

DUT MMI – IUT de Marne-la-Vallée
04/02/2019
M2202 - Algorithmique

Cours 2

Tris

Résumé du cours / TD précédent : récursivité

Exemple des factorielles :

- **initialisation** : la première factorielle est **1**
- **hérédité** : pour passer de factorielle($n-1$) à factorielle(n), je multiplie par n

Algorithme **factorielle**

Entrées : entier n

Type de sortie : entier

Variable : entier *resultat*

Début

Si $n=1$ alors :

$resultat \leftarrow 1$

sinon :

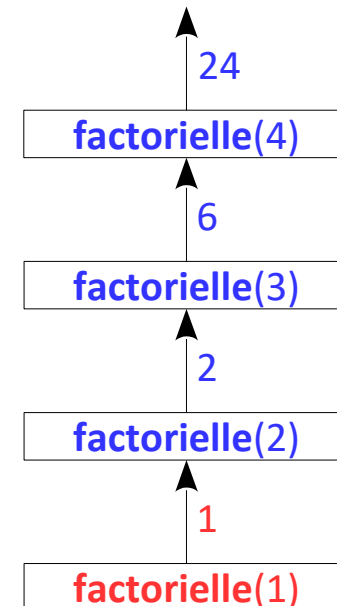
$resultat \leftarrow n \times \text{factorielle}(n-1)$

Fin si

renvoyer *resultat*

Fin

Trace de **factorielle(4)** :



Plan du cours 2 – Tris

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

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Ensemble d'éléments : quel type ?

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Ensemble d'éléments : quel type ?

Type pour stocker plusieurs éléments ?

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Ensemble d'éléments : quel type ?

Type pour stocker plusieurs éléments ?

→ un tableau !

→ autres structures de données au prochain cours.

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Ensemble d'éléments	Ordre	Tous comparables deux à deux ?
tableau d'entiers	plus petit	
tableau de chaînes de caractères	lexicographique (du dictionnaire)	
tableau de cartes à jouer	$2 \leq 3 \leq \dots \leq 10 \leq$ $V \leq D \leq R \leq 1$	
tableau de cartes à jouer	$\clubsuit \leq \diamondsuit \leq \heartsuit \leq \spadesuit$ puis $2 \leq 3 \leq \dots \leq 10 \leq$ $V \leq D \leq R \leq 1$	

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Ensemble d'éléments	Ordre	Exemples
tableau d'entiers	plus petit	$1 \leq 2, 2 \leq 3, 1 \leq 3...$
tableau de chaînes de caractères	lexicographique (du dictionnaire)	$a \leq b, \text{char} \leq \text{charles}, \text{char} \leq \text{chat}...$
tableau de cartes à jouer	$2 \leq 3 \leq \dots \leq 10 \leq V \leq D \leq R \leq 1$	$5 \clubsuit \leq 5 \heartsuit$
tableau de cartes à jouer	$\clubsuit \leq \diamondsuit \leq \heartsuit \leq \spadesuit$ puis $2 \leq 3 \leq \dots \leq 10 \leq V \leq D \leq R \leq 1$	$A \clubsuit \leq 5 \heartsuit, 5 \heartsuit \leq 6 \heartsuit, \dots$

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Exemple avec un tableau d'entiers :

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



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

En anglais : trié = **sorted**
trier = **to sort**

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Exemple avec un tableau d'entiers :

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



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

En anglais : trié = sorted
trier = to sort

Algorithme **inferieurOuEgal**

Entrées : ?

Type de sortie : ?

Exemple : **inferieurOuEgal**(5,8) renvoie ...

Principe d'un algorithme de tri

Entrée :

Ensemble d'éléments **tous comparables deux à deux** par un ordre \leq

Sortie :

Éléments **triés selon cet ordre**, du plus petit au plus grand

Exemple avec un tableau d'entiers :

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



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

En anglais : trié = sorted
trier = to sort

Algorithme **inferieurOuEgal**

Entrées : deux entiers a et b

Type de sortie : booléen

Exemple : **inferieurOuEgal(5,8)** renvoie VRAI

Plan du cours 1 – Récursivité et tris

- Introduction à la récursivité
- Traces d'exécution de fonctions récursives
- Les tris
- Le tri par sélection
- Le tri à bulles
- 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)

La "minute culturelle"

Select-sort with Gypsy folk dance

AlgoRythmics



S'abonner

6 vidéos

facebook.com/AlgoRythmics

Intercultural Computer Science Education



J'aime



+ Ajouter à

Partager



98 228



Ajoutée par AlgoRythmics le 2 avril 2011

Created at Sapientia University, Tirgu Mures (Marosvásárhely), Romania.

