Tris : insertion, bulle, sélection, rapide & fusion

S1-3 Algorithmique - Partie I

Lycée Carnot - Dijon, 2024 - 2025

Germain Gondor

Sommaire

1. Trier ?
2. Stratégies par parcours des données
3. Stratégies « diviser pour mieux régner »

Objectifs

A la fin de la séquence d'enseignement les élèves doivent :

Objectifs

A la fin de la séquence d'enseignement les élèves doivent :

- pouvoir détailler les principales méthodes de tris

Objectifs

A la fin de la séquence d'enseignement les élèves doivent :

- pouvoir détailler les principales méthodes de tris
- pouvoir programmer les principales méthodes de tris

Sommaire

1 Trier ?

2 Stratégies par parcours des données

3 Stratégies « diviser pour mieux régner »

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Ce cours porte donc sur les principales méthodes de tris :

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Ce cours porte donc sur les principales méthodes de tris :

- tri par insertion

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Ce cours porte donc sur les principales méthodes de tris :

- tri par insertion
- tri par propagation (tri bulle)

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Ce cours porte donc sur les principales méthodes de tris :

- tri par insertion
- tri par propagation (tri bulle)
- tri par sélection

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Ce cours porte donc sur les principales méthodes de tris :

- tri par insertion
- tri par propagation (tri bulle)
- tri par sélection
- tri fusion

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Ce cours porte donc sur les principales méthodes de tris :

- tri par insertion
- tri par propagation (tri bulle)
- tri par sélection
- tri fusion
- tri rapide (Quick Sort)

Tris

Le recours au tri est extrêmement fréquent. Pour le réaliser, il existe différentes méthodes qui n'ont pas toutes la même rapidité. Un moteur de recherche célèbre est capable de proposer, à partir de mots clés et en une fraction de seconde, un nombre hallucinant de pages triées suivant des critères qui lui sont propres.

Ce cours porte donc sur les principales méthodes de tris :

- tri par insertion
- tri par propagation (tri bulle)
- tri par sélection
- tri fusion
- tri rapide (Quick Sort)

Étant donné un ensemble d'éléments munis d'une relation d'ordre, l'objectif est de classer les éléments suivant un des sens de la relation d'ordre.

Sommaire

1 Trier ?

2 Stratégies par parcours des données
- Tri par insertion
- Tri par propagation (tri bulle)
- Tri par sélection

3 Stratégies « diviser pour mieux régner »

Tri par insertion

Principe :

Étant donné un ensemble d'éléments ordonnables, le tri par insertion :

- *sélectionne le premier élément de l'ensemble et le place dans une nouvelle structure.*
- *sélectionne le deuxième élément, le compare au premier élément sélectionné puis le place avant ou après la structure.*
- *le ième élément étant sélectionné, on parcourt la structure, triée, à (i-1) éléments, soit par ordre croissant, soit par ordre décroissant pour le ranger à sa place.*

