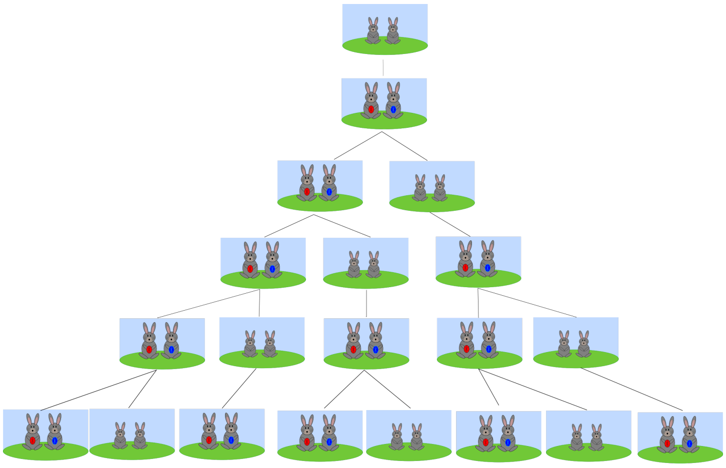
Cours de Complexité

Fiche de TD n°4

Les nombres de Fibonacci

Extrait de Wikipedia, consulté le 7-11-2016 :

> *La suite de Fibonacci est une suite d'entiers dans laquelle* ***chaque terme est la somme des deux termes qui le précèdent****. Elle commence généralement par les termes 0 et 1 (parfois 1 et 1) et ses premiers termes sont : 0, 1, 1, 2, 3, 5, 8, 13, 21, etc.*
>
> *Elle doit son nom à Leonardo Fibonacci qui, dans un problème récréatif posé dans l'ouvrage Liber abaci publié en 1202, décrit la croissance d'une population de lapins : « Un homme met un couple de lapins dans un lieu isolé de tous les côtés par un mur. Combien de couples obtient-on en un an si chaque couple engendre tous les mois un nouveau couple à compter du troisième mois de son existence ? »*



> *Cette suite est fortement liée au nombre d'or, φ (phi). Ce nombre intervient dans l'expression du terme général de la suite. Inversement, la suite de Fibonacci intervient dans l'écriture des réduites de l'expression de φ (phi) en fraction continue : les quotients de deux termes consécutifs de la suite de Fibonacci sont les meilleures approximations du nombre d'or.*

1. A partir de la définition en gras ci-dessus écrivez une version récursive de la fonction fibo(n), qui renvoie le $n^{ème}$ nombre de Fibonacci. Quelle est la complexité de cet algorithme ? pourquoi est-il si lent ?

On va chercher à éviter de recalculer plusieurs fois le même nombre. Dès qu'un nombre de Fibonacci aura été calculé, on va le mémoriser dans une table. Si on a de nouveau besoin de le calculer, on ira simplement lire sa valeur dans la table. C'est une amélioration appelée « Recensement ».

2. Ecrivez une version « par recensement » de la fonction qui calcule le $n^{ème}$ nombre de Fibonacci. Quelle est sa complexité en temps ? en mémoire ?

3. Ecrivez une version linéaire en temps et constante en mémoire pour calculer le $n^{ème}$ nombre de Fibonacci.

4. Calculez le carré, le cube, … la puissance $n^{ème}$ de la matrice suivante :

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$$

5. Proposez un algorithme logarithmique pour calculer x^n

6. En déduire un algorithme logarithmique pour calculer le $n^{ème}$ nombre de Fibonacci