443 aiment, 9 n'aiment pas

<http://www.youtube.com/watch?v=Ns4TPTC8whw>

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
---	---	----------	---	---	---	---

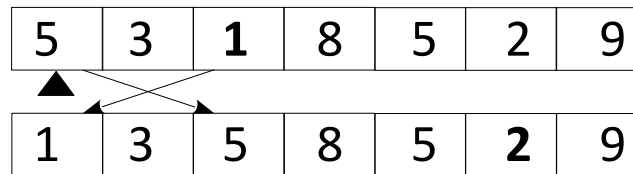


i vaut 1

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.

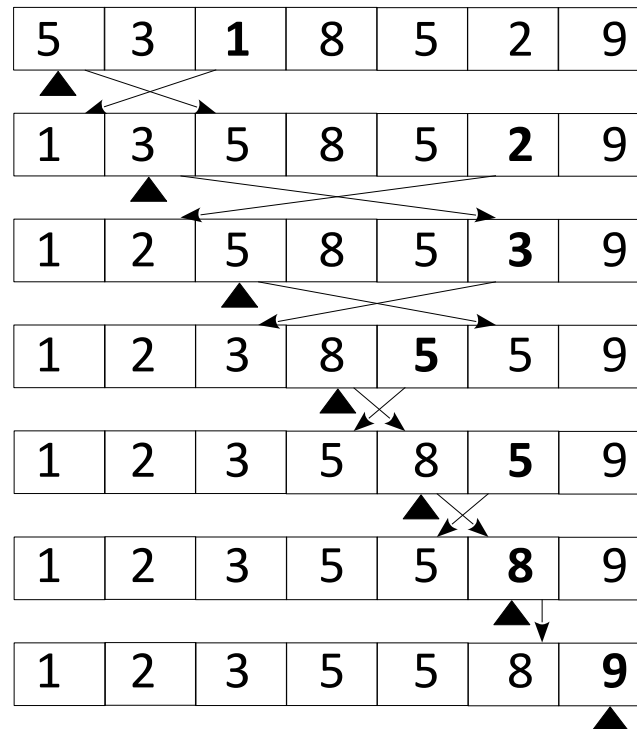
Exemple avec un tableau d'entiers :



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 :



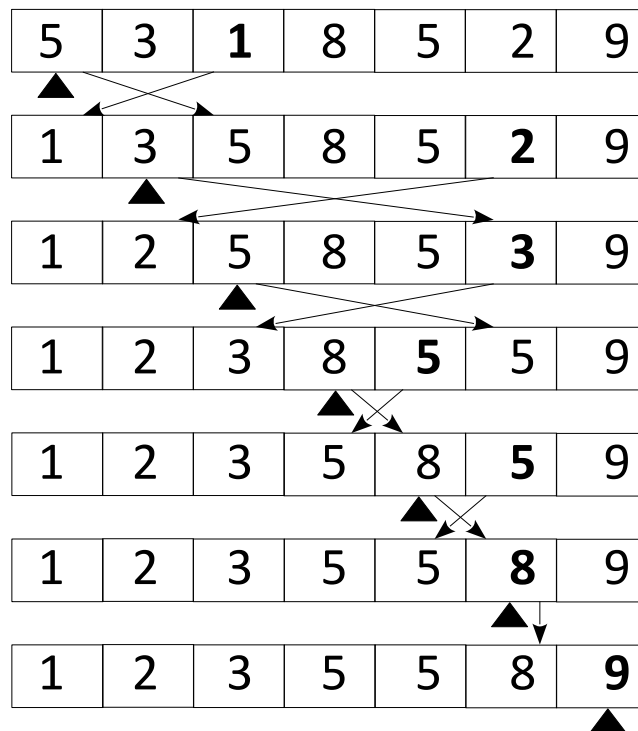
Tri en place :

