Compte rendu TP2 API

Paul Blanchet Eloi Charra

21/10/2021

Contents

1 Fibonacci

1.1 Récursion

Pour la première version de l'algorithme de fibonacci, on remarque que la complexité en temps croît rapidement en fonction de N entré. On obtient :

N	Résultat	Temps
10	55	0
20	6765	0
40	102334155	1s
75	?	trop long
100	?	trop long

Cette version récursive de l'algorithme possède une complexité de $\mathcal{O}(2^n)$ ce qui fait que des valeurs supérieures à 40 metteront plus de 5 secondes à se calculer et cette durée ce cessera d'augmenter. Cette complexité est due à l'addition des deux sous calculs de `n - 1` et `n - 2`.

1.2 Iteration

La version itérative de l'algorithme possède une complexité de $\mathcal{O}(n)$ (car l'algorithme ne comporte qu'une boucle `for`), ce qui nous permet d'obtenir les résultats plus rapidement que la version récursive.

N	Résultat	Temps
10	55	0
20	6765	0
40	102334155	0
75	2111485077978050	0
100	3736710778780434371	0

Nous remarquons que les valeurs sont calculées instantanément, il faudra attendre d'avoir N supérieur à 10000000000000 (dix billions) pour que le résultat sorte en plus de 1 seconde. De plus, nous avons changé la taille sur laquelle les entiers sont stockés en choissant le type long (ou long long) pour pouvoir acceuilir des valeurs plus

grandes dans nos variables.

1.3 Récursion v2

Cette version se repose sur la version itérative en calculant `(F(n),F(n-1))` à partir de `(F(n-2),F(n-1))`. Nous obtenons les mêmes résultats que la version itérative :

N	Résultat	Temps
10	55	0
20	6765	0
40	102334155	0
75	2111485077978050	0
100	3736710778780434371	0

Cet algorithme possède une complexité de $\mathcal{O}(n)$ **???**

1.4 Nombre d'or

Nous pouvons remarquer qu'en utilisant cet aglorithme, les résultats sont globalement identiques aux algorithmes précédents. Cependant, nous pouvons voir une légère différence qui s'accentue en même tant que N grandi du au fait que des nombres réels sont utilisés en plus de l'arrondi de $\sqrt{5}$. De plus nous pouvons voir que pour `N = 100`, nous obtenons un nombre négatif puisque nous dépassons la zone mémoire associée à cette variable.

N	Résultat	Temps
10	55	0
20	6765	0
40	102334155	0
75	2111485077978055	0
100	-4124817371235594752	0

La complexité de cet algorithme est de $\mathcal{O}(**?? * *)$

1.5 Matrices

N	Résultat	Temps
10	55	0
20	6765	0
40	102334155	0
75	2111485077978055	0
100	-4 124 817 371 235 594 752	0