Mise en images

Étape 0

Étape 1

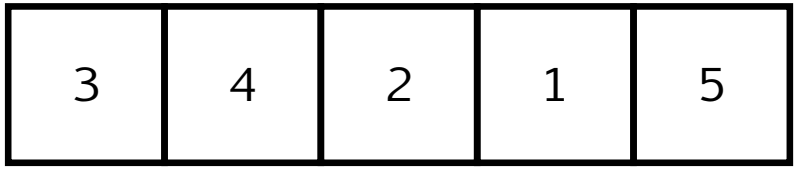
Étape 2

Étape 3

Étape 4

Étape 5

Mise en images



Étape 0

Étape 1

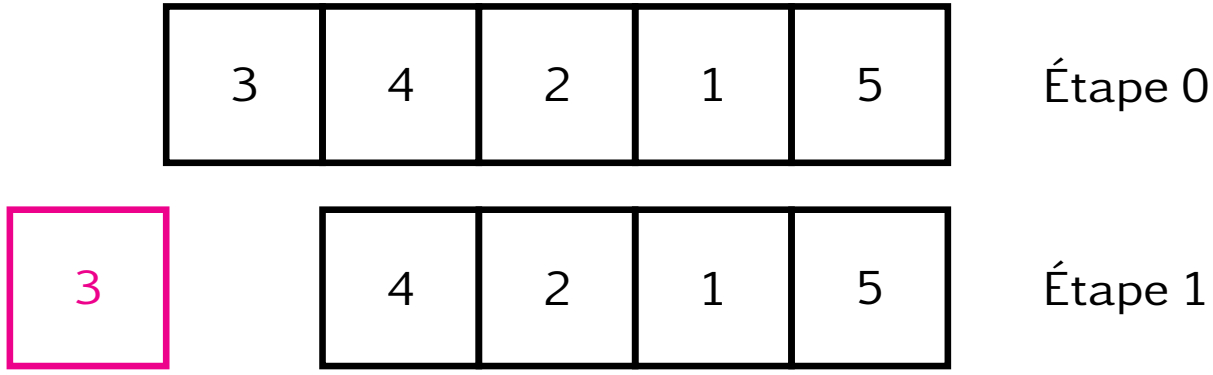
Étape 2

Étape 3

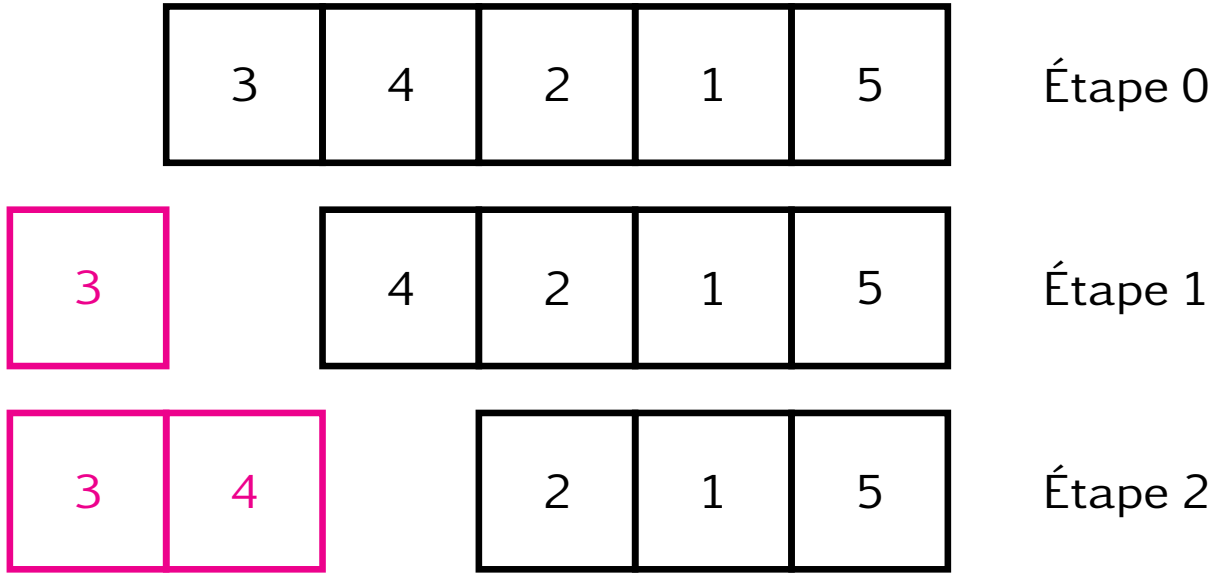
Étape 4

Étape 5

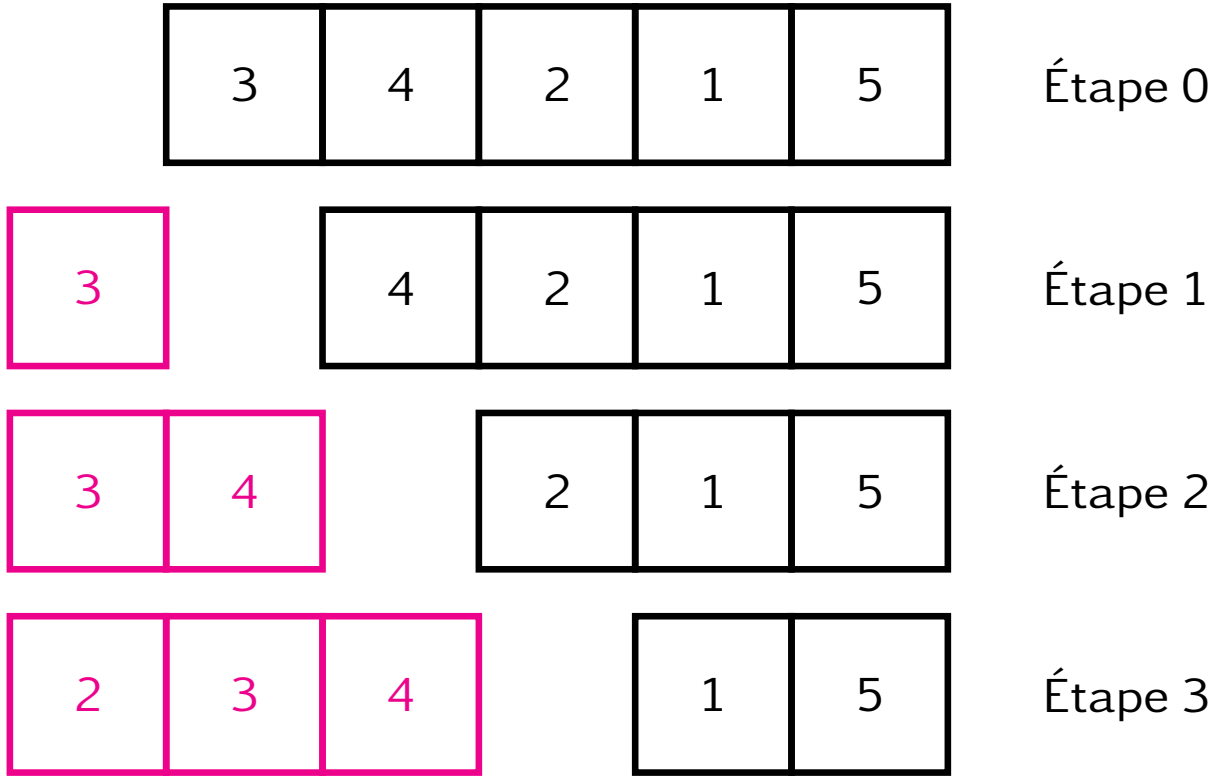
Mise en images



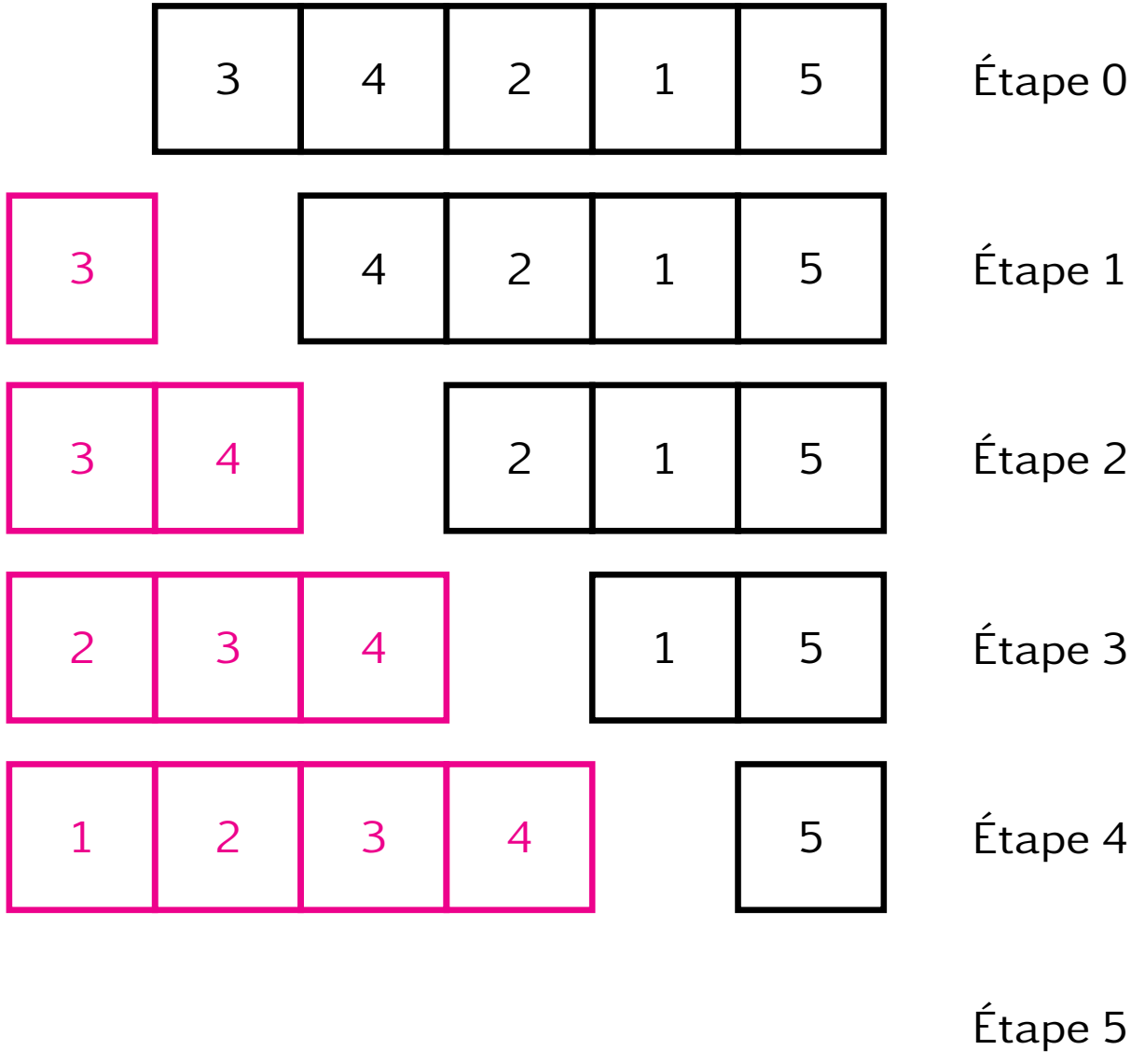
Mise en images



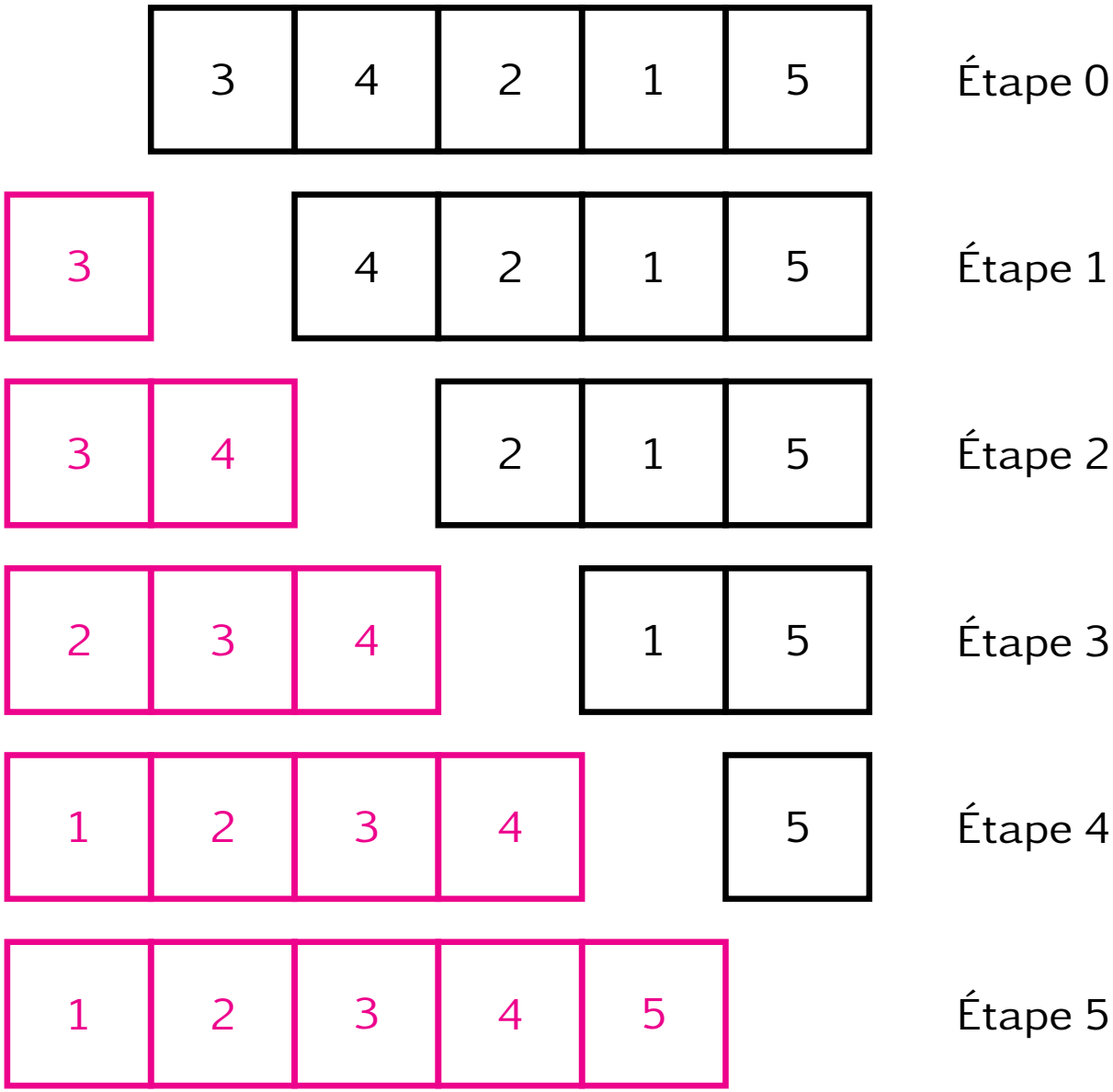
Mise en images



Mise en images



Mise en images



Algorithme

Algorithm 1 Tri insertion

entrée: T un tableau de valeurs ordonnables
résultat: T trié par ordre croissant
tri_insertion(T):

```
1: pour i de 0 à taille(T) -1 faire
2:     x ← T[i]
3:     j ← i
4:     tant que j > 0 et T[j-1] > x faire
5:         T[j] ← T[j-1]
6:         j ← j - 1
7:     fin tant que
8:     T[j] ← x
9: fin pour
   renvoi: T
```

Tri par propagation (tri bulle)

Principe :

Étant donné un tableau T de valeurs ordonnables, le tri bulle consiste :

Tri par propagation (tri bulle)

Principe :

Étant donné un tableau T de valeurs ordonnables, le tri bulle consiste :

- *à parcourir le tableau dans le sens des indices croissants et pour chaque indice j, si la valeur d'indice $j+1$ est plus petite que la valeur d'indice j, les deux valeurs sont permutées*

Tri par propagation (tri bulle)

Principe :

Étant donné un tableau T de valeurs ordonnables, le tri bulle consiste :

- *à parcourir le tableau dans le sens des indices croissants et pour chaque indice j, si la valeur d'indice $j+1$ est plus petite que la valeur d'indice j, les deux valeurs sont permutées*
- *à parcourir le tableau dans le sens des indices décroissants et pour chaque indice j, si la valeur d'indice $j-1$ est plus grande que la valeur d'indice j, les deux valeurs sont permutées*

Tri par propagation (tri bulle)

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri bulle consiste :

- *à parcourir le tableau dans le sens des indices croissants et pour chaque indice j, si la valeur d'indice $j+1$ est plus petite que la valeur d'indice j, les deux valeurs sont permutées*
- *à parcourir le tableau dans le sens des indices décroissants et pour chaque indice j, si la valeur d'indice $j-1$ est plus grande que la valeur d'indice j, les deux valeurs sont permutées*
- *à limiter l'étude du tableau au tableau `T[i-1:-i]` pour le ième parcours.*