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*

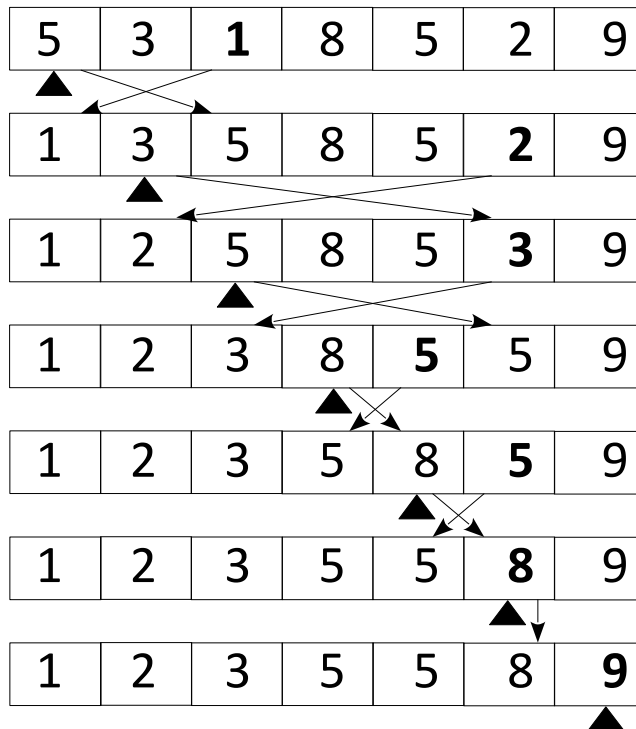
Début

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

$position \leftarrow debut$

$min \leftarrow \text{case}(tab, debut)$

$i \leftarrow debut$

Tant que $i \leq \text{longueur}(tab)$ faire :

Si $\text{case}(tab, i) < min$ alors :

$position \leftarrow i$

$min \leftarrow \text{case}(tab, i)$

Fin si

$i \leftarrow i+1$

Fin Tant que

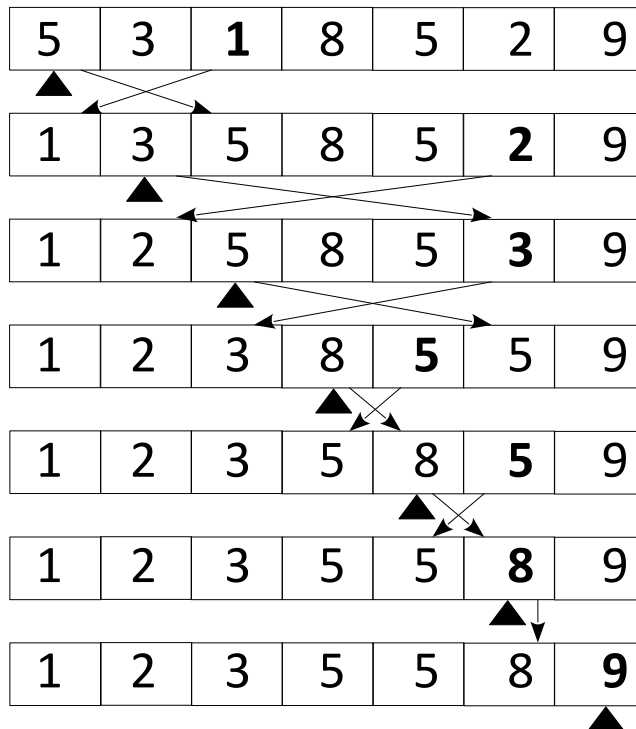
renvoyer $position$

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 :



Algorithme **triSelection**

Entrée : tableau d'entiers *tab*

Type de sortie : tableau d'entiers

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

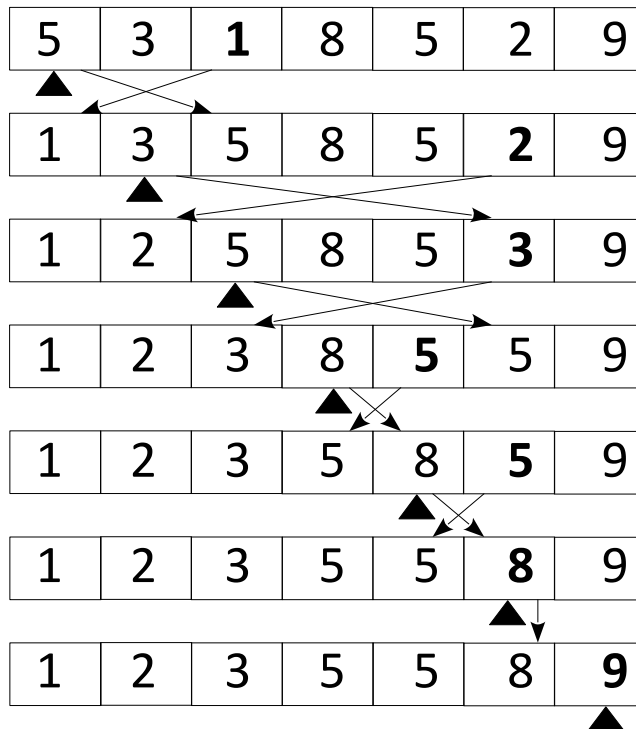
Début

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 1 – Récursivité et tris


- Introduction à la récursivité
- Traces d'exécution de fonctions récursives
- Les tris
- Le tri par sélection
- **Le tri à bulles**
- Complexité des tris

