Université Lille 1 *IEEA*

Rapport de Travaux Pratiques

Algorithmes et ComplexiTé (ACT)
TP1 : Ordres de grandeur asymptotiques

Auteur :

Salla DIAGNE
Anis TELLO

Professeur :

Sophie TISON

17 Septembre 2014

Listing des fichiers

- `casn.txt` avec (n ∈ [0..6]) : les fichiers générés par l'exécution du code Java
- `Mesure.java` et `Methodes.java` les fichiers source requis pour la génération de tous les fichiers de données
- `tpog.plot` : unique fichier de commande gnuplot permettant la génération de tous les graphiques en une seule exécution

1 Gnuplot

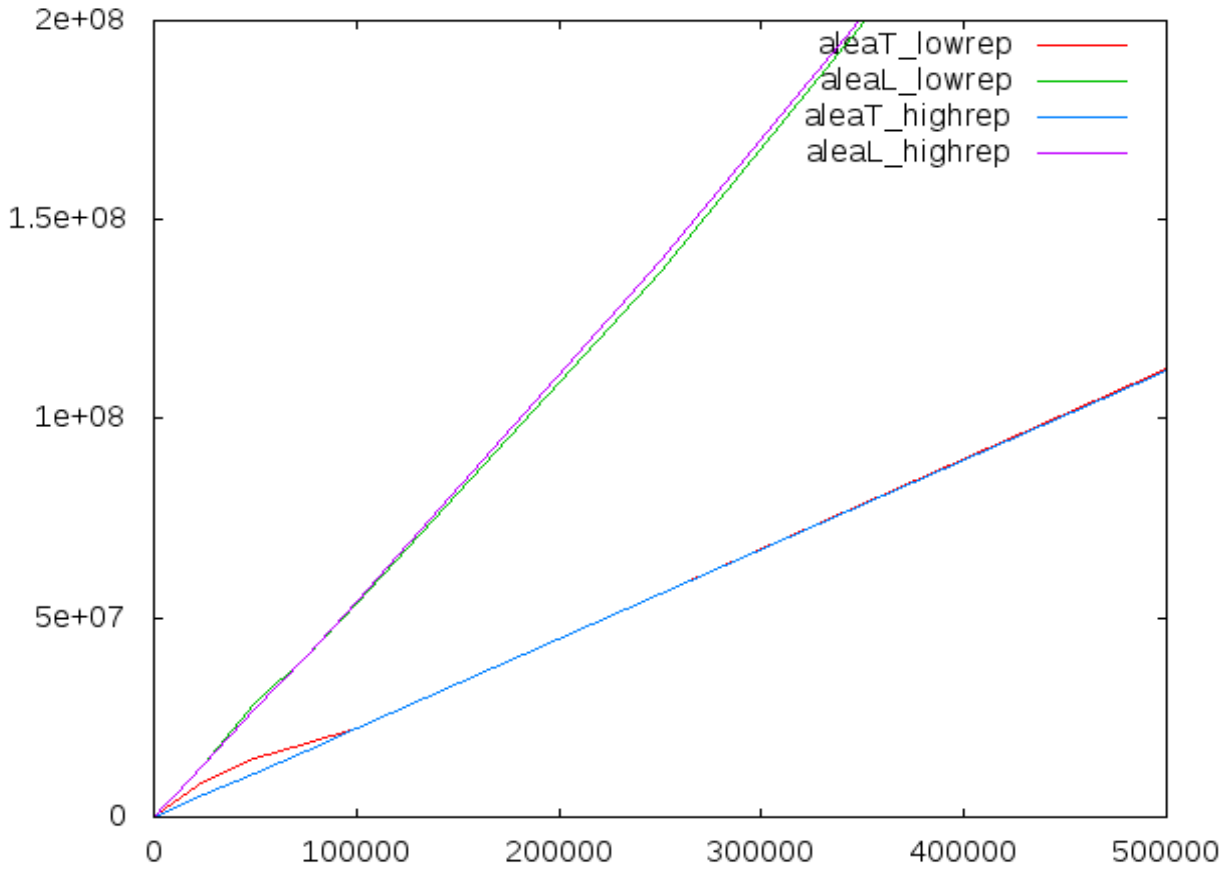
Voir les fichiers :

- `fichier.txt`
- `graphique.png`
- `tpog.plot`

2 Comparaisons de méthodes Java et d'algorithmes fait maison

2.1 Influence sur les données

Q1.