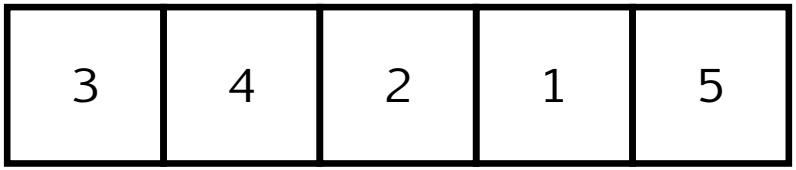
Tri par propagation (tri bulle)

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri bulle consiste :

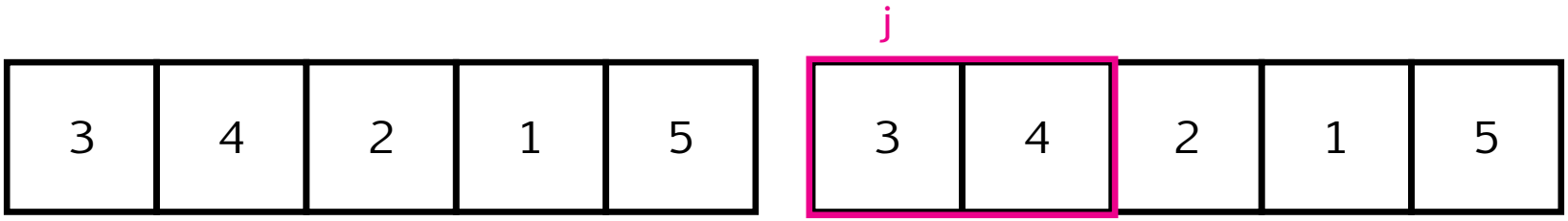
- *à parcourir le tableau dans le sens des indices croissants et pour chaque indice j, si la valeur d'indice $j+1$ est plus petite que la valeur d'indice j, les deux valeurs sont permutées*
- *à parcourir le tableau dans le sens des indices décroissants et pour chaque indice j, si la valeur d'indice $j-1$ est plus grande que la valeur d'indice j, les deux valeurs sont permutées*
- *à limiter l'étude du tableau au tableau `T[i-1:-i]` pour le ième parcours.*
- *à poursuivre jusqu'au statu quo ou quand la taille du tableau à étudier est inférieure à 2*

