TRI D'UN TRAVAUX, RECHERCHE D'UN ÉLÉMENT

TRI

- Etant donné une collection d'entier placés dans un tableau. L'idée fondamentale est de trier le tableau dans l'ordre croissant.
- Les opérateurs de comparaison (<=, >=, >, <, …) sont activement utilisés.
- On peut citer quelques algorithmes de tris:
  - Tris élémentaires (tris naïfs)
    - Tri par insertion
    - Tri par sélection
    - …
  - Tris avancés (Diviser pour régner)
    - Tri fusion
    - Tri rapide
    - …

TRI
PAR INSERTION

Le tri par insertion consiste à pré-trier une liste afin d'entrer les éléments à leur bon emplacement dans la liste triée. à l'itération i, on insère le i i-ième élément à la bonne place dans la liste des i−1 éléments qui le précède.

- Principe:
  - On commence par comparer les deux premiers éléments de la liste et de les trier dans un ordre;
  - puis un troisième qu'on insère à sa place parmi les deux précédents;
  - puis un quatrième qu'on insère à sa place parmi les trois autres;
  - ainsi de suite jusqu'au dernier.

TRI
PAR INSERTION

Considérons un tableau d'entiers de n éléments à trier.

- Algorithme

```
Pour (i allant de 2 à n) faire
  j ← i;
  tampon ← tab[i] ;
  Tant que (j>1 ET tab[j-1]>tampon) faire
      Tab[j] ← tab[j-1];
      j ← j-1;
  Fin tant que
  Tab[j] ← tampon;
Fin pour
```

- Complexité

Pour apprécier la complexité de cet algorithme, il suffit d'analyser le nombre de comparaisons effectué ainsi que le nombre d'échange lors du tri. On remarque qu'il s'exécute en $\Theta(n^2)$.

- n|1|(|2| + |3| + n|4|(|5| + |6|) + |7|) = n(|2| + |3| + |7|) + n^2(|5| + |6|) → $\Theta(n^2)$

TRI
PAR SÉLECTION

Le tri par sélection consiste à recherche le minimum parmi les éléments non triés pour le placer à la suite des éléments déjà triés.

- Principe:
  - Il suffit de trouver le plus petit élément et le mettre au début de la liste;
  - Ensuite, de trouver le deuxième plus petit et le mettre en seconde position;
  - Puis, de trouver le troisième plus petit élément et le mettre à la troisième place;
  - Ainsi de suite jusqu'au dernier.

TRI
PAR SÉLECTION

Considérons un tableau d'entiers de n éléments à trier.

- Algorithme

```
Pour (i allant de 1 à n-1) faire
    Pour (j allant de i+1 à n) faire
        Si(Tab[i] > tab[j]) alors
            tampon ← tab[i];
            tab[i] ← tab[j];
            tab[j] ← tampon;
        Fin Si
    Fin pour
Fin pour
```

- Complexité

On remarque qu'il s'exécute en $\Theta(n^2)$.

TRI
PAR FUSION

Le tri par fusion consiste à fusionner deux tableaux triés pour former un unique tableau trié. Il s'agit d'un algorithme "diviser-pour-régner".

- Principe:
- Etant donné un tableau tab[n]:
  - si n= 1, retourner le tableau tab;
  - Sinon:
    - ✓ Trier le sous-tableau tab[1 . . . n/2];
    - ✓ Trier le sous-tableau tab[n/2 + 1 . . . n];
    - ✓ Fusionner ces deux sous-tableaux…

TRI
PAR FUSION

Considérons un tableau d'entiers de n éléments à trier.

- Algorithme

[Devoir maison]

- Complexité

On remarque qu'il s'exécute en $\Theta(n \log_2 n)$ opérations.

TRI
RAPIDE

Le tri rapide ou encore tri de Hoare (du mon de l'inventeur) est aussi un tri basé sur le principe "diviser-pour-régner".

- Principe:
  - Il consiste à placer un élément du tableau (le pivot) à sa place définitive, en permutant tous les éléments qui lui sont inférieurs à gauche et ceux qui lui sont supérieurs à droite (le partitionnement).
  - Pour chacun des sous-tableaux, on définit un nouveau pivot et on répète l'opération de partitionnement.

TRI
RAPIDE

Considérons un tableau d'entiers de n éléments à trier.

- Algorithme

[Devoir maison]

- Complexité

On remarque qu'il s'exécute en $\Theta(n^2)$ dans le pire des cas. Mais elle peut être en $\Theta(n \log_2 n)$ en moyenne.

RECHERCHE D'UN ÉLÉMENT

RECHERCHE LABORIEUSE

Soit x l'élément à rechercher dans un tableau t de n entiers.

- Principe:
  - On parcourt complétement le tableau et pour chaque élément, on teste l'égalité avec x.
  - En cas d'égalité, on mémorise la position.
- Algorithme

```
.....
indice← 0;
Pour i allant de 1 à n faire
    Si(t[i]=x) alors
        indice ← i;
    Fin Si
Fin Pour
retourner indice ;
.....
```

RECHERCHE D'UN ÉLÉMENT
RECHERCHE SEQUENTIELLE

Soit x l'élément à rechercher dans un tableau t de n entiers.

- Principe:
  - On parcourt séquentiellement le tableau jusqu'à trouver l'élément dans une séquence.
  - Si on arrive à la fin sans le trouver c'est qu'il n'est pas contenu dans la séquence.
- Algorithme

```
.....
Pour i allant de 1 à n faire
    Si(t[i]=x) alors
        retourner i;
    Fin Si
Fin Pour
retourner 0;
.....
```

RECHERCHE D'UN ÉLÉMENT

RECHERCHE DICHOTOMIQUE

Soit x l'élément à rechercher dans un tableau t **ordonné** de n entiers.

- Principe:
  - On compare l'élément à rechercher avec celui qui est au milieu du tableau.
  - Si les valeurs sont égales, la tâche est accomplie sinon on recommence dans la moitié du tableau pertinente.
- Algorithme

```
.....
bas ← 1;
haut ← taille(t);
position ← -1;
Repeter
    Si(x = t[milieu]) alors
        position ← milieu;
    Sinon  Si (t[milieu]<x) alors
        bas ← milieu + 1
    Sinon
        haut ← milieu-1
    Fin Si
jusqu'à (x = t[milieu] OU bas > haut)
retourner position
.....
```