

DUT MMI – IUT de Marne-la-Vallée
28/01/2015
M2202 - Algorithmique

Cours 2

Tris

Résumé de l'épisode précédent

- **Récurtivité**

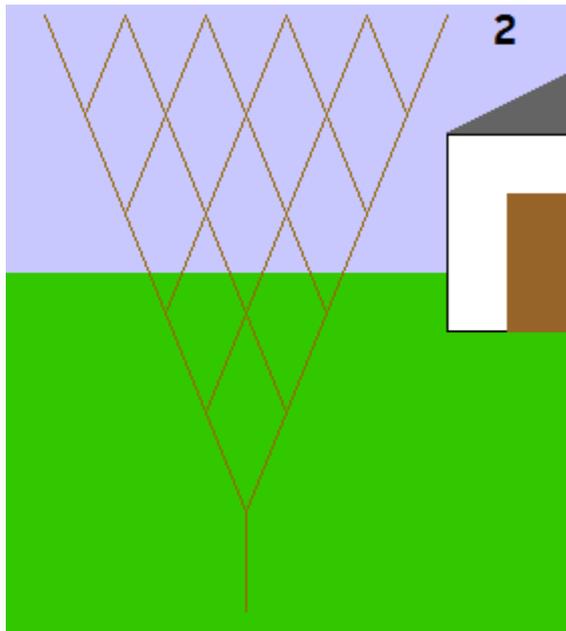
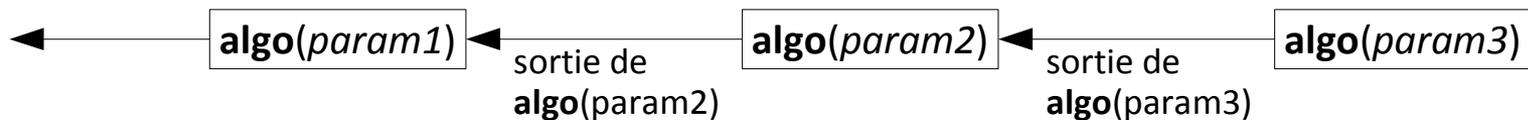
Un algorithme **récurusif** = un algorithme qui s'appelle lui-même

On suppose que :

- * on sait construire/calculer le premier objet (**initialisation**)

- * à partir du $(n-1)$ -ième objet on sait construire/calculer le n -ième (**hérédité**)

alors on arrive à construire/calculer tous les objets.



- **Tris**

Ranger un ensemble d'objets tous comparables deux à deux :

→ tri à bulles (étapes d'échanges successifs de gauche à droite)

→ tri par sélection

Sources

- Cours de Jean-François Berdjugin à l'IUT de Grenoble
<http://berdjugin.com/enseignements/inf/inf220/>
- Cours de Xavier Heurtebise à l'IUT de Provence
<http://x.heurtebise.free.fr>
- *Le livre de Java premier langage*, d'A. Tasso
- <http://xkcd.com>, <http://xkcd.free.fr>

Plan du cours 2 – Tris

- Les tris
- Le tri par sélection
- Complexité des tris

Plan du cours 2 – Tris

- Les tris
- Le tri par sélection
- Complexité des tris

Plan du cours 2 – Tris

- Les tris
- Le tri par sélection
- Complexité des tris

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :

5	3	1	8	5	2	9
---	---	----------	---	---	---	---

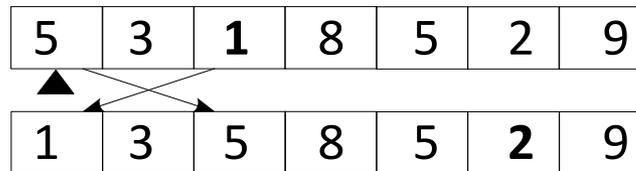


i vaut 1

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

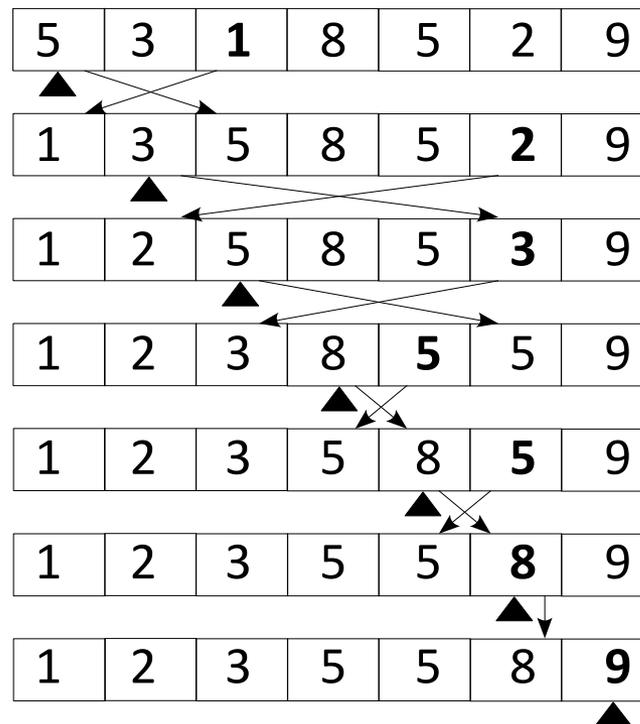
Exemple avec un tableau d'entiers :



Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :



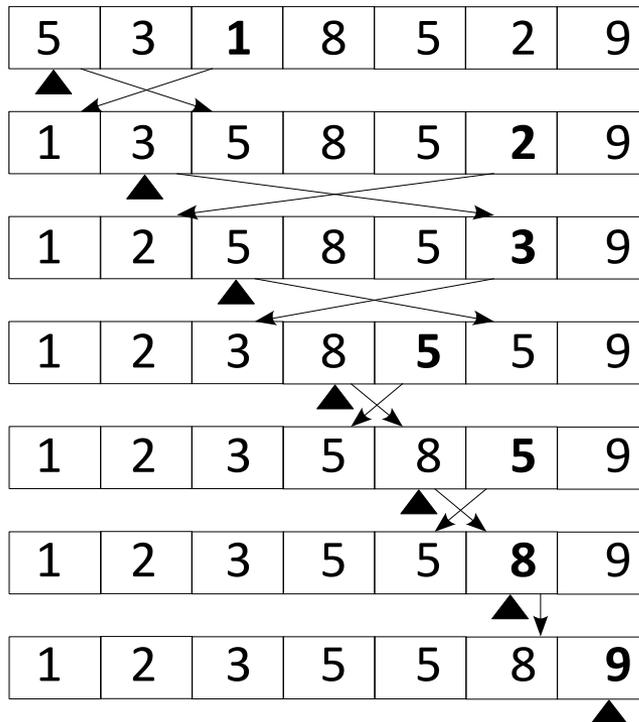
Tri en place :

on arrive à trier le tableau sans avoir besoin de créer un nouveau tableau.

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :



Algorithme **positionMinimum**

Entrée : tableau d'entiers *tab*, entier *debut*

Type de sortie : entier

Variables : entiers *position*, *i* et *min*

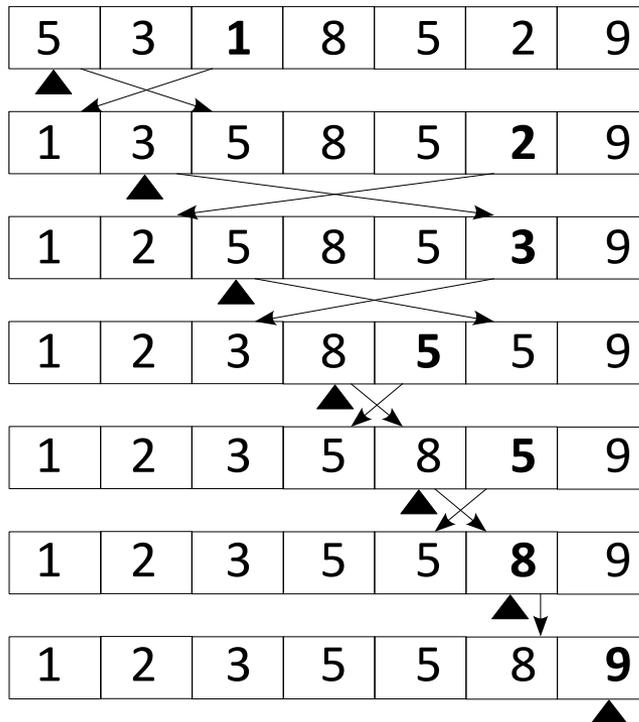
Début

Fin

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :



Algorithme **positionMinimum**

Entrée : tableau d'entiers tab , entier $debut$

Type de sortie : entier

Variables : entiers $position$, i et min

Début

$position \leftarrow debut$

$min \leftarrow \mathbf{Case}(tab,debut)$

$i \leftarrow debut$

Tant que $i \leq \mathbf{Longueur}(tab)$ faire :

Si $\mathbf{Case}(tab,i) < min$ alors :

$position \leftarrow i$

$min \leftarrow \mathbf{Case}(tab,i)$

Fin si

$i \leftarrow i+1$

Fin Tant que

renvoyer $position$

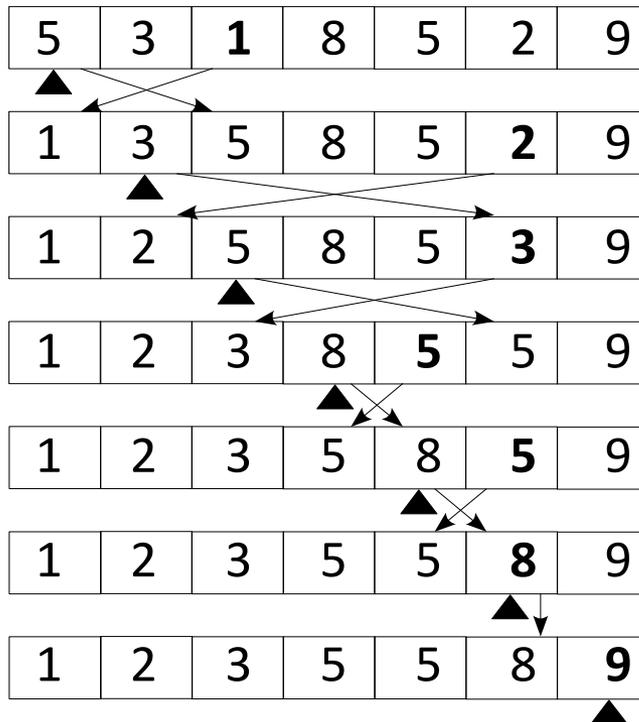
Fin

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :

version récursive vue en cours !



Algorithme **positionMinimum**

Entrée : tableau d'entiers *tab*, entier *debut*

Type de sortie : entier

Variables : entier *position*

Début

Si $debut = \text{Longueur}(tab)$ alors :

renvoyer *debut*

Sinon :

$position \leftarrow \text{positionMinimum}(tab, debut+1)$

Si $\text{Case}(tab, debut) < \text{Case}(tab, position)$ alors :

renvoyer *debut*

Sinon :

renvoyer *position*

Fin si

Fin si

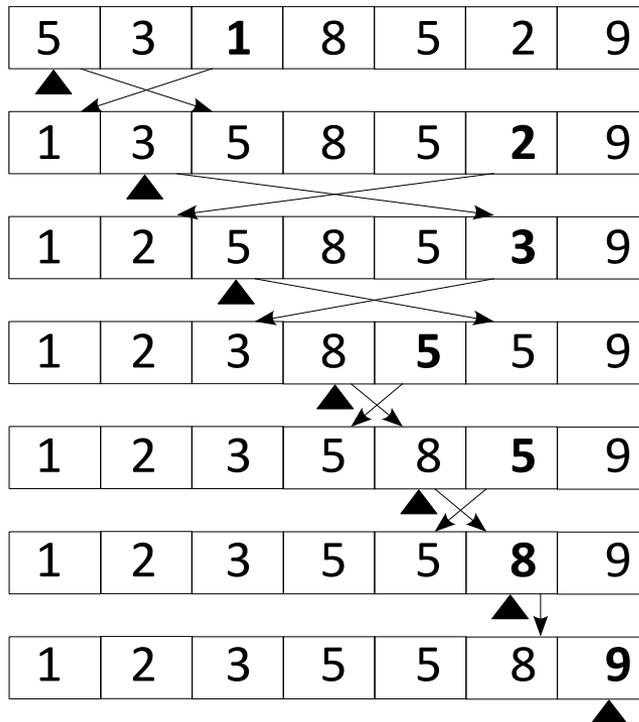
Fin

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :

version récursive plus courte !