Mise en images



Étape 0

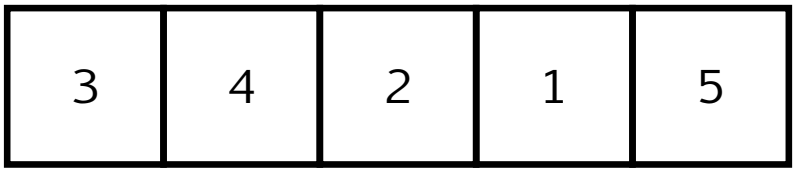
Mise en images



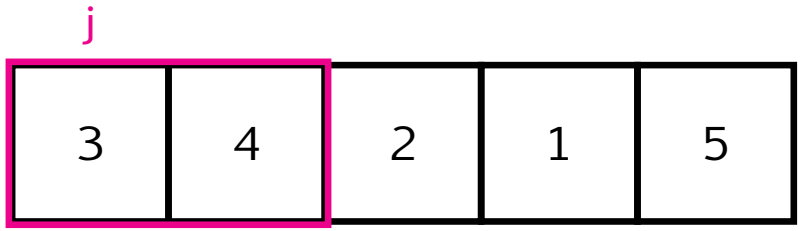
Étape 0

Étape 1

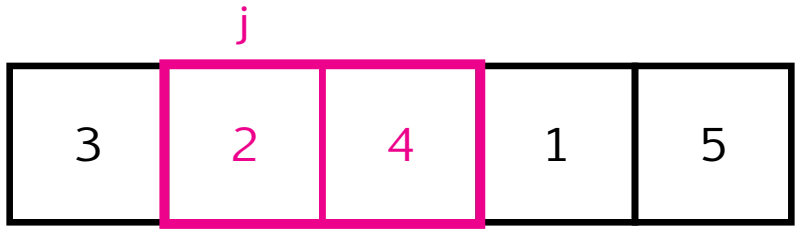
Mise en images



Étape 0

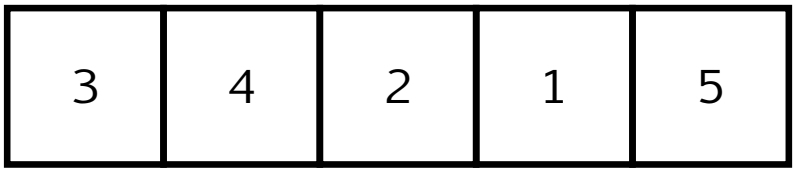


Étape 1

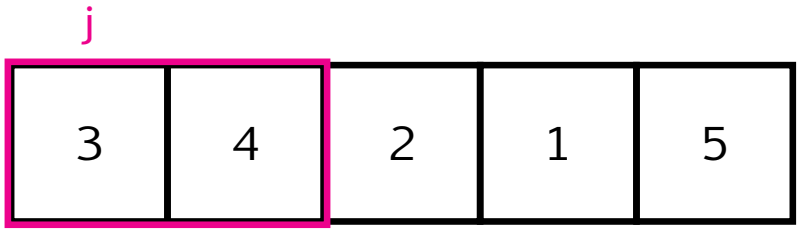


Étape 2

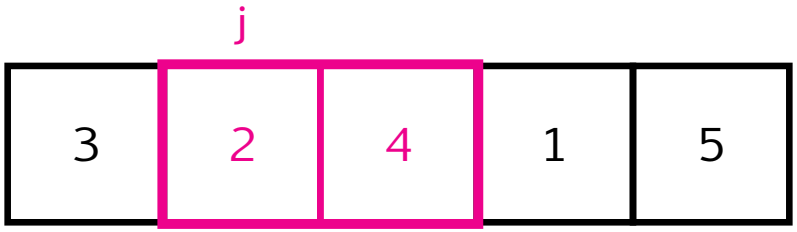
Mise en images



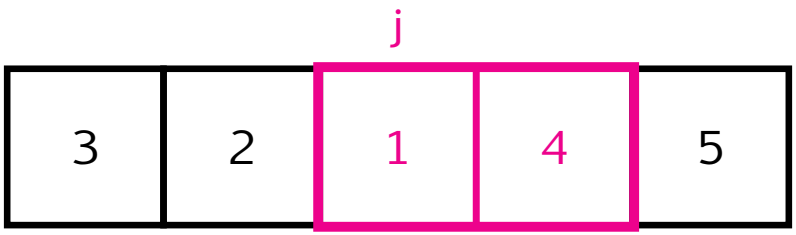
Étape 0



Étape 1

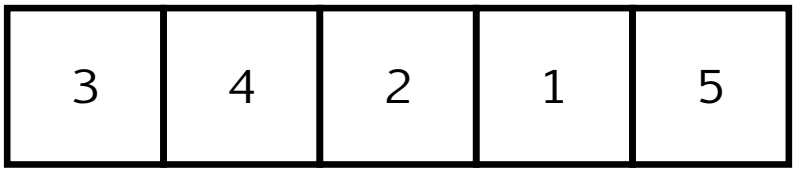


Étape 2

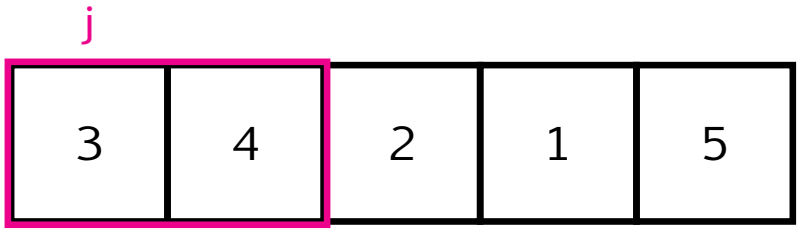


Étape 3

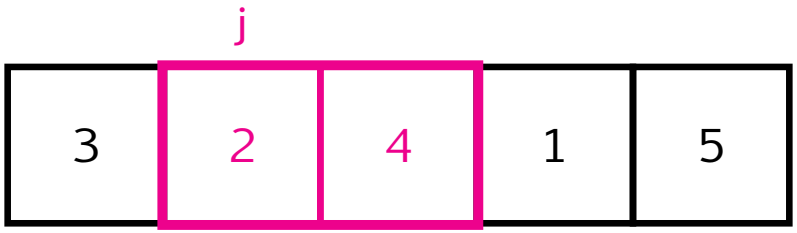
Mise en images



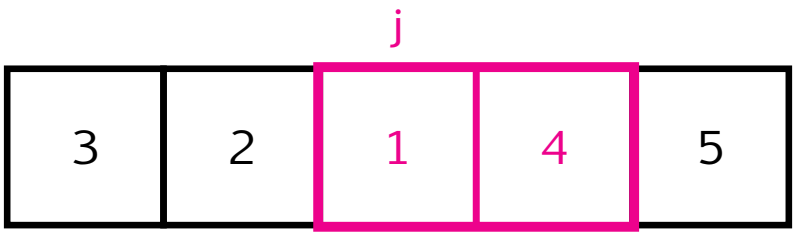
Étape 0



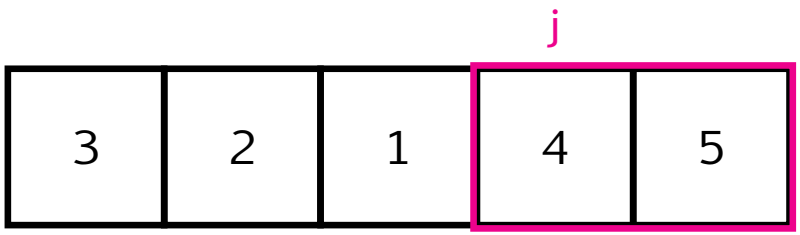
Étape 1



Étape 2

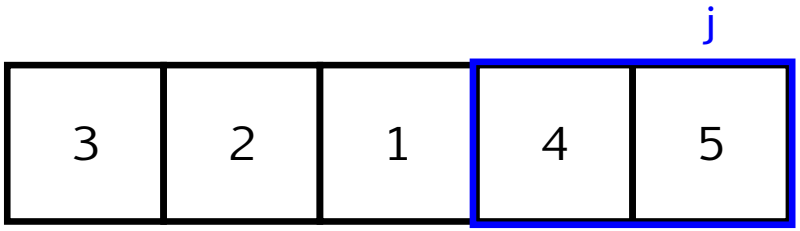


Étape 3



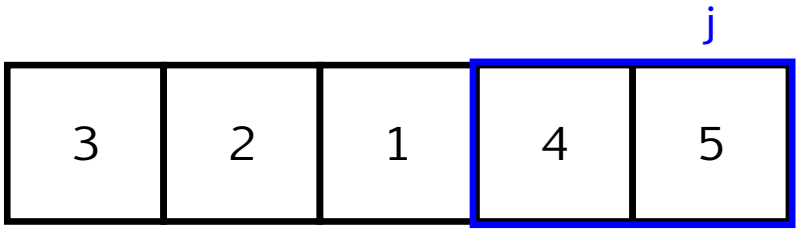
Étape 4

Mise en images

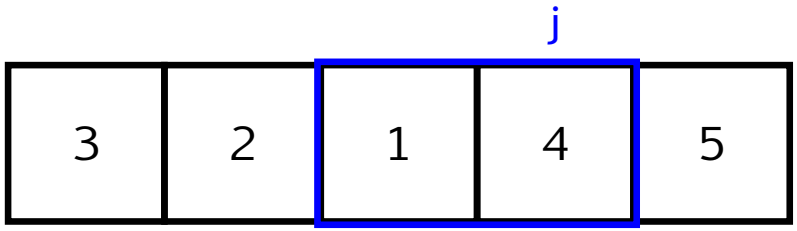


Étape 5

Mise en images

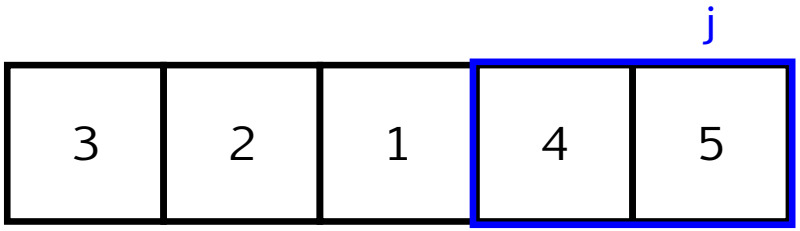


Étape 5

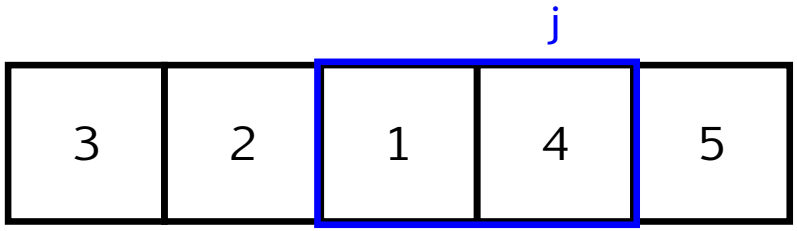


Étape 6

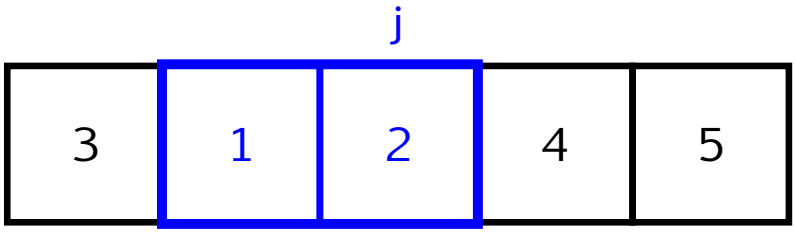
Mise en images



Étape 5

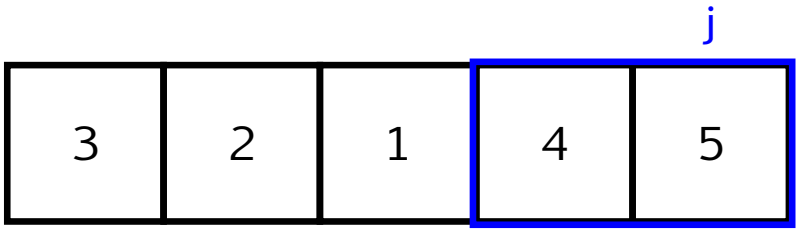


Étape 6

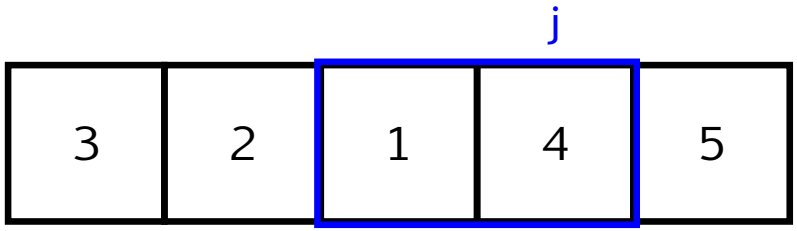


Étape 7

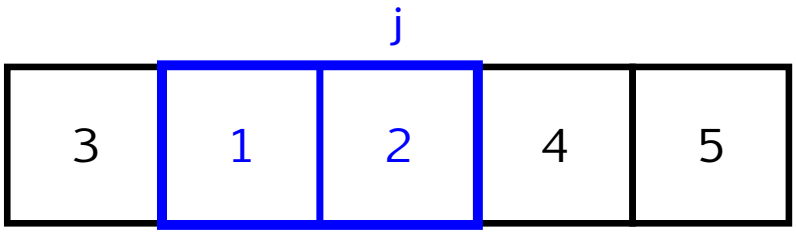
Mise en images



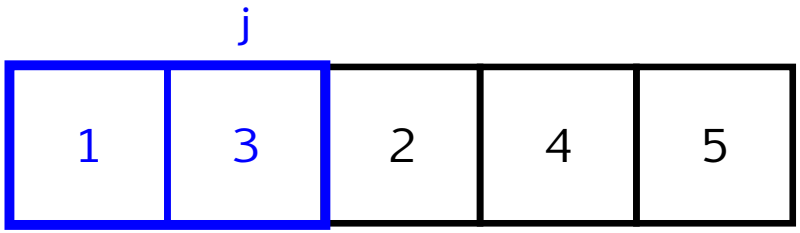
Étape 5



Étape 6

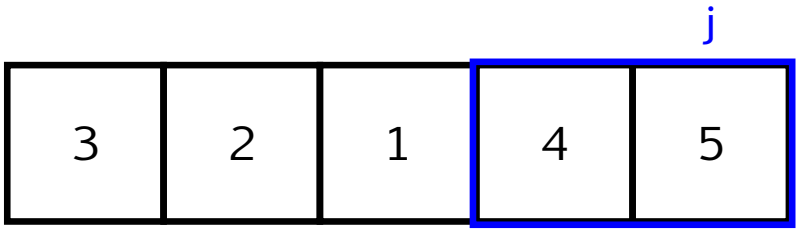


Étape 7

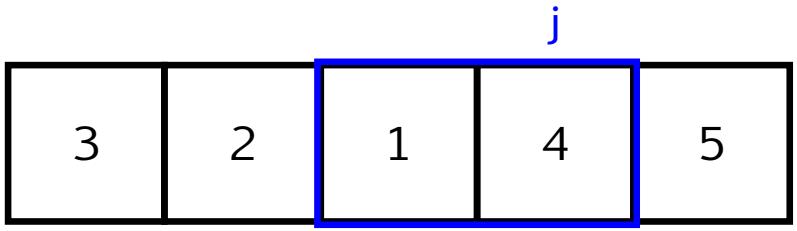


Étape 8

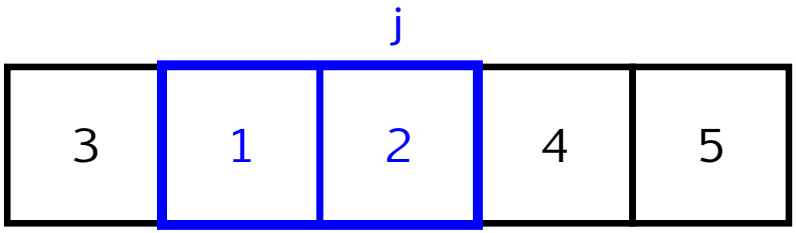
Mise en images



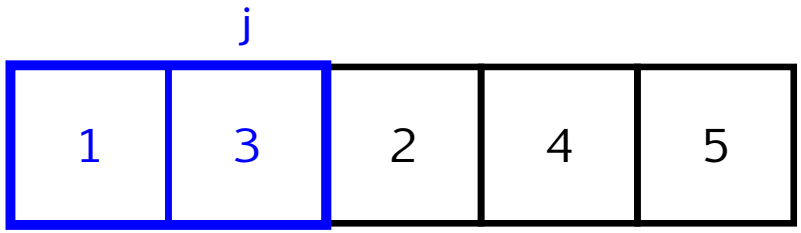
Étape 5



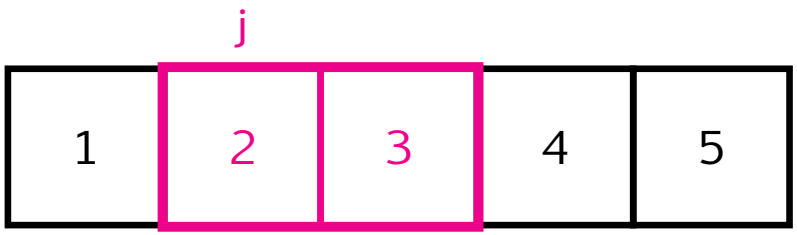
Étape 6



Étape 7

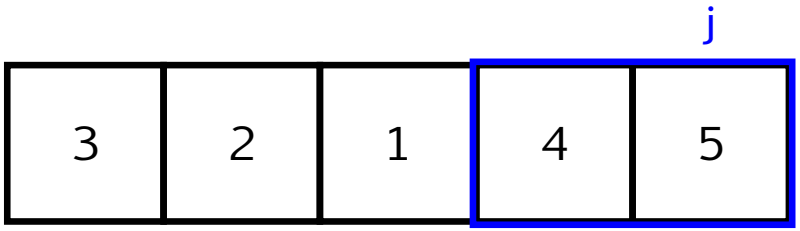


Étape 8