Nous constatons que la méthode `aleaT` est plus rapide que la méthode `aleaL`, quelque soit le nombre de répétitions.

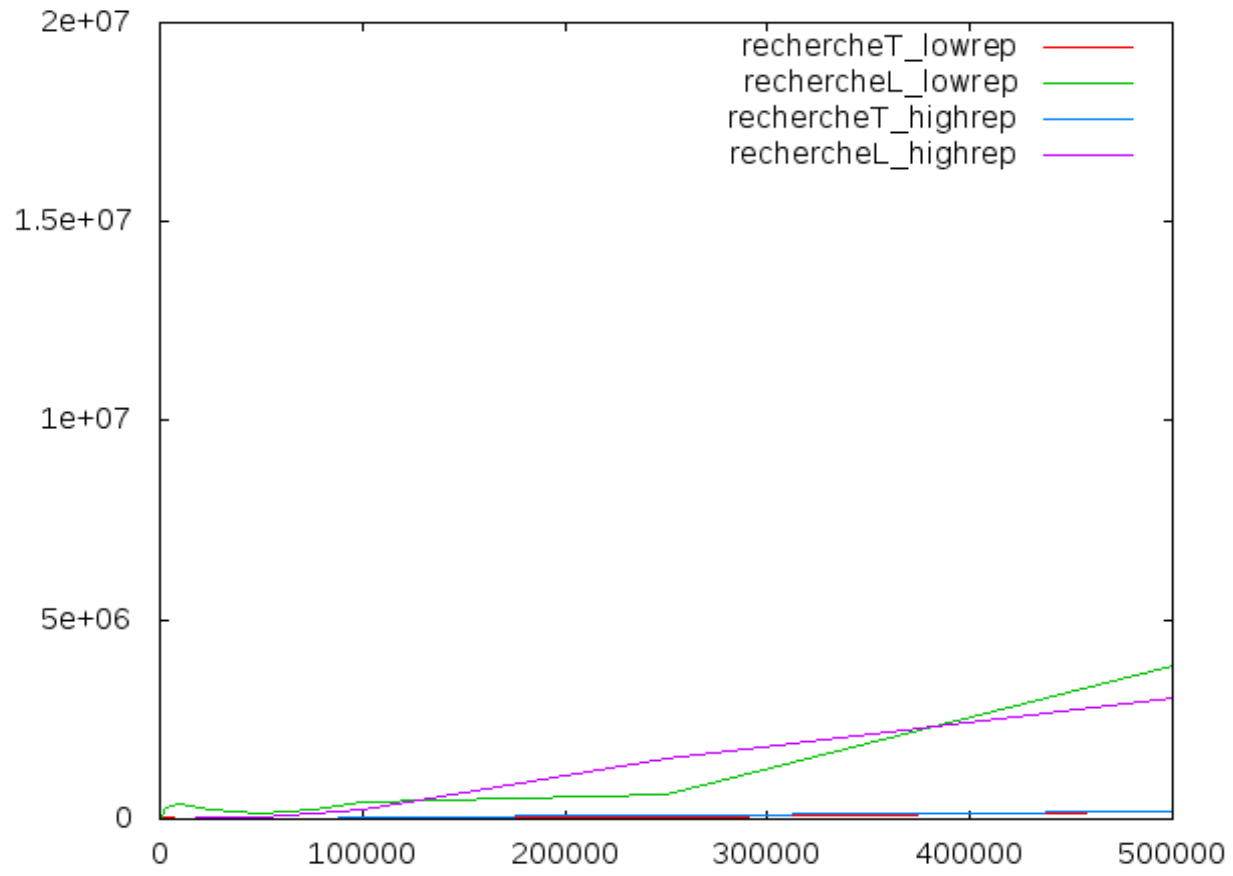
Q2.

Le nombre de répétitions n'a quasiment pas d'influence sur les courbes.

Q3.

//TODO

Q4.



Nous constatons que la méthode `rechercheT` est plus rapide que la méthode `rechercheL`, quelque soit le nombre de répétitions.

Q5.

Le nombre de répétitions a une petite influence sur les courbes.

Q6.

Sur les huit mesures précédentes, nous constatons que les opérations sur les tableaux sont plus rapides que sur les listes.

Q7.