Algorithme **positionMinimum**

Entrée : tableau d'entiers *tab*, entier *debut*

Type de sortie : entier

Variables : entier *position*

Début

$position \leftarrow debut$

Si $debut < \text{Longueur}(tab)$ alors :

$position \leftarrow \text{positionMinimum}(tab, debut+1)$

Si $\text{Case}(tab, debut) < \text{Case}(tab, position)$ alors :

$position \leftarrow debut$

Fin si

Fin si

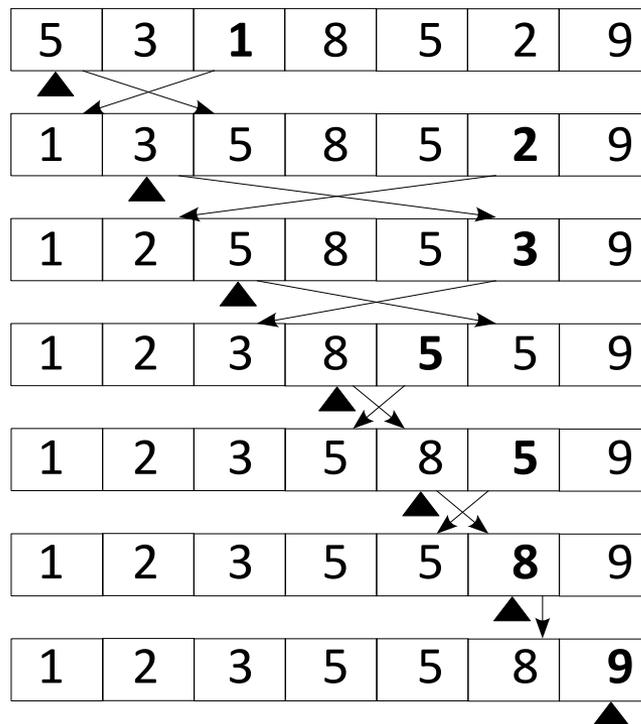
renvoyer *position*

Fin

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :



Algorithme **triSelection**

Entrée : tableau d'entiers *tab*

Type de sortie : tableau d'entiers

Variables : entiers *i*, *temp*, *posMin*

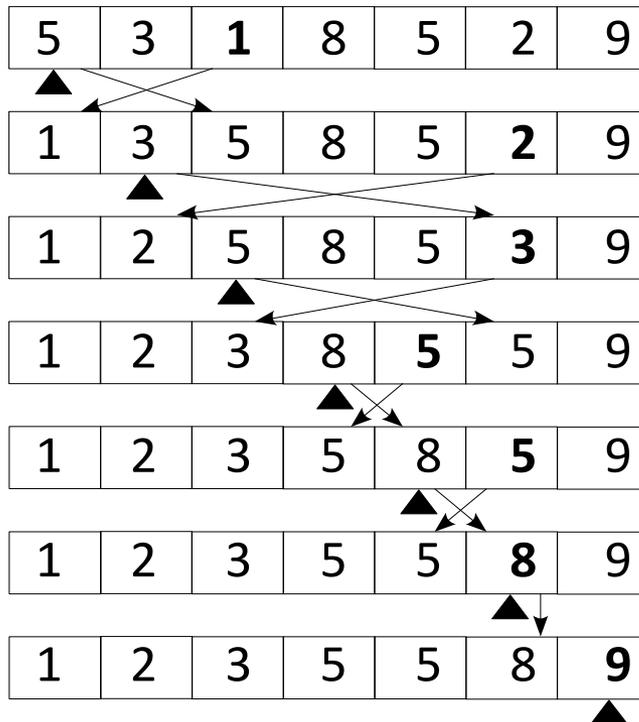
Début

Fin

Tri par sélection (ou tri par extraction)

Idée : à la i -ième étape, on prend le plus petit élément à partir de la i -ième case (comprise) et on l'échange avec l'élément de la i -ième case.