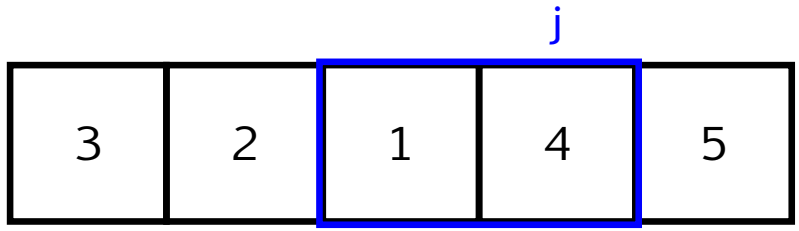


Étape 9

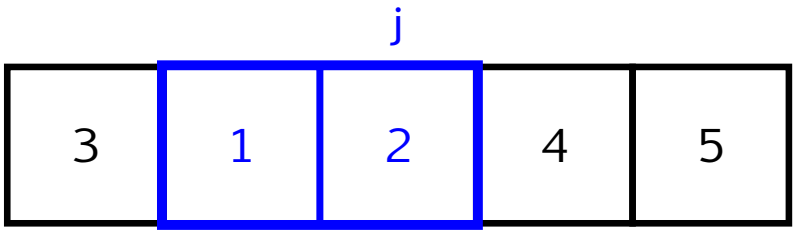
Mise en images



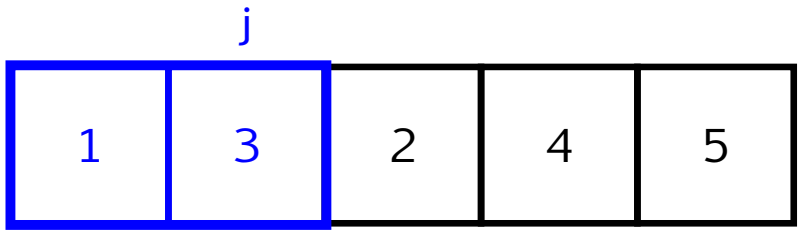
Étape 5



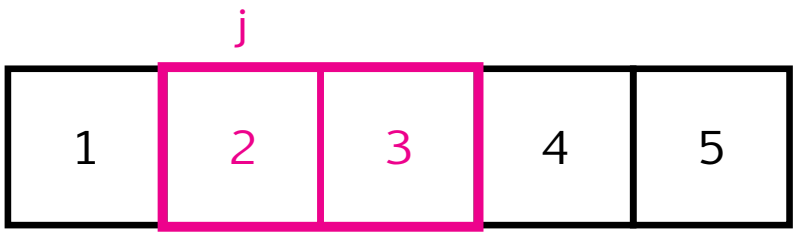
Étape 6



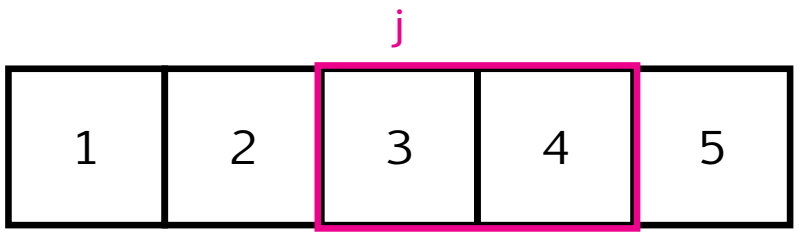
Étape 7



Étape 8



Étape 9



Étape 10

Algorithme

Algorithm 2 Tri par propagation (tri bulle)

entrée: T un tableau de valeurs ordonnables
résultat: T trié par ordre croissant
tri_bulle(T):

```
1:  imin, imax ← 0, taille(T) - 1
2:  tant que imin < imax faire
3:      pour i de imin à imax-1 faire
4:          si T[i] > T[i+1] alors
5:              T[i], T[i+1] ← T[i+1], T[i]
6:          fin si
7:      fin pour
8:      imax ← imax -1
9:      pour i de imax à imin + 1 avec un pas de -1 faire
10:         si T[i] < T[i-1] alors
11:             T[i], T[i-1] ← T[i-1], T[i]
12:         fin si
13:     fin pour
14:     imin ← imin + 1
15: fin tant que
16: renvoi: T
```

Tri par sélection

Principe :

Étant donné un tableau T de valeurs ordonnables, le tri par sélection consiste :

Tri par sélection

Principe :

Étant donné un tableau T de valeurs ordonnables, le tri par sélection consiste :

- *à déterminer l'élément le plus petit du tableau et l'échanger de place avec le premier élément du tableau*

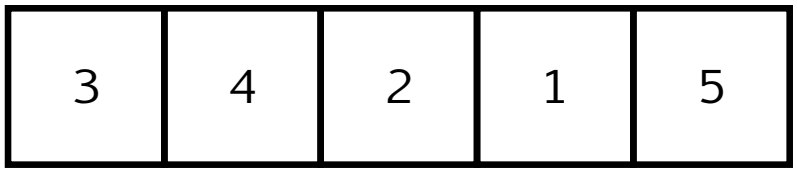
Tri par sélection

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri par sélection consiste :

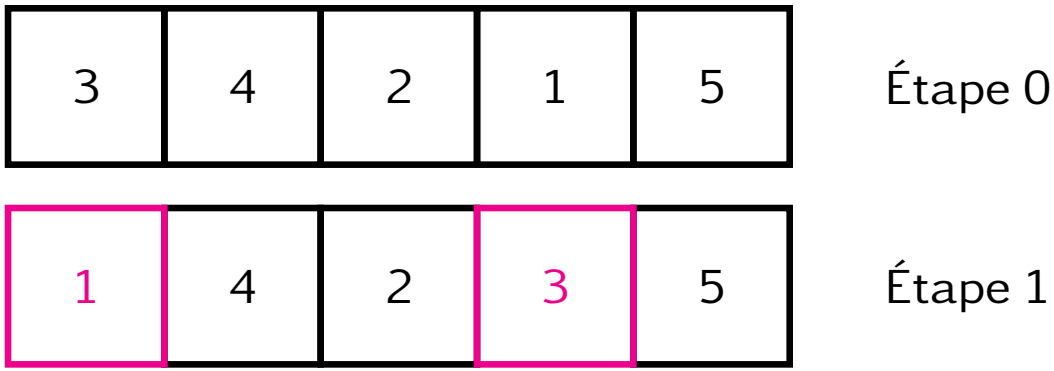
- *à déterminer l'élément le plus petit du tableau et l'échanger de place avec le premier élément du tableau*
- *à recommencer à l'itération `i+1` en omettant les `i` premières valeurs du tableau*

