TP n° 8 : Suite de Fibonacci

La suite de Fibonacci a été introduite comme problème récréatif par Léonard DE PISE (1175 — v.1250), aussi connu sous le nom de Leonardo FIBONACCI. Elle peut être considérée comme le tout premier modèle mathématique en dynamique des populations. En effet, elle y décrit la croissance d'une population de lapins sous des hypothèses très simplifiées, à savoir : chaque couple de lapins, dès son troisième mois d'existence, engendre chaque mois un nouveau couple de lapins, et ce indéfiniment. Partons donc d'un couple de lapins le premier mois. Le deuxième mois, on n'a toujours que ce même couple, mais le troisième mois on a déjà 2 couples, puis 3 couples le quatrième mois, 5 couples le cinquième mois, etc.

Les nombres de Fibonacci apparaissent souvent dans la nature lorsque des spirales logarithmiques sont construites à partir d'une unité discrète, telles que dans les tournesols ou dans les pommes de pin. Par exemple le nombre de pétales de la marguerite appartient à la suite de Fibonacci (souvent 34, 55 ou 89), ce qui s'explique par le mécanisme de développement de la plante.

Dans ce TP, la suite de Fibonacci $(f_n)_{n\in\mathbb{N}}$ est définie par

$$\begin{cases} f_0 = 0 \\ f_1 = 1 \\ f_{n+2} = f_{n+1} + f_n \quad \text{pour } n \in \mathbb{N} \end{cases}$$

1 Suite de Fibonacci en PYTHON

1. Écrire en PYTHON, en utilisant une boucle `for`, une fonction `fibonacci(n)` qui calcule f_n, c'est-à-dire qui prend en argument un entier $n \geq 0$ et qui renvoie le terme d'indice n de la suite de Fibonacci.
2. Donner un invariant de boucle qui permet de justifier la correction de cette fonction.
3. Écrire cette même fonction en utilisant une boucle `while`.
4. Donner un variant pour justifier la terminaison de la fonction et un invariant pour en justifier la correction.
5. En utilisant une des fonctions précédentes, écrire une fonction `est_fibonacci` qui vérifie si un entier $n \geq 2$ est un entier de Fibonacci, c'est-à-dire s'il existe un indice i tel que $n = f_i$.
6. Trouver un invariant et un variant de boucle pour justifier la correction et la terminaison.
7. Modifier votre fonction pour qu'elle renvoie plutôt la valeur de cet indice s'il existe et -1 sinon.
8. Dans la fenêtre IPYTHON, utiliser la commande magique `%timeit` pour mesurer le temps d'exécution de cette fonction pour certaines valeurs de n. Par exemple, on peut essayer `%timeit est_fibonacci(10 ** 1000)`.
9. À l'intérieur de la fonction `fibonacci`, ajouter la ligne `print("* Calcul de f_{}".format(i))` à l'endroit où l'on effectue le calcul de f_i. Ceci doit nous permettre de visualiser chaque occurrence d'un calcul de f_i.
10. Qu'affiche l'appel `est_fibonacci(100)` ? Commenter.
11. Écrire une fonction `est_fibonacci_efficace(n)` qui vérifie si un entier $n \geq 2$ appartient à la suite de Fibonacci en calculant la suite à la volée plutôt qu'en utilisant la fonction `fibonacci`. *Indication : il suffit d'adapter la version de la fonction `fibonacci` qui utilise une boucle `while`.*
12. Vérifier, en utilisant `%timeit`, que cette fonction est bien plus efficace et commenter.

2 Nombres de Fibonacci se terminant par n zeros

13. Écrire une fonction prenant en argument un entier $n \geq 1$ et déterminant la valeur et le rang du premier entier de Fibonacci non nul se terminant par n zéros.
14. Essayer pour $n = 4$, puis pour $n = 5$. Pour $n = 6$, être très très patient.
15. Pouvez-vous expliquer ce temps de latence ?
16. Modifier la fonction précédente en construisant la suite des restes modulo 10^n des entiers de Fibonacci plutôt que les entiers de Fibonacci eux-mêmes. Pourquoi est-ce suffisant pour répondre à la question ?
17. Observer (et mesurer avec `%timeit`) l'accélération à l'exécution pour $n \in \{3, 4, 5\}$. Pouvez-vous l'expliquer ?

3 Suite de Fibonacci et nombres premiers

18. Écrire une fonction `est_premier` qui vérifie si un entier $n \in \mathbb{N}$ est premier.
19. Écrire une fonction prenant en argument un entier $n \geq 0$ et renvoyant la liste des n premiers entiers de Fibonacci premiers.

On ne sait pas si la suite de Fibonacci contient ou non une infinité de nombres premiers. Pour en savoir plus, on pourra lire l'article ci-dessous.

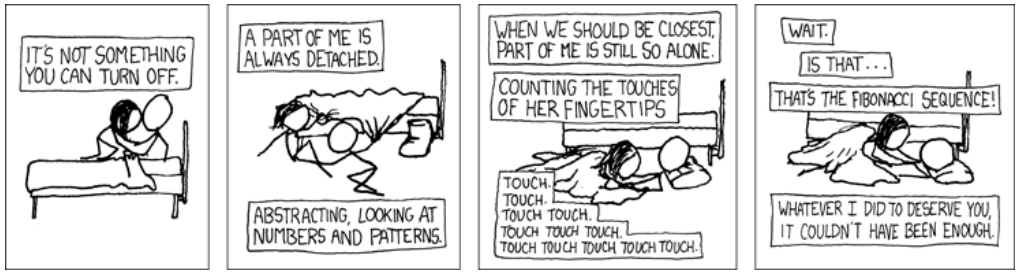
4 La suite de Fibonacci dans le Project Euler

20. Résoudre :
    - https://projecteuler.net/problem=2
    - https://projecteuler.net/problem=25

5 Pour les plus rapides

21. Lire l'article ci-dessous et vérifier la conjecture 1 pour autant de nombres premiers que possible (cf. partie *Travaux pratiques*).

http://images.math.cnrs.fr/Mysteres-arithmetiques-de-la-suite-de-Fibonacci.html



https://xkcd.com/289/

Complexité

On dispose d'un ordinateur capable d'effectuer un milliard d'opérations élémentaires par seconde. Combien de temps faut-il pour résoudre un problème de taille n, en supposant qu'il faut $f(n)$ opérations élémentaires ? Compléter le tableau suivant en choisissant l'unité appropriée : ns, μs, ms, s, minutes (min.), années (a.), jamais (∞).

		Temps nécessaire				
Croissance	$f(n)$	$n = 10$	$n = 100$	$n = 1000$	$n = 10^6$	$n = 10^9$
logarithmique →	$\ln n$	2 ns				
	$\sqrt{n}$		10 ns			
linéaire →	n					1 s
	$n \ln n$					
polynomiale →	n^2					
polynomiale →	n^3					
exponentielle →	2^n					
	$n!$	4 ms				∞

Pour une fonction $f(n)$ donnant le nombre d'opérations élémentaires nécessaires pour résoudre un problème de taille n, quelle est la taille maximale d'un problème que l'on peut résoudre en 1 s ? Compléter le tableau suivant.

$f(n)$	$\ln n$	$\sqrt{n}$	n	$n \ln n$	n^2	n^3	2^n	$n!$
$n_{\max}$			10^9					