Les variations d'influence du nombre de répétitions de la mesure s'explique par la variation de la complexité des deux algorithmes :

- `alea()` : place des entiers de 0 à size - 1 dans un ordre aléatoire
- `recherche(int e)` : recherche l'entier e dans la structure de données

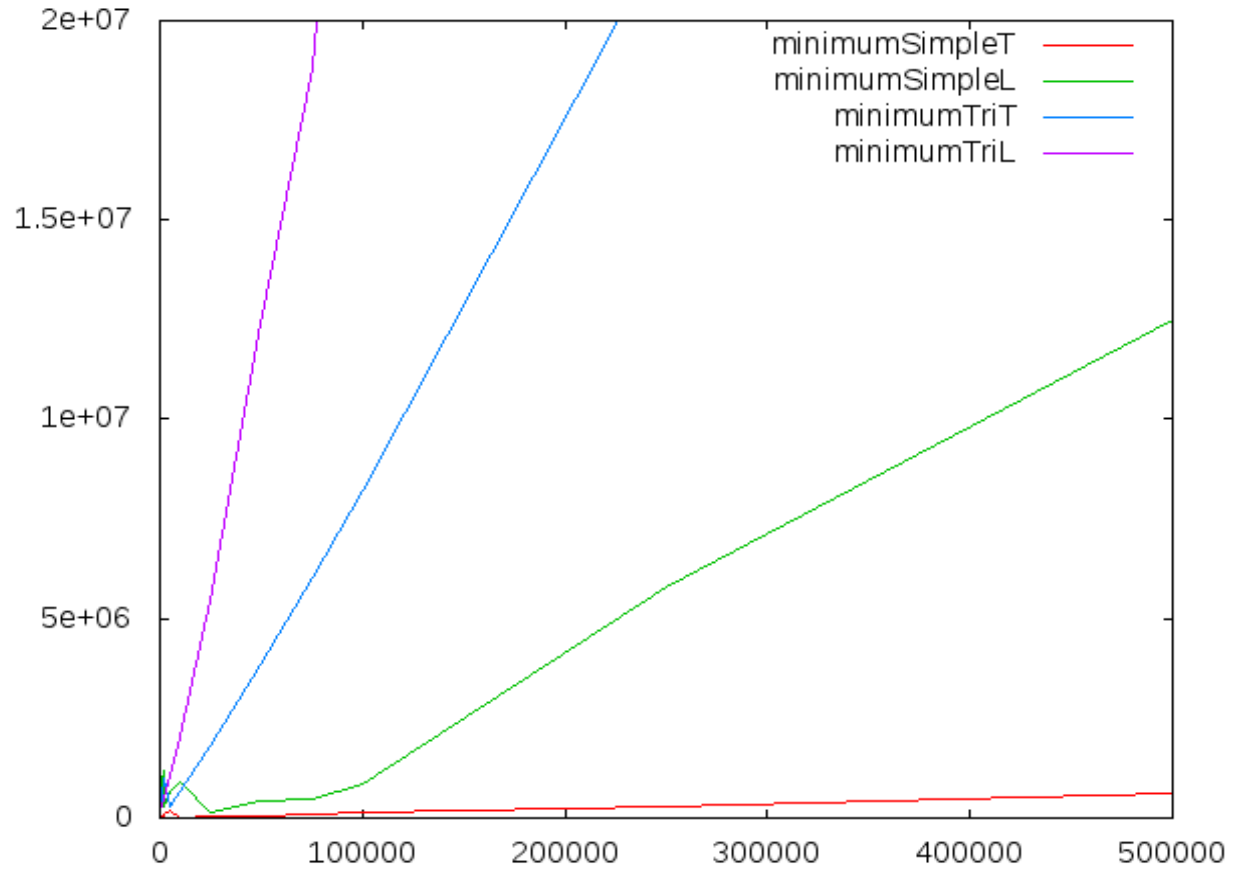
La complexité d'`alea` est constante, alors que celle de `recherche` dépend de la disposition des entiers dans la structure. Voilà pourquoi la répétition a une influence dans `recherche` et n'en a pas dans `alea`.

Q8.

Nous pouvons conclure qu'il faut choisir un nombre faible de répétitions (même si ce nombre n'a pas d'influence quand la complexité de l'algorithme étudié est constante.

2.2 Recherches de minima

Q9.



