COURS ALGORITHME : COMPLEXITÉ ET TRIS

CHARLES 'ARYS' YAICHE

RAPPEL

- Structure abstraite
- Structure de vecteur (tableau)

COMPLEXITÉ : DÉFINITION

- La complexité algorithmique est le calcule de ressource d'un algorithme, cela consiste le plus souvent a calculer le nombre d'opération d'un algorithme.
- La complexité est l'évaluation de la performance d'un algorithme

si je donne à mon programme une entrée de taille N, quel est l'ordre de grandeur, en fonction de N, du nombre d'opérations qu'il va effectuer ?

COMPLEXITÉ DE CODE VS COMPLEXITÉ D'ALGO

Code	Algorithme
Contrainte Matériel Rapidité arbitraire Besoin d'une implémentation Complexité Mémoire	Abstrait Rapidité fixe
Complexité d'opérations	

COMPLEXITÉ ASYMPTOTIQUE

Cas 1. faire n fois l'opération A -> la complexité est n (il y a n opération)

*Cas 2. faire n fois (l'opération B puis l'opération C) -> la complexité est 2n (on fait n * b puis n * c)*

Dans le calcule de complexité, on s'intéressera à l'ordre de grandeur le plus grands

Ici, si il y a un très grand nombre de n (voir vers l'infini), il n'y a pas de différence entre le cas 1 et le cas 2.

Rappel mathématique

$$\lim_{n\to\infty} 2n = \lim_{n\to\infty} n$$

NOTATION O (GRAND O)

- On utilisera la notation O() pour noter la complexité.

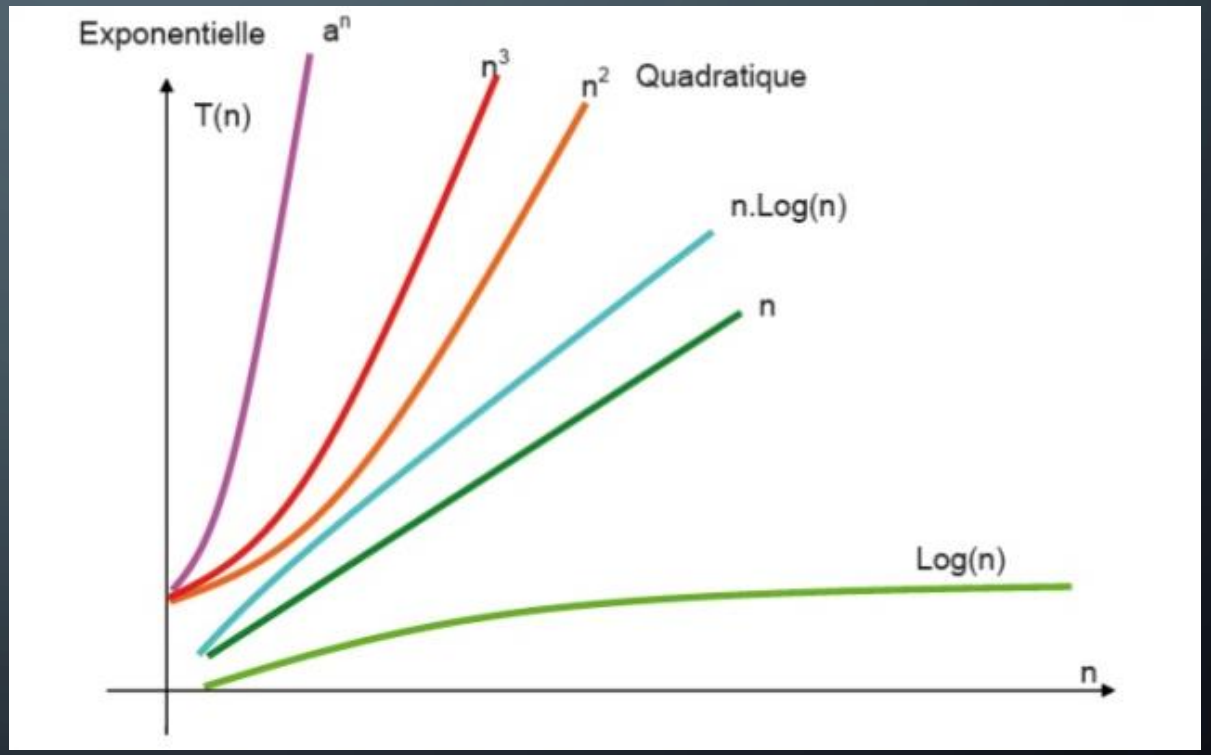
Exemple :

un algorithme faisant n^2 opération aura une complexité $O(n^2)$

- Notation dans le pire cas

GRAND O PARTICULIERS

Complexité	Classe
O(1)	Constant
O(Log(n))	Logarithmique
O(n)	Linéaire
O(nLog(n))	Sous-quadratique
$O(n^2)$	Quadratique
$O(n^3)$	Cubique
$O(2^n)$	Exponentiel
O(n!)	Factorielle



