Méthode de Horner

1. Présentation

Soit un polynôme $p(x) = a_0 + a_1.x + a_2.x^2 + a_3.x^3 + \ldots + a_n.x^n$.

La première idée pour calculer p en x_0 consiste à calculer chaque puissance de x_0, de multiplier par les coefficients a_i, puis de tout additionner.

Cette méthode oblige à calculer plusieurs fois les mêmes termes.

La méthode de Horner consiste à remarquer que :

$$p(x_0) = a_0 + x_0.(a_1 + x_0.(\ldots + x_0.(a_{n-1} + a_n.x_0))\ldots)$$

2. Travail demandé

Le travail demandé est à remplir dans le fichier horner.py que vous aurez préalablement copié sur votre espace. Il contient déjà la fonction ***list_rand(N)*** qui retourne une liste aléatoire de taille N. Cela vous permet de tester vos fonctions sur des polynômes aléatoires de taille N.

2.1. *Ecrire une fonction* ***horner_naif****(p,x) retournant la valeur de* p *en* x*, où* p *est la liste des coefficients* a_i *du polynôme :* p=[a_0, a_1 ,…, a_n], **en utilisant une méthode naïve.**

2.2. *Écrire une fonction récursive* **horner_rec**(p,x) *retournant la valeur de* p *en* x*, où* p *est la liste des coefficients* a_i *du polynôme :* p=[a_0, a_1 ,…, a_n] **en utilisant une méthode récursive.**

2.3. *Déterminer le temps de calcul pour chacune des deux approches en testant des cas comparables. On pourra pour cela utiliser la bibliothèque time. (voir annexe). Si le temps le permet, comparer avec le la méthode « classique », c'est à dire telle que vous l'auriez codé sans la méthode de Horner.*

Annexe TIME

On rappelle comment mesurer le temps de calcul d'instruction à l'aide de la bibliothèque `time`. La fonction `time()`de cette bibliothèque renvoie le temps instantané.

Etape 1. Ecrire en en-tête de programme la commande permettant d'importer la librairie :

```
import time # import de la bibliothèque
```

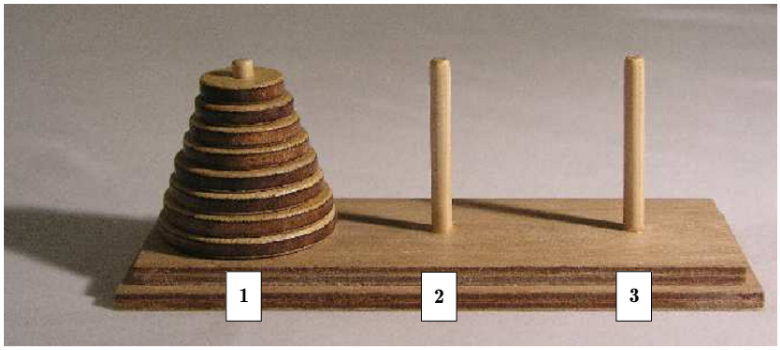
Etape 2. Ecrire dans le corps du programme :

```
debut= time.time() # temps initial
Instructions dont on veut mesurer le temps d'exécution
fin =time.time() # temps final
print('temps de calcul =', fin-début ) # affichage du temps de calcul
```

Les tours de Hanoï

1. Présentation

Les Tours de Hanoï sont un jeu inventé par le mathématicien Édouard Lucas en 1883. Le jeu fait appel **à trois piquets verticaux** notés 1, 2 et 3 et à **n disques** de tailles strictement différentes munis d'un trou en leurs centres. Initialement, les disques sont empilés sur le piquet 1 par ordre de tailles décroissantes.



Le but du jeu consiste à déplacer l'ensemble des disques pour que ceux-ci se retrouvent enfilés autour du piquet 3 en respectant les règles suivantes :

- les disques doivent être déplacés **un par un** ;
- un disque ne doit jamais être posé sur un disque plus petit que lui (mais il n'est pas obligé d'être au-dessus du disque immédiatement plus grand).

Ce problème se résout facilement de manière récursive. Pour déplacer **nb** disques de la position **pos_init** à la position **pos_fin** il faut :

- déplacer les **nb-1** premiers éléments vers la position restante **pos_inter**,
- déplacer l'élément restant (le plus gros) de la position **pos_init** à la position **pos_fin**,
- déplacer la pile des **nb-1** éléments de la position **pos_inter** à la position **pos_fin** sur le plus gros déjà en place.

2. Travail demandé

Travail 1. *Donner les déplacements nécessaires pour résoudre le problème avec 1, 2 et 3 disques. On pourra abréger le «Je déplace un disque du piquet 3 au piquet 2 » par « 3→2 ».*

Travail 2. *Donner de façon générale, le nombre de déplacements nécessaires pour déplacer n disques.*

Travail 3. *Écrire une fonction récursive* ***hanoi****(nb, pos_init, pos_int, pos_fin) résolvant le problème en affichant les étapes sous la forme « Déplacer le disque de 1 vers 0 ». Vérifier le résultat obtenu sur un exemple à trois disques.*

Travail 4. ***Nombre de coups :*** *Modifier le programme afin de compter le nombre de coups.*