Nous constatons que les méthodes `minimumSimpleT` et `minimumTriT` sont plus rapides que les méthodes `minimumSimpleL` et `minimumTriL`, quelque soit le nombre de répétitions.

Q10.

Nous pouvons dire que le nombre de répétitions fait beaucoup varier l'efficacité de ces deux structures de données.

Q11.

Nous pouvons dire qu'il est plus efficace de ne pas trier le tableau ou la liste avant d'en rechercher le minimum.

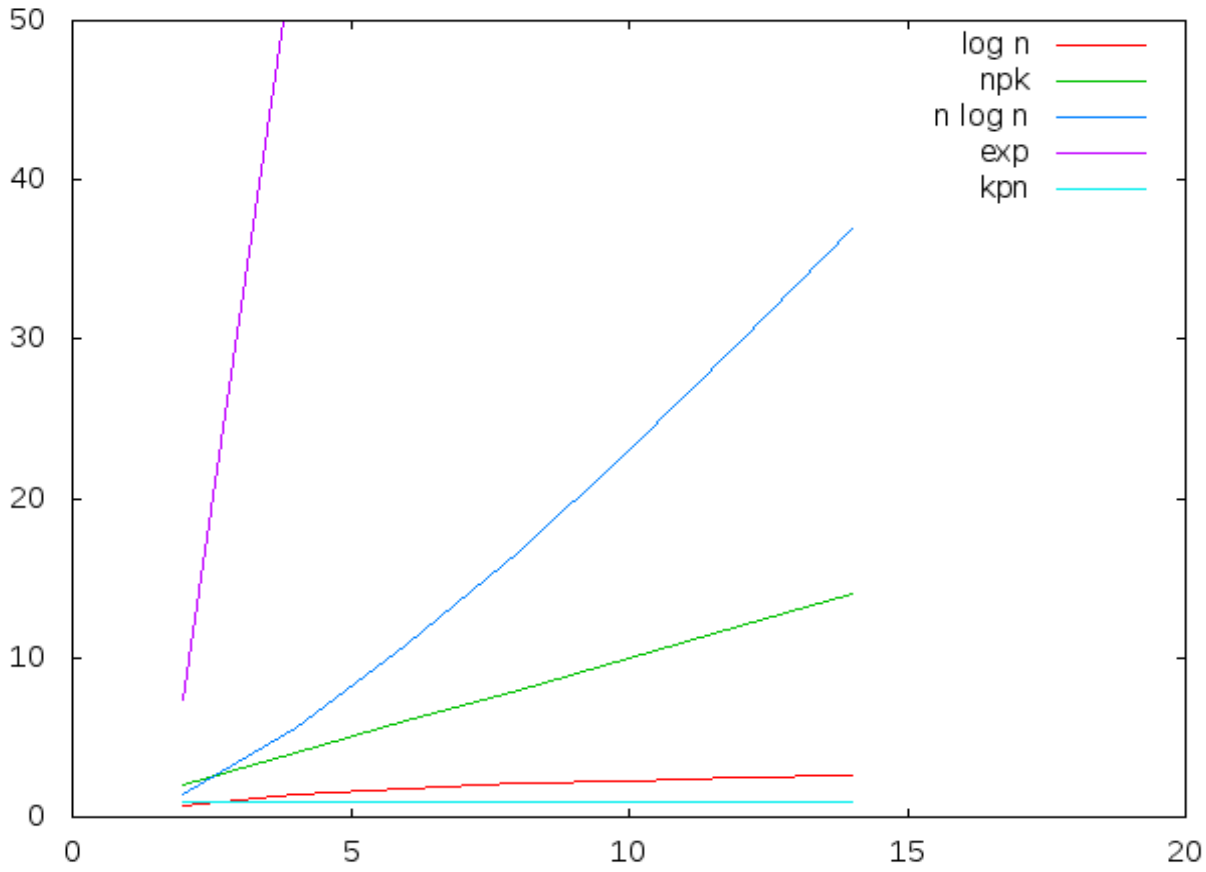
3 Comparaisons des fonctions de référence

3.1 Vue d'ensemble

Q12.