Tri à bulles

Idée : tant que le tableau n'est pas trié, on le parcourt en effectuant l'opération suivante à la i -ième case : si elle est supérieure à la suivante, on les échange.

Exemple avec un tableau d'entiers :

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



Tri à bulles

Idée : tant que le tableau n'est pas trié, on le parcourt en effectuant l'opération suivante à la i -ième case : si elle est supérieure à la suivante, on les échange.

Exemple avec un tableau d'entiers :

Étape 1 :

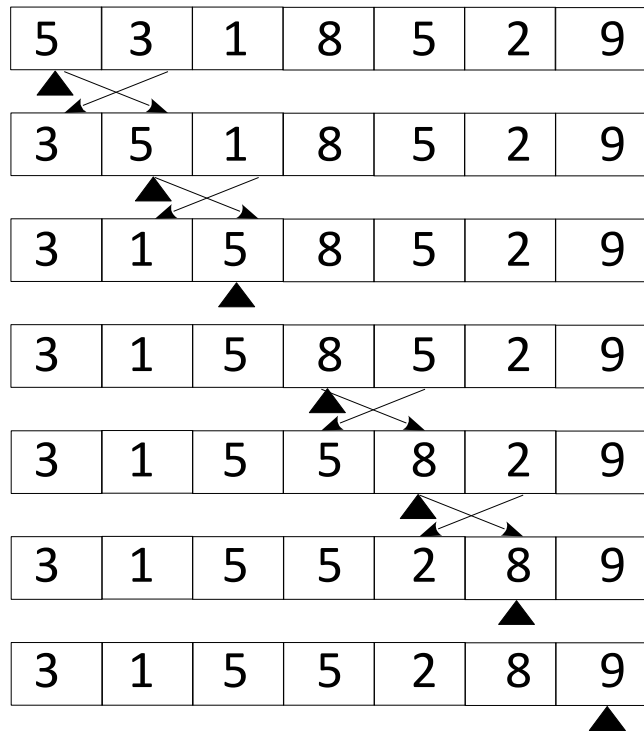


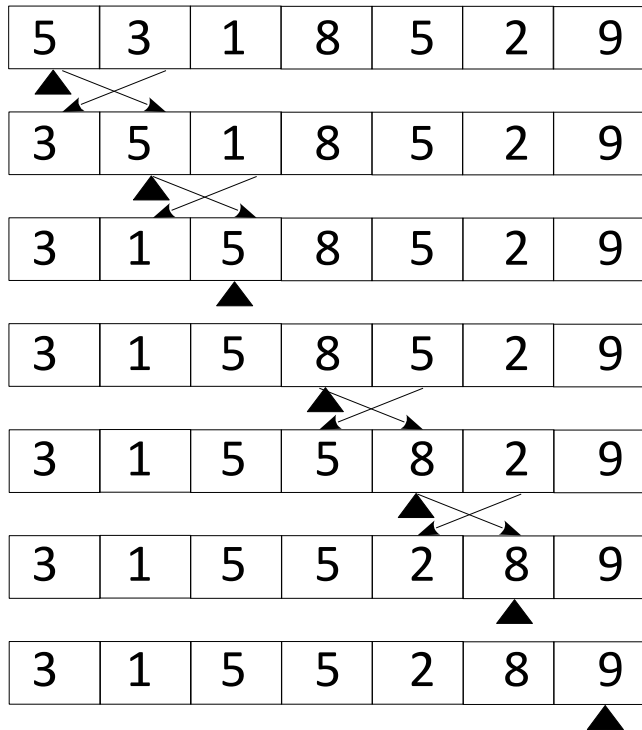
Tableau trié ?

Tri à bulles

Idée : tant que le tableau n'est pas trié, on le parcourt en effectuant l'opération suivante à la i -ième case : si elle est supérieure à la suivante, on les échange.

