.
2. Prouvez que cet algorithme est valide.
3. Donnez sa complexité en nombre de multiplications.

**Exercice 5.**

L’algorithme Tri fusion repose sur le principe « Diviser pour régner ». Pour trier un tableau de *n* éléments, on le découpe en deux tableaux de *n/2* éléments à trier récursivement et l'étape fusion permettra de combiner les deux solutions (tableaux de *n/2* triés).

Reprendre les mêmes questions de l’exercice 4.