Exemple avec un tableau d'entiers :



Algorithme **triSelection**

Entrée : tableau d'entiers tab

Type de sortie : tableau d'entiers

Variables : entiers i , $temp$, $posMin$

Début

$i \leftarrow 1$

Tant que $i < \text{Longueur}(tab)$ faire :

$posMin \leftarrow \text{positionMinimum}(tab, i)$

// $posMin$ peut être égal à i

$temp \leftarrow \text{Case}(tab, i)$

$\text{Case}(tab, i) \leftarrow \text{Case}(tab, posMin)$

$\text{Case}(tab, posMin) \leftarrow temp$

$i \leftarrow i + 1$

Fin Tant que

renvoyer tab

Fin

Plan du cours 2 – Tris

- Les tris
- Le tri par sélection
- Complexité des tris

Complexité des tris

Combien de comparaisons ? (lié au **temps d'exécution** de l'algorithme)

Sur un exemple ou dans le pire des cas, pour n éléments.

Tri à bulles

Tri par sélection

Complexité des tris

Combien de comparaisons ? (lié au **temps d'exécution** de l'algorithme)

Sur un exemple ou dans le pire des cas, pour n éléments.

Tri à bulles

Dans tous les cas : $\leq n$ étapes, $n-1$ comparaisons par étape
donc $\leq n \times (n-1)$ comparaisons au total

Tri par sélection

Pour n éléments : n étapes, $n-i$ comparaisons par étape pour trouver le min

Complexité des tris

Combien de comparaisons ? (lié au **temps d'exécution** de l'algorithme)

Sur un exemple ou dans le pire des cas, pour n éléments.

Tri à bulles

Dans tous les cas : $\leq n$ étapes, $n-1$ comparaisons par étape
donc $\leq n \times (n-1)$ comparaisons au total

Tri par sélection

Pour n éléments : n étapes, $n-i$ comparaisons par étape pour trouver le min

étape 1 : $n-1$ comparaisons

étape 2 : $n-2$ comparaisons

...

étape $n-2$: 2 comparaisons

étape $n-1$: 1 comparaison

étape n : 0 comparaison

Complexité des tris

Combien de comparaisons ? (lié au **temps d'exécution** de l'algorithme)

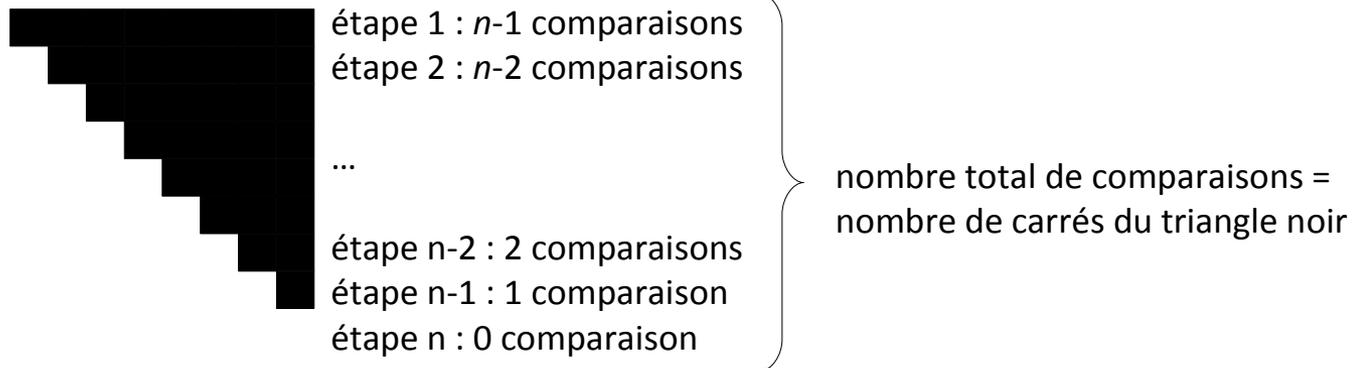
Sur un exemple ou dans le pire des cas, pour n éléments.

