algos	tableau		listes SC		listes DC	
	non trié	trié ↗	non triée	triée ↗	non triée	triée ↗
$v \leftarrow$ Access(s, k) (valeur du $k^{\text{è}}$ élé. de s.)	$\Theta(1)$	$\Theta(1)$	$\Theta(k) = O(n)$	$\Theta(k) = O(n)$	$\Theta(k) = O(n)$	$\Theta(k) = O(n)$
$p \leftarrow$ Search(s, v) (position de v dans s)	$O(n)$	$O(\log n)$	$O(n)$	$O(n)$	$O(n)$	$O(n)$
Insert(s, v) (insère v dans S)	$\Theta(1)$	$O(n)$	$\Theta(1)$	$O(n)$	$\Theta(1)$	$O(n)$
Delete(s, p) (supprime la val. à la position p)	$\Theta(1)$	$O(n)$	$O(n)$	$O(n)$	$\Theta(1)$	$\Theta(1)$
$v \leftarrow \min(s)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$
$v \leftarrow \max(s)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$	$\Theta(n)$	$\Theta(1)$
$p \leftarrow$ succ(s, p)	$O(n)$	$\Theta(1)$	$O(n)$	$\Theta(1)$	$O(n)$	$\Theta(1)$
$p \leftarrow$ pred(s, p)	$O(n)$	$\Theta(1)$	$O(n)$	$\Theta(n)$	$O(n)$	$\Theta(1)$

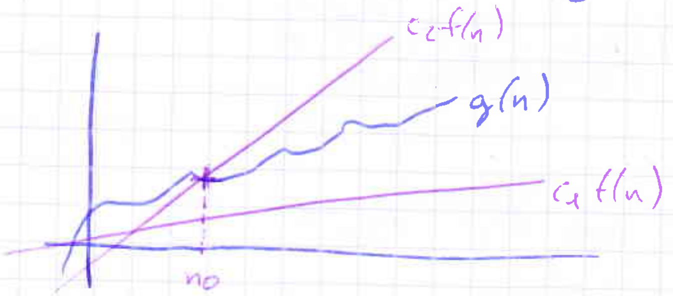
TRI	PIRE	MOYEN	MIEUX	En général	Tri stable	Tri "en place"
Select Sort	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	NON	OUI
Insert Sort	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n)$	$\Theta(n^2)$	OUI	OUI
Merge Sort	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	OUI	NON
Heap Sort	$\Theta(n \log n)$			$\Theta(n \log n)$	NON	OUI ← conso. mém. $O(\log n)$ à cause de la récursion
Quick Sort	$\Theta(n^2)$	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	NON	OUI ← si il est optimisé
Intro Sort	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	NON	OUI
Counting Sort	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	$\Theta(n)$	OUI	OUI

ALGO

http://www.lrde.epita.fr/nadl/ens/algo/

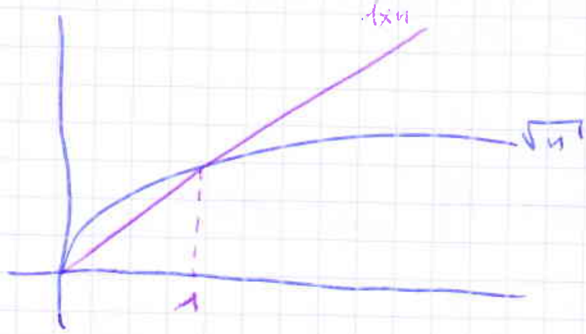
→ LUNDI

* $\Theta(f(n)) = \{ g(n) \mid \exists c_1 \in \mathbb{R}^{+*}, \exists c_2 \in \mathbb{R}^{+*}, \exists n_0 \in \mathbb{N}$
$\forall n \geq n_0, \; c_1 . f(n) \leq g(n) \leq c_2 . f(n) \}$



$g(n)$ est asymptotiquement équivalent à $f(n)$

* $O(f(n)) = \{ g(n) \mid \exists c_2 \in \mathbb{R}^{+*}, \exists n_0 \in \mathbb{N}$
$\forall n \geq n_0, \; g(n) \leq c_2 f(n) \}$



$\sqrt{n} \in O(n)$

$g(n)$ est asymptotiquement dominée par $f(n)$.

* $\Omega(f(n)) = \{ g(n) \mid \exists c_1 \in \mathbb{R}^{+*}, \exists n_0 \in \mathbb{N},$
$\forall n \geq n_0, \; c_1 f(n) \leq g(n) \}$

$g(n)$ domine asymptotiquement $f(n)$.

* $\Theta(f(n)) \subset O(f(n))$ $\quad \Theta(f(n)) \subset \Omega(f(n))$ $\quad O(f(n)) \cap \Omega(f(n)) = \Theta(f(n))$

* $\lim_{n \to +\infty} \frac{f(n)}{g(n)} = 0 \iff f(n) = o(g(n))$

$$\iff \begin{cases} f(n) = O(g(n)) \\ f(n) \neq \Theta(g(n)) \end{cases}$$

* $\lim_{n \to +\infty} \frac{f(n)}{g(n)} = \infty \iff f(n) = \omega(g(n))$

$$\iff \begin{cases} f(n) = \Omega(g(n)) \\ f(n) \neq \Theta(g(n)) \end{cases}$$

$$\iff \begin{cases} g(n) = O(f(n)) \\ g(n) \neq \Theta(f(n)) \end{cases}$$

* $\lim_{n \to +\infty} \frac{f(n)}{g(n)} = c \neq 0 \implies f(n) = \Theta(g(n))$

$c \in \mathbb{R}^{+*}$

$\Theta(-)$ complexité <u>exacte</u> d'une ligne ou d'un algo entier

$O(-)$ complexité <u>au pire</u>

$\Omega(-)$ complexité <u>au mieux</u>

<u>Théorème général</u>: pour les équations de complexité récursives de type :

$$\begin{cases} T(n) = a\,T(n/b) + f(n) \\ T(1) = \Theta(n) \end{cases} \quad \text{avec} \quad \begin{matrix} a \geq 1, \\ b > 1 \end{matrix}$$

Trois cas sont considérés :

1) si $f(n) = O(n^{\log_b a - \varepsilon})$ avec $\varepsilon > 0$.
   alors $T(n) = \Theta(n^{\log_b a})$
2) si $f(n) = \Theta(n^{\log_b a})$ alors $T(n) = \Theta(n^{\log_b a} \log n)$
3) si $f(n) = \Omega(n^{\log_b a + \varepsilon})$ avec $\varepsilon > 0$
   et si de plus il existe $c < 1$ tq. $a f(n/b) \leq c f(n)$
   alors $T(n) = \Theta(f(n))$

⚠ le th. ne couvre pas tous les cas possibles.