Mise en images

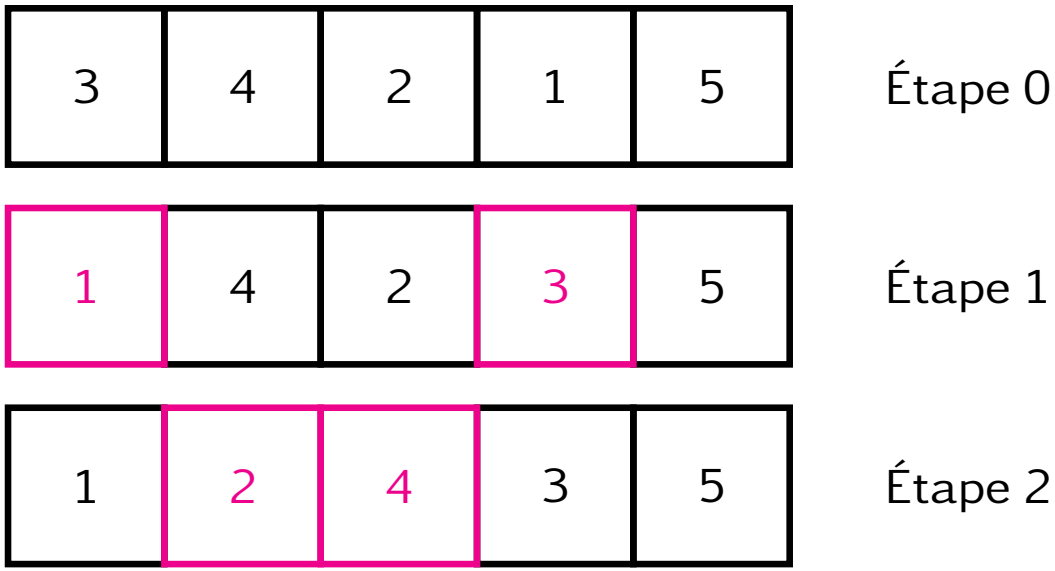


Étape 0

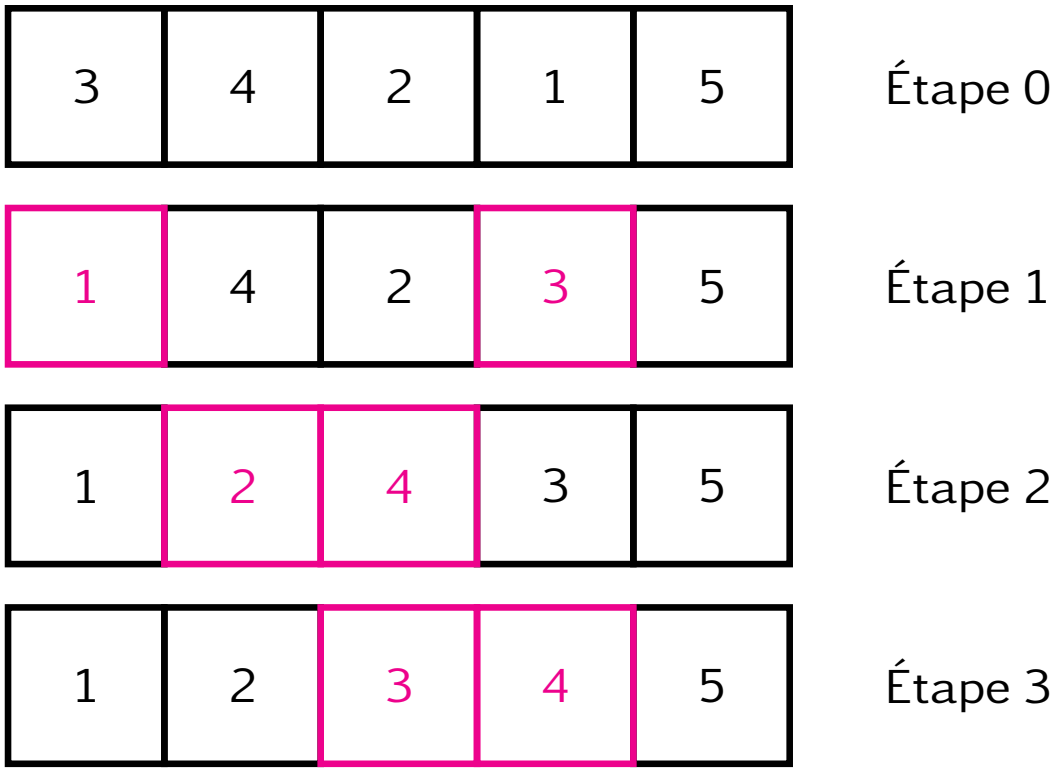
Mise en images



Mise en images



Mise en images



Algorithme

Algorithm 3 Tri sélection

entrée: T un tableau de valeurs ordonnables
résultat: T trié par ordre croissant
tri_selection(T):

```
1:  deb, n ← 0, taille(T)
2:  tant que deb < n-1 faire
3:      imin, vmin← deb, T[deb]
4:      pour j de deb+1 à n-1 faire
5:          si vmin > T[j] alors
6:              imin, vmin ← j, T[j]
7:          fin si
8:      fin pour
9:      T[imin], T[deb] ← T[deb], T[imin]
10:     deb ← deb+1
11: fin tant que
12: renvoi: T
```

Sommaire

1 Trier ?

2 Stratégies par parcours des données

3 Stratégies ≪ diviser pour mieux régner ≫
- Tri rapide (Quick Sort)
- Tri fusion

Tri rapide (Quick Sort)

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri rapide consiste :

Tri rapide (Quick Sort)

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri rapide consiste :

- *à prendre au hasard un élément du tableau qu'on appelle pivot*

Tri rapide (Quick Sort)

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri rapide consiste :

- *à prendre au hasard un élément du tableau qu'on appelle pivot*
- *à créer une liste avec toutes les valeurs de `T` plus petites que le pivot et une autre liste avec toutes les valeurs de `T` plus grande que le pivot*

Tri rapide (Quick Sort)

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri rapide consiste :

- *à prendre au hasard un élément du tableau qu'on appelle pivot*
- *à créer une liste avec toutes les valeurs de `T` plus petites que le pivot et une autre liste avec toutes les valeurs de `T` plus grande que le pivot*
- *à assembler à gauche les valeurs plus petites que le pivot triées par tri rapide, le pivot au centre et les valeurs plus grandes que le pivot triées par tri rapide à droite*

Tri rapide (Quick Sort)

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri rapide consiste :

- *à prendre au hasard un élément du tableau qu'on appelle pivot*
- *à créer une liste avec toutes les valeurs de `T` plus petites que le pivot et une autre liste avec toutes les valeurs de `T` plus grande que le pivot*
- *à assembler à gauche les valeurs plus petites que le pivot triées par tri rapide, le pivot au centre et les valeurs plus grandes que le pivot triées par tri rapide à droite*
- *à retourner le tableau si celui-ci est vide ou n'a qu'un élément*

Tri rapide (Quick Sort)

Principe :

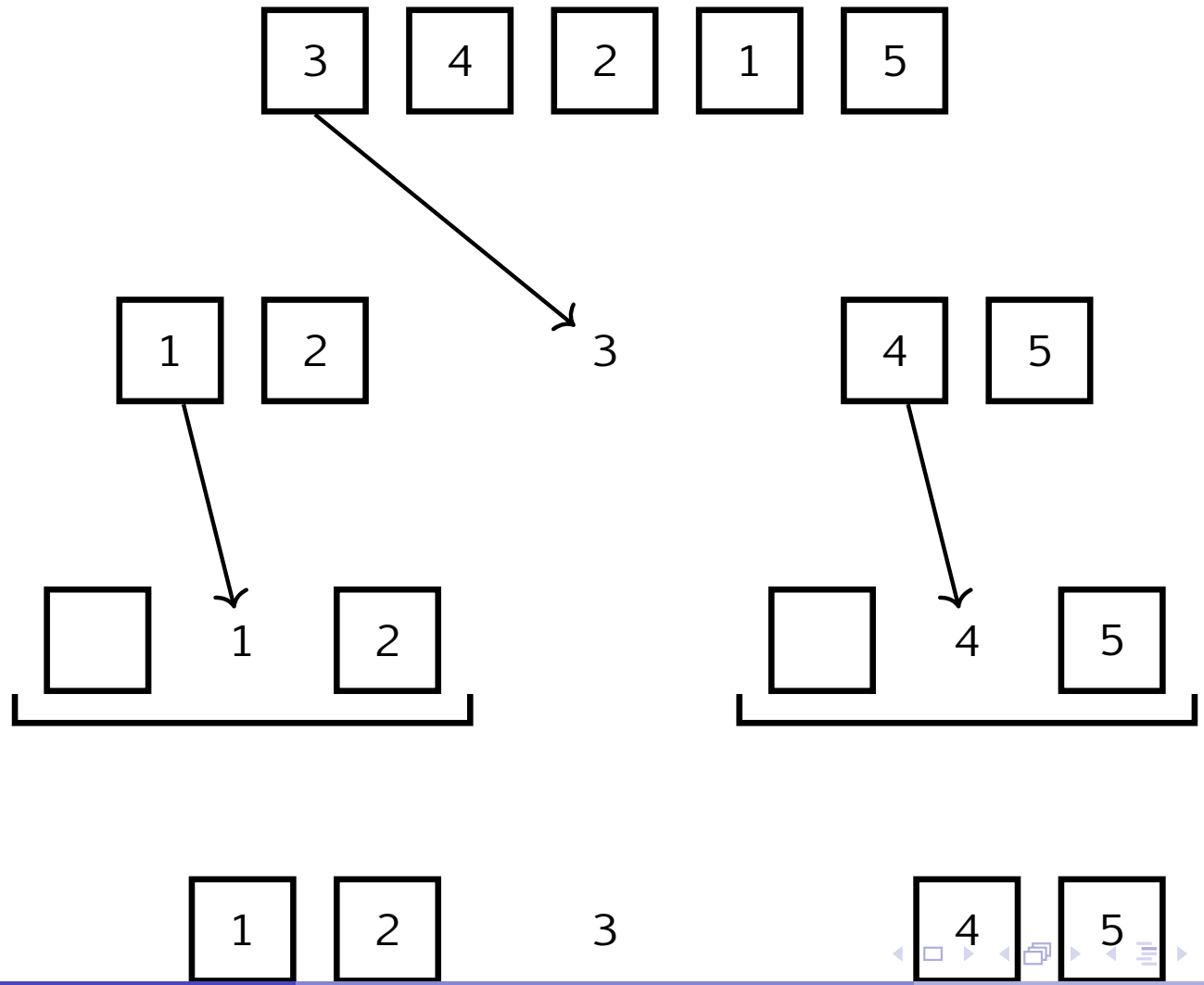
Étant donné un tableau `T` de valeurs ordonnables, le tri rapide consiste :

- *à prendre au hasard un élément du tableau qu'on appelle pivot*
- *à créer une liste avec toutes les valeurs de `T` plus petites que le pivot et une autre liste avec toutes les valeurs de `T` plus grande que le pivot*
- *à assembler à gauche les valeurs plus petites que le pivot triées par tri rapide, le pivot au centre et les valeurs plus grandes que le pivot triées par tri rapide à droite*
- *à retourner le tableau si celui-ci est vide ou n'a qu'un élément*

C'est l'algorithme de tri le plus utilisé et peut être même l'algorithme le plus utilisé dans le monde.

Mise en images

Par défaut, on choisit comme pivot le premier élément du tableau. Le premier pivot est donc 3.



Algorithme

Algorithm 4 Tri rapide

entrée: T un tableau de valeurs ordonnables
résultat: T trié par ordre croissant
tri_rapide(T):

```
1:  si taille(T) < 2 alors
2:      renvoi: T
3:  fin si
4:  petits, grands ← [], []
5:  pivot ← T.pop()
6:  pour i de 1 à taille(T) faire
7:      x ← T.pop()
8:      si x ≤ pivot alors
9:          petits.append(x)
10:     sinon
11:         grands.append(x)
12:     fin si
13: fin pour
14: tri_rapide(petits)
15: tri_rapide(grands)
16: renvoi: petits + [pivot] + grands
```

Tri fusion

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri fusion consiste :

Tri fusion

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri fusion consiste :

- *à couper le tableau en deux parts égales*

Tri fusion

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri fusion consiste :

- *à couper le tableau en deux parts égales*
- *à trier par tri fusion chacune des deux parties*

Tri fusion

Principe :

Étant donné un tableau `T` de valeurs ordonnables, le tri fusion consiste :

- *à couper le tableau en deux parts égales*
- *à trier par tri fusion chacune des deux parties*
- *à fusionner les deux parties*

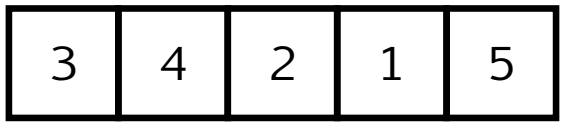
Tri fusion

Principe :

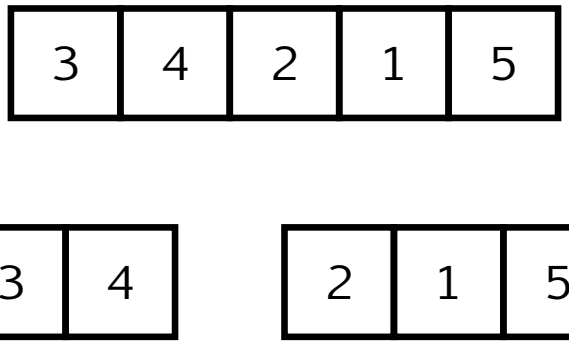
Étant donné un tableau `T` de valeurs ordonnables, le tri fusion consiste :

- *à couper le tableau en deux parts égales*
- *à trier par tri fusion chacune des deux parties*
- *à fusionner les deux parties*
- *à retourner directement la partie si elle ne comprend qu'un élément*