Exemple avec un tableau d'entiers :

Étape 1 :



Étape 2 :

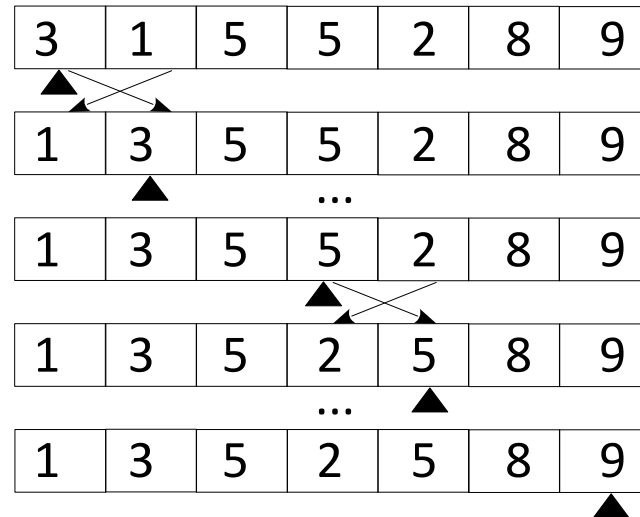


Tableau trié ?

Tri à bulles

Idée : tant que le tableau n'est pas trié, on le parcourt en effectuant l'opération suivante à la i -ième case : si elle est supérieure à la suivante, on les échange.

Exemple avec un tableau d'entiers :

Étape 2 :

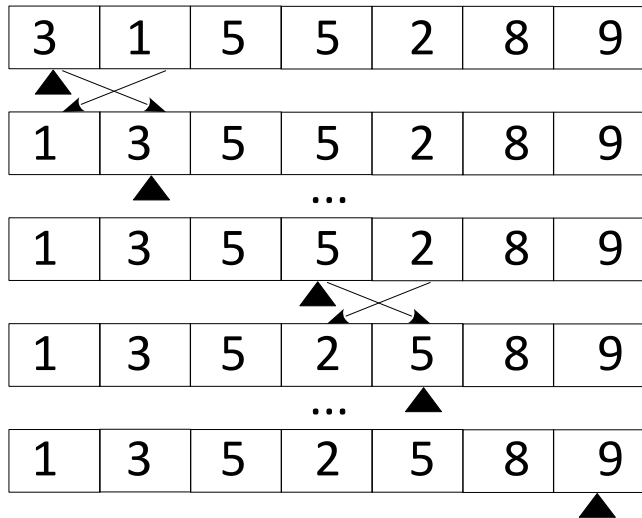
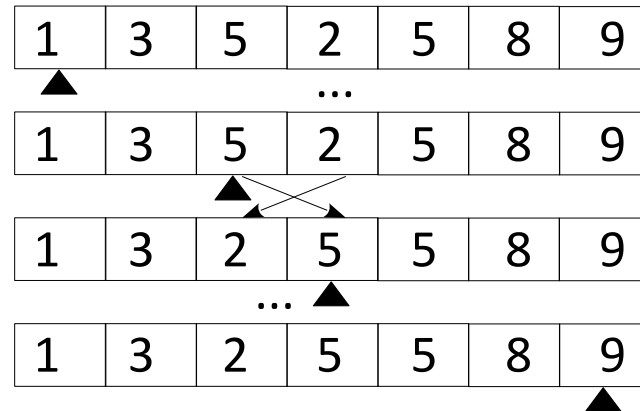


Tableau trié ?

Étape 3 :



Tri à bulles

Idée : tant que le tableau n'est pas trié, on le parcourt en effectuant l'opération suivante à la i -ième case : si elle est supérieure à la suivante, on les échange.

Exemple avec un tableau d'entiers :

Étape 3 :

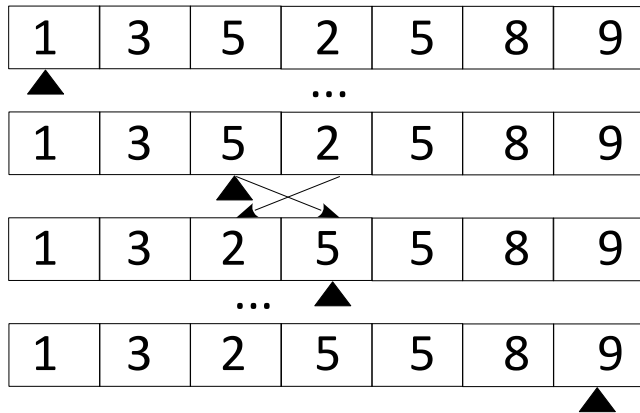


Tableau trié ?