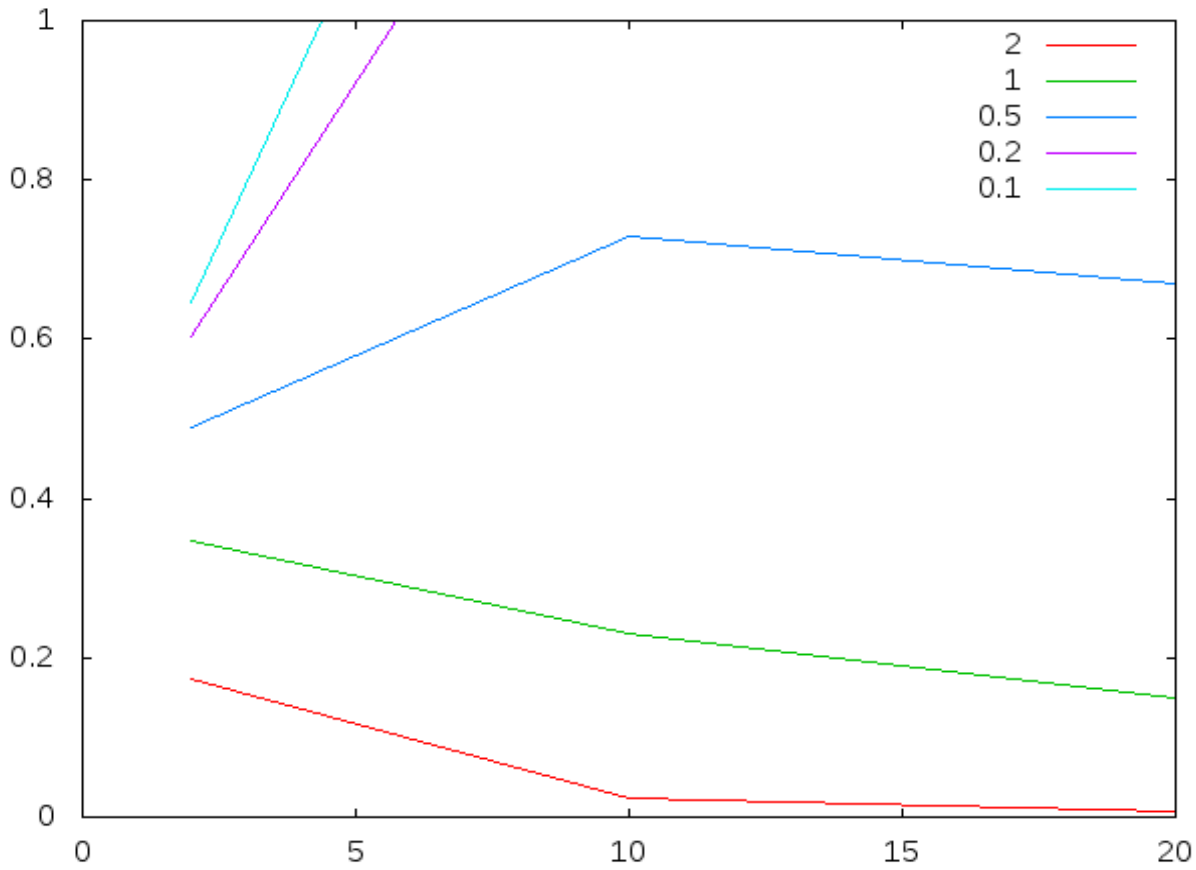
Cela n'a pas été facile. Il a fallu adapter notre tableau de tailles pour ne pas avoir des valeurs infinies et l'échelle de notre graphique pour voir les courbes.

Q13.



Nous pouvons déduire que la fonction exponentielle croît beaucoup plus vite que les autres fonctions de référence.

Q14.