Tri à bulles

Dans tous les cas : $\leq n$ étapes, $n-1$ comparaisons par étape
donc $\leq n \times (n-1)$ comparaisons au total

Tri par sélection

Pour n éléments : n étapes, $n-i$ comparaisons par étape pour trouver le min



Complexité des tris

Combien de comparaisons ? (lié au **temps d'exécution** de l'algorithme)

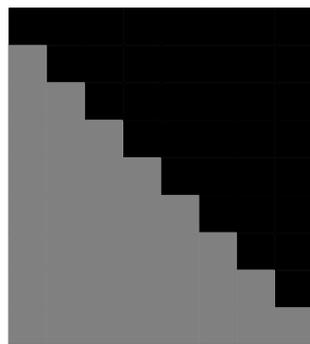
Sur un exemple ou dans le pire des cas, pour n éléments.

Tri à bulles

Dans tous les cas : $\leq n$ étapes, $n-1$ comparaisons par étape
donc $\leq n \times (n-1)$ comparaisons au total

Tri par sélection

Pour n éléments : n étapes, $n-i$ comparaisons par étape pour trouver le min



étape 1 : $n-1$ comparaisons

étape 2 : $n-2$ comparaisons

...

étape $n-2$: 2 comparaisons

étape $n-1$: 1 comparaison

étape n : 0 comparaison

nombre total de comparaisons =
nombre de carrés du triangle noir =
nombre de carrés du rectangle divisé par 2 =

Complexité des tris

Combien de comparaisons ? (lié au **temps d'exécution** de l'algorithme)

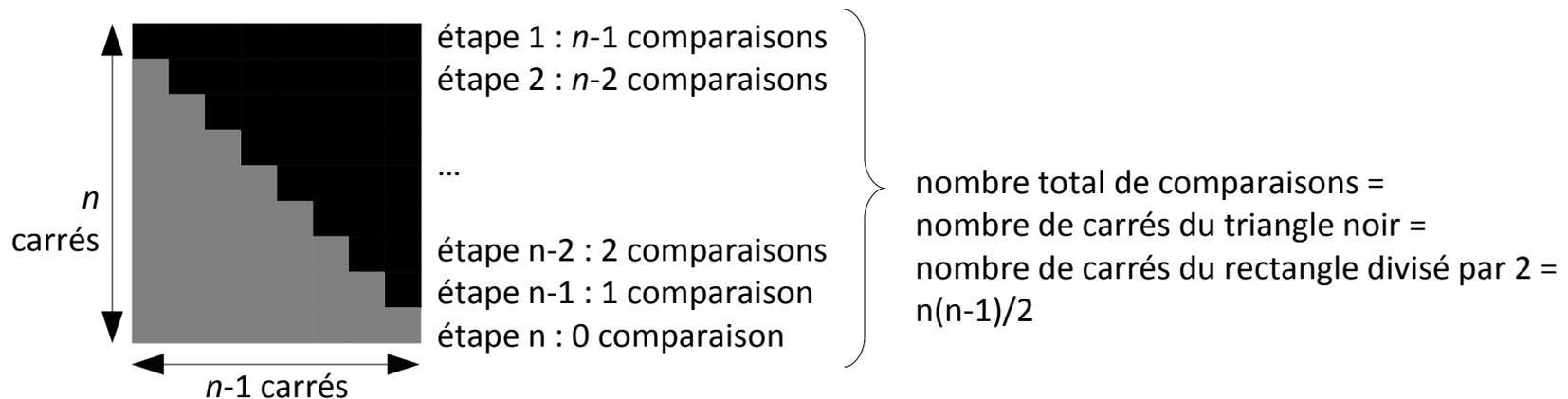
Sur un exemple ou dans le pire des cas, pour n éléments.

Tri à bulles

Dans tous les cas : $\leq n$ étapes, $n-1$ comparaisons par étape
donc $\leq n \times (n-1)$ comparaisons au total

Tri par sélection

Pour n éléments : n étapes, $n-i$ comparaisons par étape pour trouver le min



Complexité des tris

Combien de comparaisons ? (lié au **temps d'exécution** de l'algorithme)

Sur un exemple ou dans le pire des cas, pour n éléments.

Tri à bulles

Dans le pire des cas (tableau trié dans le sens inverse) : $n \times (n-1)$ comparaisons

Dans le meilleur cas (tableau trié) : $n-1$ comparaisons

Tri par sélection

Dans tous les cas : $n \times (n-1)/2$ comparaisons