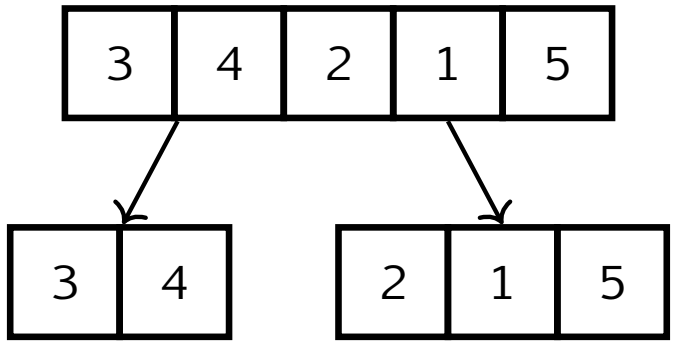
Mise en images globale



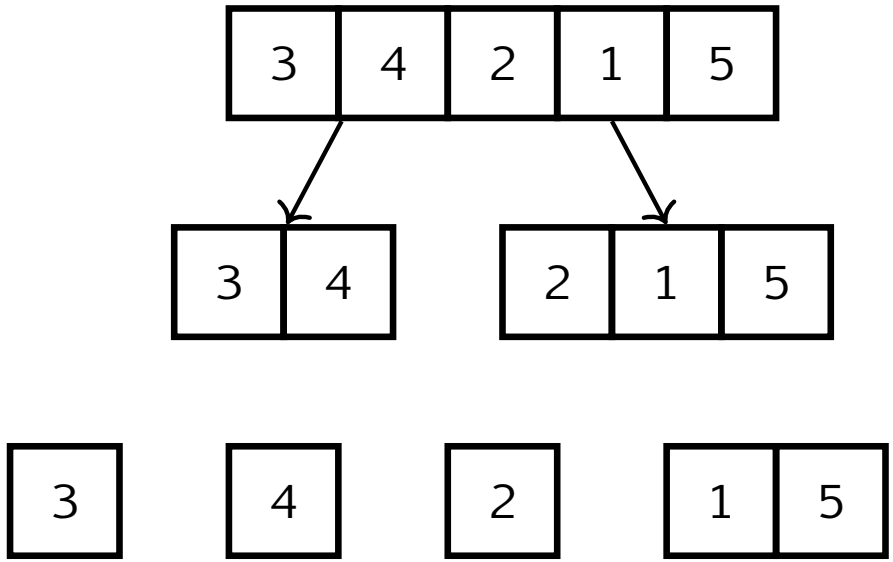
Mise en images globale



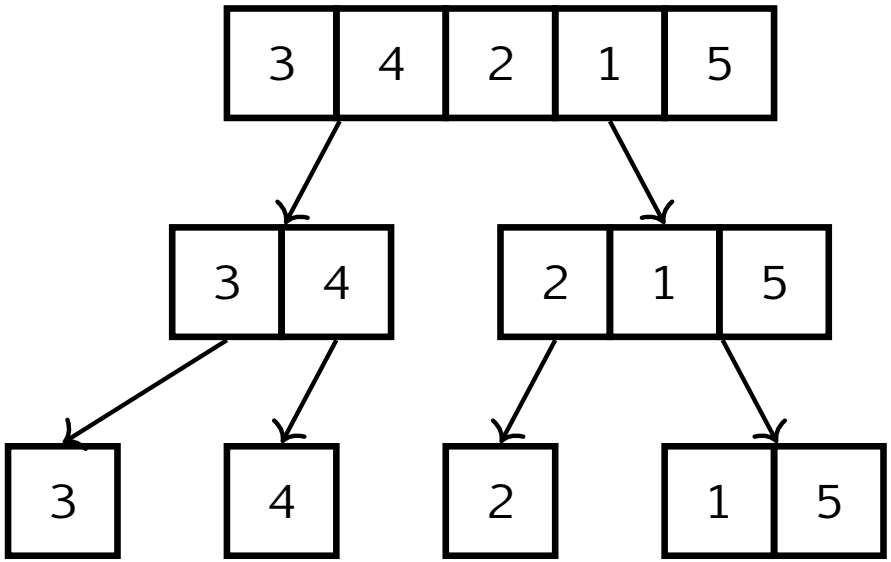
Mise en images globale



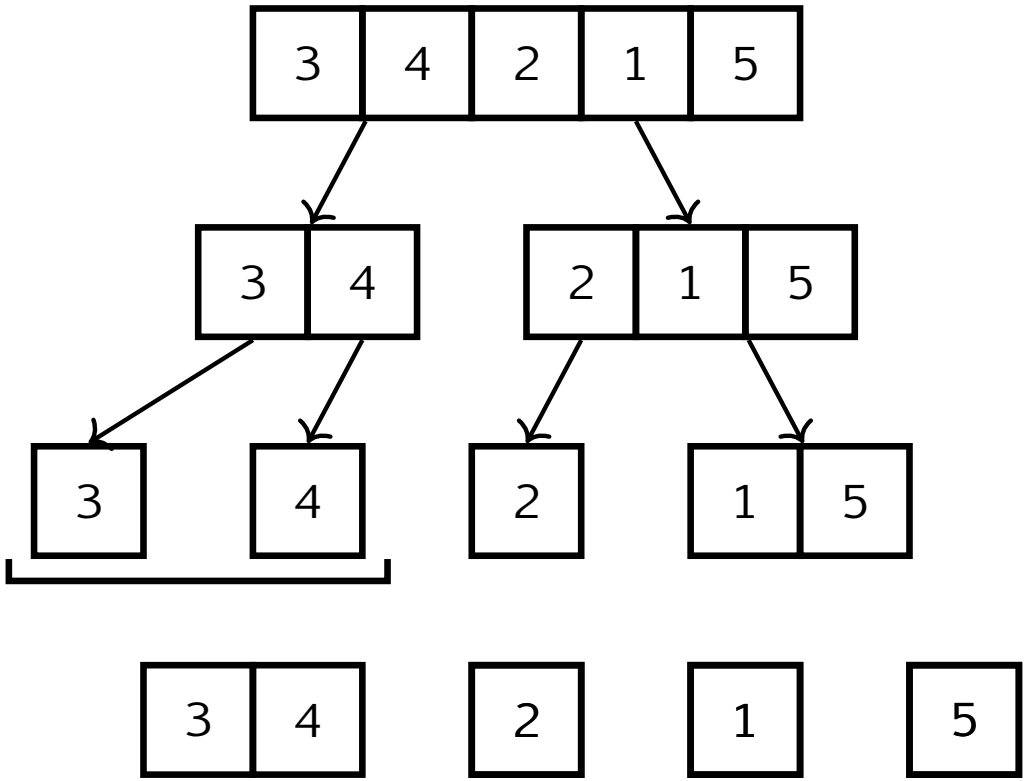
Mise en images globale



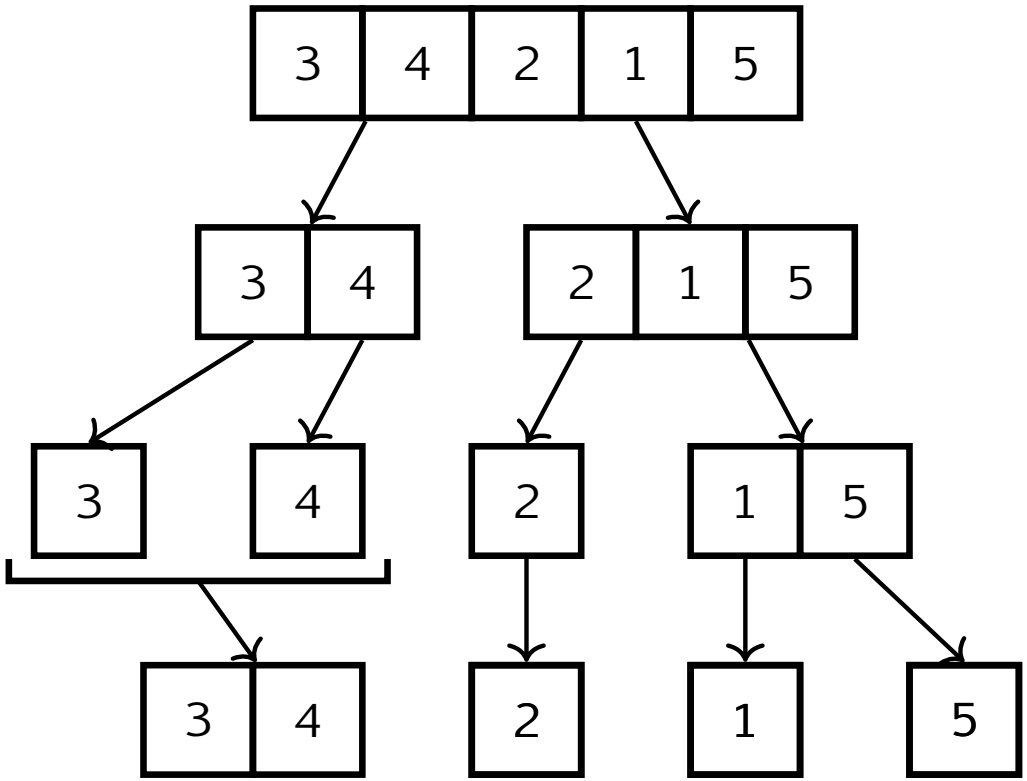
Mise en images globale