CE QU'IL FAUT RETENIR

- Pour nous complexité est le calcule du nombre d'opération d'un algorithme
- On s'intéresse à l'ordre de grandeur le plus grand
- La notation O
- On considère le pire cas

EXEMPLE 1 : COMPLEXITÉ CONSTANTE

```
algo fonction max : Entier
    Paramètres
        entier a
        entier b
    Debut
        si a > b alors
            retourne a
        sinon
            retourne b
    Fin
Fin algo
```

EXEMPLE 2 : COMPLEXITÉ LINÉAIRE

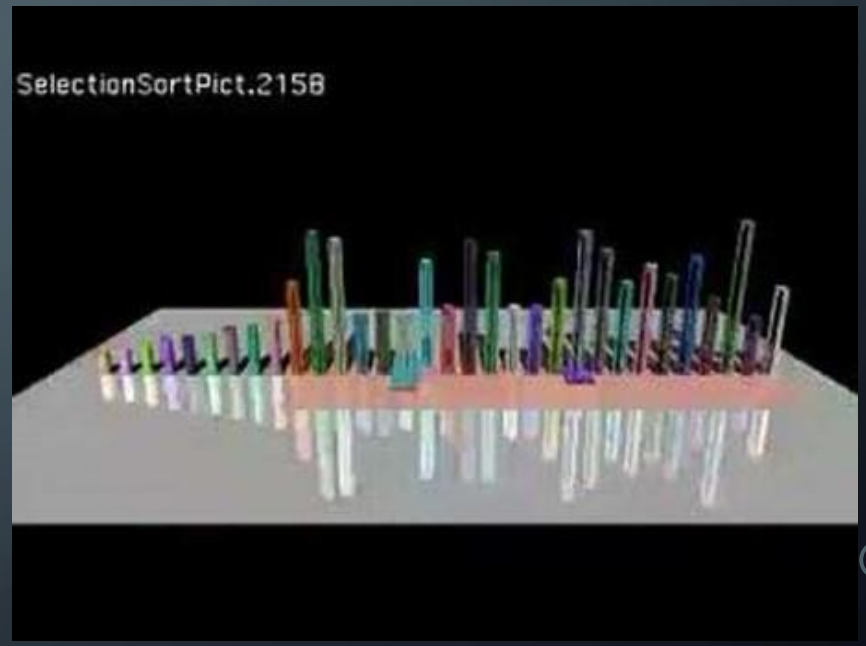
```
algo fonction puissance : Entier
    Paramètres
        Entier a
        Entier b
    Variables
        Entier boucle
        Entier resultat
    Debut
        boucle <- 0
        resultat <- a

        tant que (boucle < b - 1) faire
            resultat <- resultat * a
        fin tant que

        retourne resultat
    Fin
Fin Algo
```

TRI DANS UN TABLEAU : SÉLECTION

45	122	12	3	21	78	64	53	89	28	84	46
3	122	12	45	21	78	64	53	89	28	84	46
3	12	122	45	21	78	64	53	89	28	84	46
3	12	21	45	122	78	64	53	89	28	84	46



TRI PAR SELECTION : ALGO

```
algo procédure insert_sort
    Paramèters
        Tableau[Entier] t
    Variables
        Entier i
        Entier j
        Entier temp
        Entier posmini
    Debut
        Pour i ← 0 à 10
            posmini ← i
            Pour j ← i + 1 à 11
                Si t(j) < t(posmini) Alors
                    posmini ← j
                Finsi
            fin pour
        temp ← t(posmini)
        t(posmini) ← t(i)
        t(i) ← temp
        fin pour
    Fin
Fin
```

TRI A BULLE

Prenons chaque élément d'un tableau, et comparons-le avec l'élément qui le suit. Si l'ordre n'est pas bon, on permute ces deux éléments. Et on recommence jusqu'à ce que l'on n'ait plus aucune permutation à effectuer. Les éléments les plus grands « remontent » ainsi peu à peu vers les dernières places

TRI À BULLE: ALGO

```
algo procédure bubble_sort
    Paramètres
        Tableau[Entier] t
    variable
        Entier i
        Entier j
        Boolean permut
        Entier temp
    Debut
        permut <- vrai
        tant que permut alors
            permut <- faux
            pour i <- 0 à taille(t) - 1
                si t[i] > t[i + 1] alors
                    temp <- t[i]
                    t[i] <- t[i + 1]
                    t[i + 1] <- temp
                    permut <- vrai
                fin si
            fin pour
        fin tant que
    Fin
FinAlgo
```

COMPARAISON DES TRI

- https://www.toptal.com/developers/sorting-algorithms

SOURCES

- http://pise.info/algo/techniques.htm
- https://openclassrooms.com/fr/courses/1467201-algorithmique-pour-lapprenti-programmeur/1467409-un-peu-de-pratique
- https://files.gogaz.org/Poly%20Krisboul%20bon.pdf