

Rattrapage

Exercice 1 On considère la complexité de la fonction `main` ci-dessous en fonction du paramètre d'entrée n :

```
import java.util.Random;
public class Multi{
    public static int increm(int n){if(n==0) return 0; else return ++n;}
    public static int decrem(int n){if(n==0) return 0; else return --n;}
    public static int MultiDecrem(int n, int k){
        while(n != 0 && k !=0){
            n=decrem(n);
            k--;
        }
        return n;
    }
    public static void main (String...args){
        int n = args[0];
        final int seed = args.length > 1 ?
            Integer.parseInt(args[1]) : (int) System.currentTimeMillis();
        final Random random = new Random(seed);
        int compt=n;
        while(compt--!=0){
            int alea = random.nextInt(3);
            if(alea==0) n=increm(n);
            if(alea==1) n=decrem(n);
            if(alea==2) n=MultiDecrem(n,n/2);
        }
    }
}
```

- 1) Montrer que la complexité est $\Omega(n)$ et $O(n^2)$.
- 2) Montrer que la complexité est $\Theta(n)$.

Exercice 2 Soient a,b,c,d des réels. Le produit de deux nombres complexes $x = a + ib$ et $y = c + id$ est $xy = (ac - bd) + i(ad + bc)$. Donnez un algorithme qui calcule xy en effectuant que 3 produits de deux nombres réels au lieu de 4.

Exercice 3 Soient N et M deux entiers n -digits à multiplier où n est (dans un premier temps) une puissance de 2. On pose $N = a \times 10^{n/2} + b$ et $M = c \times 10^{n/2} + d$ avec a, b, c, d des nombres entiers $\frac{n}{2}$ -digits.

1) Donnez une procédure Java `int A(int N, int n)` qui renvoie la valeur de a en fonction de l'entier N de n -digit et une procédure Java `int B(int N, int n)` qui renvoie la valeur de b en fonction de l'entier N de n -digit.

2) Montrer que pour tous couples d'entiers N, M on a:

$$(\alpha) \quad NM = ac \times 10^n - ((a - b)(c - d) - ac - bd) \times 10^{n/2} + bd$$

$$(\beta) \quad NM = ac \times 10^n + ((a + b)(c + d) - ac - bd) \times 10^{n/2} + bd$$

3) En utilisant l'une des deux égalités (α) ou (β) et les procédures **A** et **B** donnez le code Java d'une procédure récursive `int Mult(int M, int N, int n)` qui retourne le produit de deux nombres entiers N et M de n -digits. Expliquez le choix de votre égalité.

4) Donnez l'équation de récurrence du temps d'exécution $T(n)$ de **Mult** et en déduire sa complexité.

5) Comment étendre son utilisation quand n n'est pas une puissance de 2 en gardant la même complexité?

Exercice 4 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:

$$\begin{array}{r|cccc} \text{premier} & 11 & 5 & 2 & 1 \\ \text{second} & 13 & 26 & 52 & 104 \end{array}$$

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

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