Étape 4 :

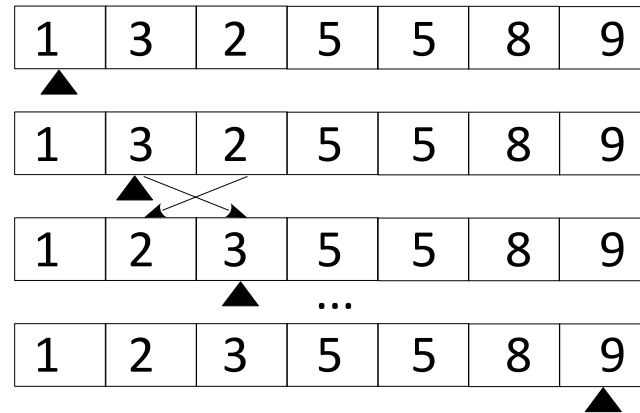


Tableau trié ?

Oui.

Plan du cours 1 – Récursivité et tris

- Introduction à la récursivité
- Traces d'exécution de fonctions récursives
- Les tris
- Le tri par sélection
- Le tri à bulles
- 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)

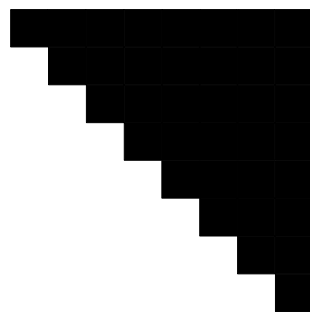
Sur un exemple ou dans le pire des cas, pour n éléments.

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



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

Complexité des tris

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

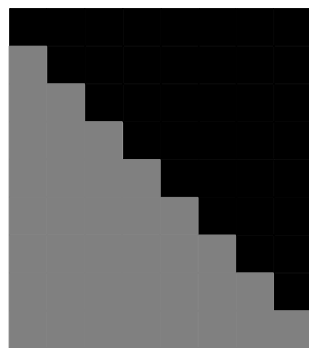
Sur un exemple ou dans le pire des cas, pour n éléments.

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)

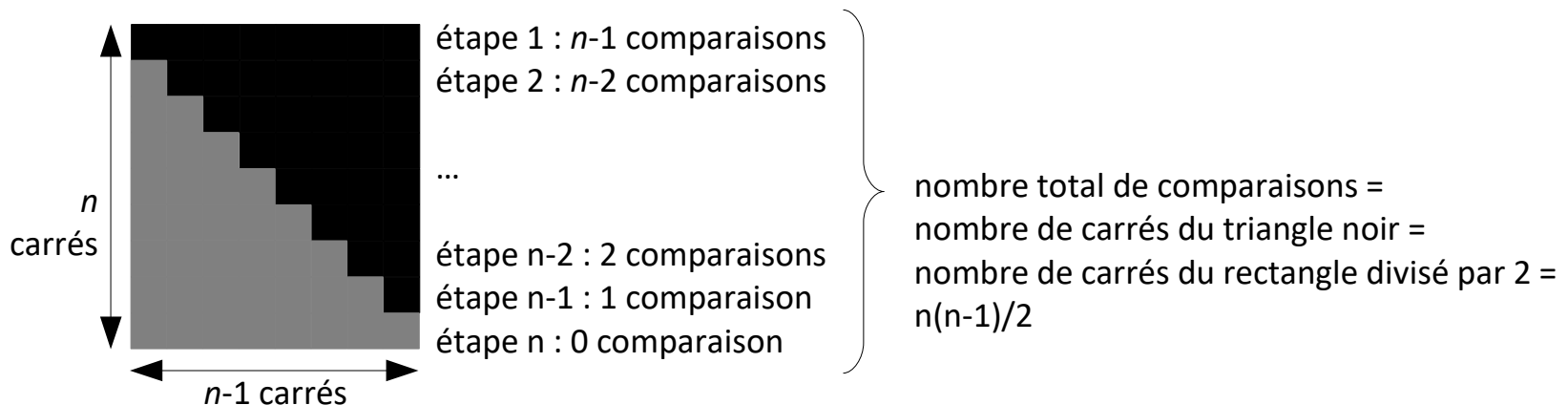
Sur un exemple ou dans le pire des cas, pour n éléments.

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