Travaux Pratiques d'algorithmique n°4

Cours d'Informatique de Deuxième Année

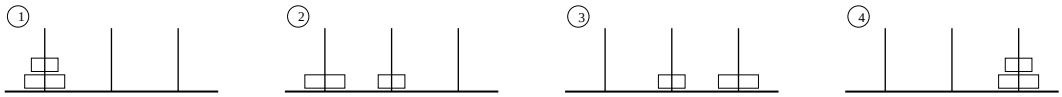
—DEUG Sciences S3-MI—

Récursivité

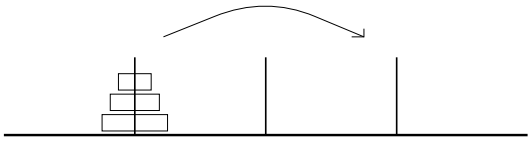
L'objet de cette séance est d'introduire la notion de récursivité. Vous devrez présenter clairement les algorithmes, ainsi que **le calcul de leur complexité**.

▸ Exercice 1.

Nous cherchons à résoudre le problème des "tours de Hanoï". Ce problème consiste à déplacer `n` *disques d'un piquet de départ à un piquet d'arrivée en utilisant un troisième piquet. La seule contrainte tient au fait qu'il est interdit de poser un disque sur un disque plus petit que lui.*



Résolution du problème pour $n = 2$.



Exemple du problème pour $n = 3$.

1. *résoudre "à la main" l'algorithme pour* $n = 3$;
2. *en déduire un algorithme récursif de résolution du problème ;*
3. *écrire la fonction correspondant à cet algorithme : la fonction devra afficher la suite des déplacements des disques ; elle retournera le nombre de déplacements nécessaires.* `int Hanoi(int n, char dep, char tmp, char arr)` *où* `n` *est la hauteur de la tour, et* `dep,tmp,arr` *sont les noms (un seul caractère) des trois piquets de départ, de transition, et d'arrivée ;*
4. *donner enfin, la complexité en temps de votre algorithme.*
5. *On suppose maintenant que la pile de disques de départ est constituée de disques de n tailles différentes, le nombre de disques par taille étant de p. Écrire la fonction* `Hanoi2` *qui résout le problème des "tours de Hanoï" pour une pile de disques ainsi constituée.*

▸ **Exercice 2.**

Écrire une fonction de prototype `void AffDecomp(int n, int b)` *qui affiche la décomposition de l'entier* `n` *dans la base* `b`. *Par exemple, pour* `n`= 21 *et* `b`= 8, *l'affichage sera* 25. *Pour cela, vous utiliserez un algorithme* ***récursif*** *dont vous estimerez la complexité.*

▸ **Exercice 3.** ***(Somme des chiffres d'un entier)***

- *On veut calculer la somme des chiffres d'un entier. Par exemple, la somme des chiffres formant* 3281 *est* 14.

  Écrire une fonction `SommeChiffre` *prenant en argument un entier* `n` *et qui calcule récursivement la somme des chiffres qui le forment.*

  Calculer la complexité de cette fonction.

- *Le critère de divisibilité suivant est connu de tous : un entier est divisible par* 3 *si la somme des chiffres qui le forment est un multiple de* 3. *Autrement dit :*

$$n \equiv 0\ [3] \Leftrightarrow \sum_{i=0}^{k} c_i \equiv 0\ [3]$$

  On remarquera que la somme des chiffres d'un entier supérieur à 10 *est toujours strictement plus petit que cet entier. Cette simplification (passer d'un entier à la somme de ses chiffres) fait naître un algorithme récursif.*

  Un entier `n` *est-il divisible par* 3 *? Si l'entier est strictement plus petit que* 10 *: facile à déterminer ; sinon on se pose la même question sur la somme de ses chiffres.*

  Effectuer le calcul complet pour 3281 *et* 1847394.

  Écrire une fonction récursive réalisant cet algorithme.

▸ **Exercice 4.**

Écrire une fonction `Puissance` *en appliquant cette définition :* $b^0 = 1$, $b^n = b\, b^{n-1}$.

Écrire une fonction `PuissanceBis` *qui tient compte des relations suivantes :*

$$b^n = \begin{cases} 1 & \text{si } n = 0 \\ (b^{n/2})^2 & \text{si } n \text{ est pair} \\ b\ (b^{(n-1)/2})^2 & \text{si } n \text{ est impair} \end{cases}$$

Évaluer la complexité dans les deux cas.