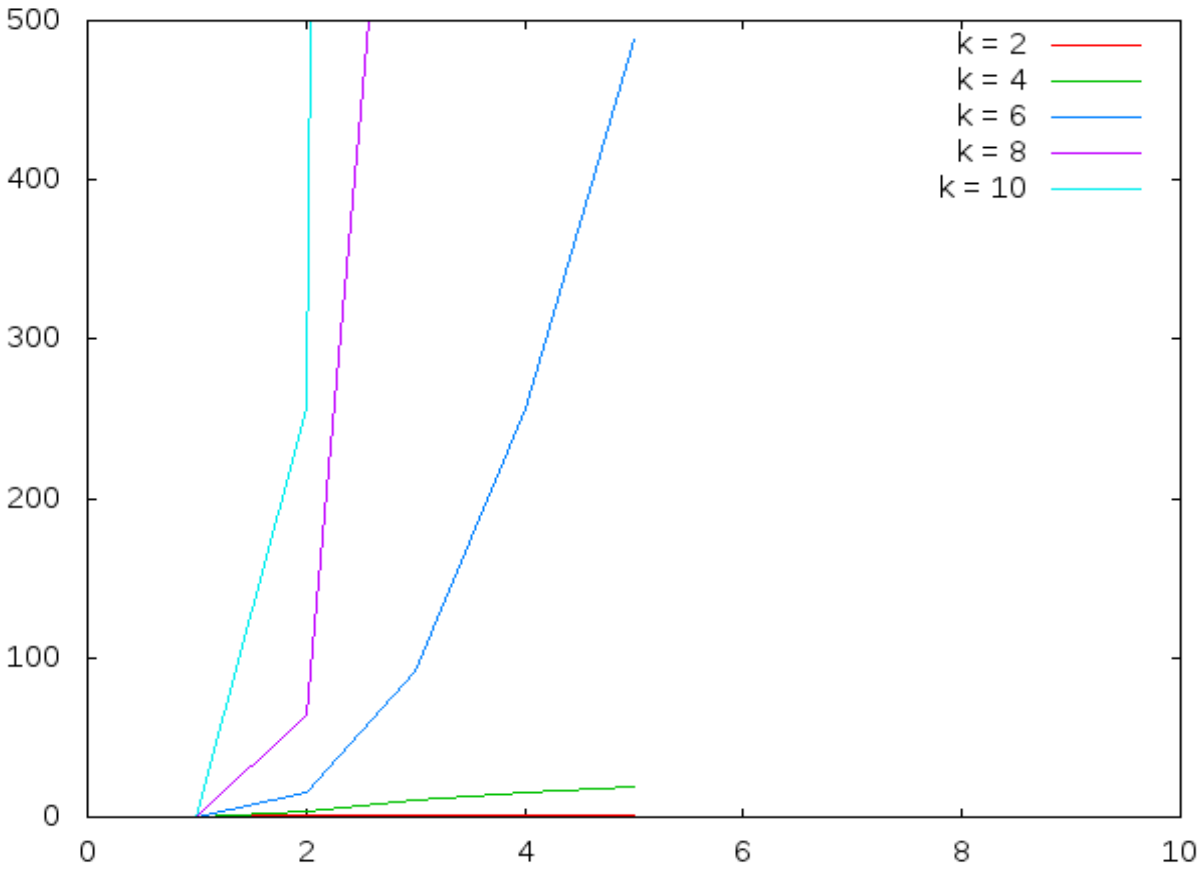
Q15.

Plus ε est grand, plus $\frac{logn}{n^\varepsilon}$ est petit.

$=> \lim_{x \to +\infty} (\frac{logn}{n^\varepsilon}) = 0$

$=> \boxed{logn = \mathrm{O}(n^\varepsilon) \ (\varepsilon > 0)}$

Q16.



Q17.

$\boxed{n^k = \Omega(2^n)}$

4 Détermination expérimentale de la complexité d'un algorithme

Q18.

L'algorithme codé par les deux méthodes mystères est une méthode de tri basée sur des renversements de portions de tableaux/listes.

Q19.

Donnons le pseudo-code de cet algorithme :

```
   soit t la structure à trier et n sa longueur
pour i allant de 0 à n - 1 faire
soit p la position du minimum dans t[i..n-1]
renverser les éléments de t entre les positions p et n - 1
(t[n-1] contient l'élément minimal de t[i..n-1])
renverser les éléments de t entre les positions i et n - 1
(t[i] contient l'élément minimal de t[i..n-1])
fin pour
```

Chaque tour de boucle est caractérisé par 3 étapes : la recherche du minimum, un renversement, et un autre renversement

Ses cas extrêmes sont :

Meilleur des cas : à chaque tour de boucle, l'élément minimum se trouve en dernière position. Le premier renversement ne fera donc rien.

Pire des cas : la structure est triée par ordre croissant. Ainsi, à chaque tour de boucle, l'indice de boucle est l'indice du minimum de la structure. Le premier renversement se fait donc toujours sur toute la partie du tableau structure (comme le deuxième).

Cet algorithme est en $\Theta(n^2)$.