ALGORITHMIQUE

Algorithme = méthode pour résoudre un problème (like une recette de couisine)

Structure de données = façon d’organiser des données

Θ(1) (= coût constant) < Θ(log(n)) < …

Déclaration de tableau

* Java : int[] tableau
* C : int tableau[10]
* Python : tableau = [10]

COMPLEXITÉ

La notion de complexité fait référence à l’efficacité d’un algorithme.

Il existe 2 formes de complexité pour un algorithme :

* Complexité spatiale = espace mémoire occupé
* Complexité temporelle = temps d’exécution

Temps d’exécution

**Pourquoi ?**

Exemple 1 : Construire une ville de 50 maisons en évitant aux livreurs de pizzas qui suivent les rues un trajet trop long depuis la pizzeria

Organisation 1 : Linéaire. Numéros croissants. Pizzeria au numéro 1

Organisation 2 : Embranchements. Forme d’arbre binaire de recherche. Ouest = n° < k. Est = n° > k. Ici k = 8 donc pizzeria au n°8

On suppose qu’il faut une unité de temps pour passer d’une maison a un autre

Dans le cas le pire, quel est le temps mis par un livreur pour aller jusqu’à une maison depuis la pizzeria

Orga 1 : il faut 14 pas O(n-1) / Orga 2 : il faut 3 pas O(log2(n))

Exemple 2 :

Déterminer si 2 ensembles E1, E2 de n entiers ont une valeur commune

Algo 1 : comparer successivement chaque élément de E1 avec chaque éléments de E2. Il faudra environ n² comparaisons

Algo 2 : avec une structure de données adaptée, on peut résoudre le problème avec environ nlog(n).

Pour n = 106 : nlog(n) -> 0.07s / n² -> 16min40s

**Comment ?**

???

Définition :

Le temps de calcul (ou complexité) d’un algo est la fonction qui a un entier n associe le nombre maximal d’instructions élémentaires que l’algo effectue, lorsqu’on travaille sur des objets de taille n

La fonction compte les instructions élémentaires au lieu de mesurer le temps.

TRIS

**Questions :**

1. Tri par sélection  
   -> 3 2 8 1 7 (début)  
   -> 1 2 8 3 7  
   -> 1 2 8 3 7  
   -> 1 2 3 8 7  
   -> 1 2 3 7 8  
     
   Tri par insertion