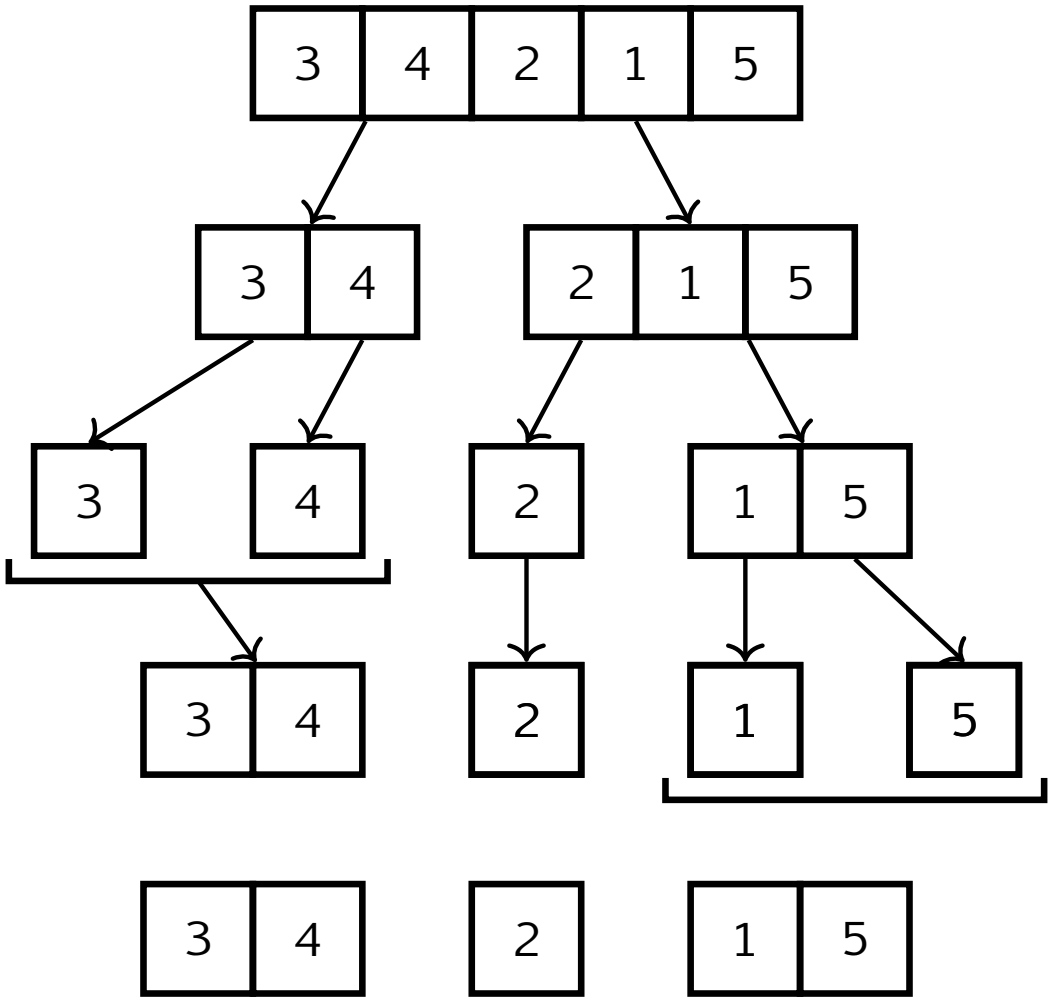
Mise en images globale



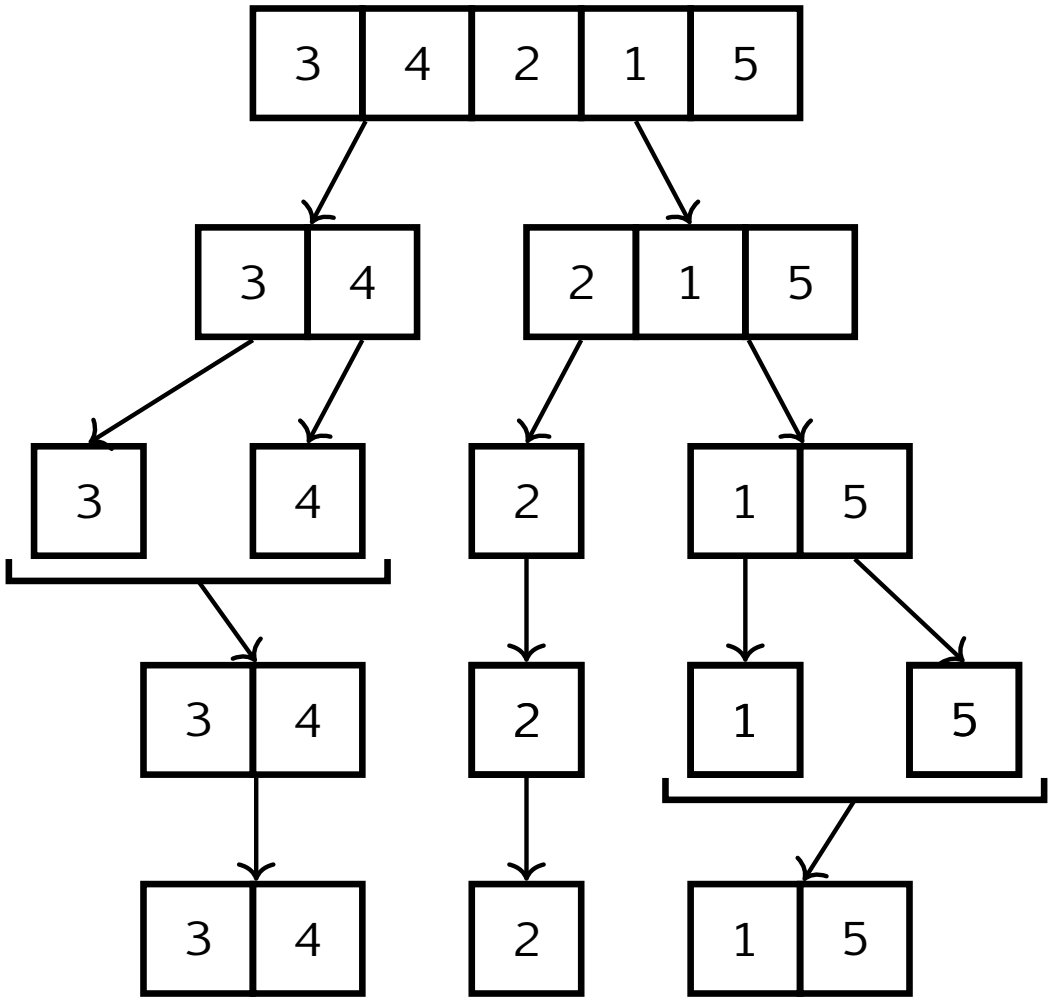
Mise en images globale



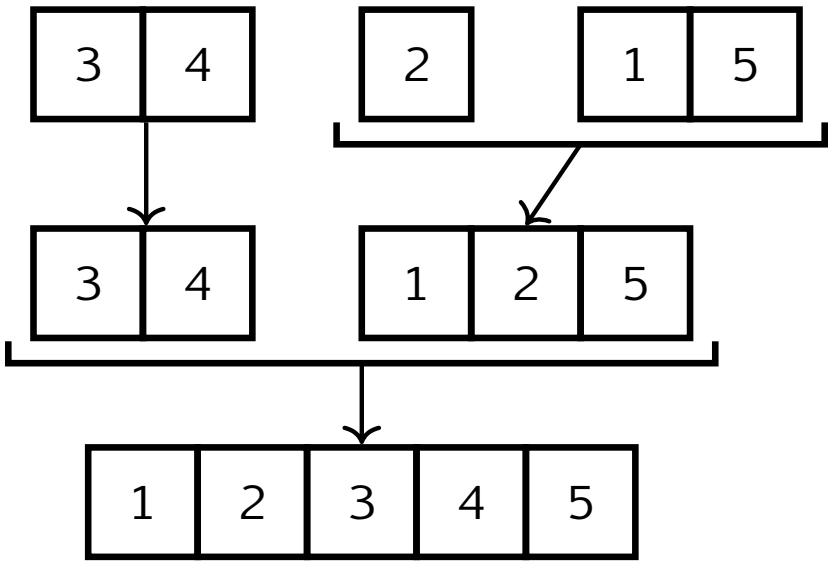
Mise en images globale



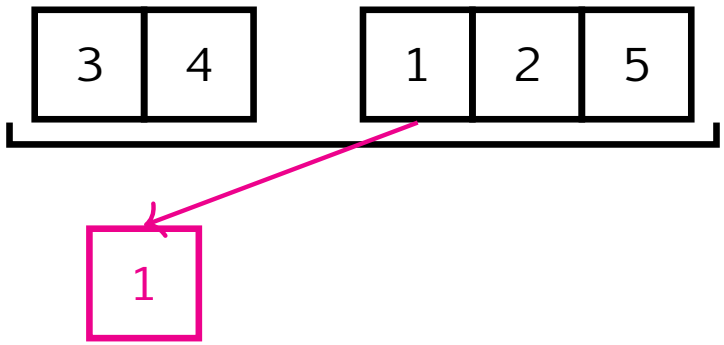
Mise en images globale



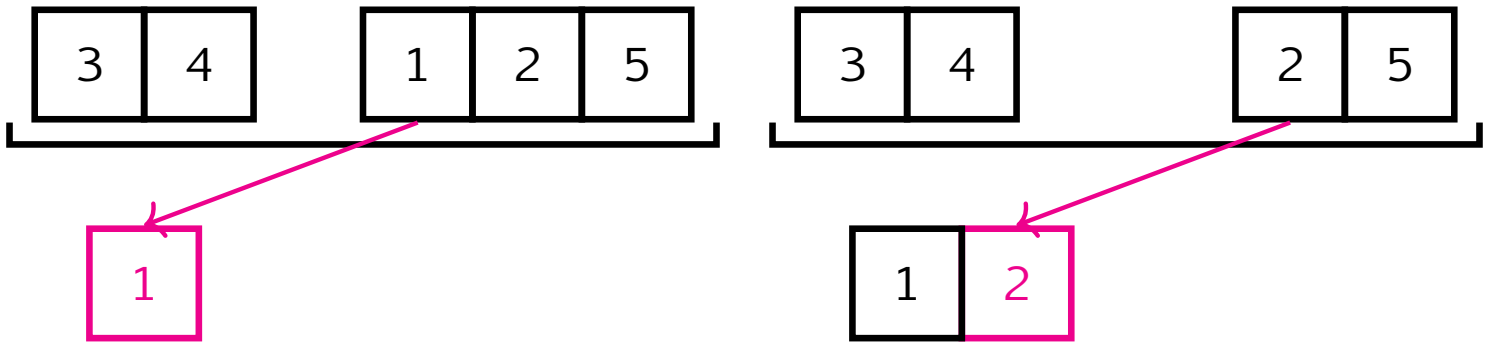
Mise en images globale



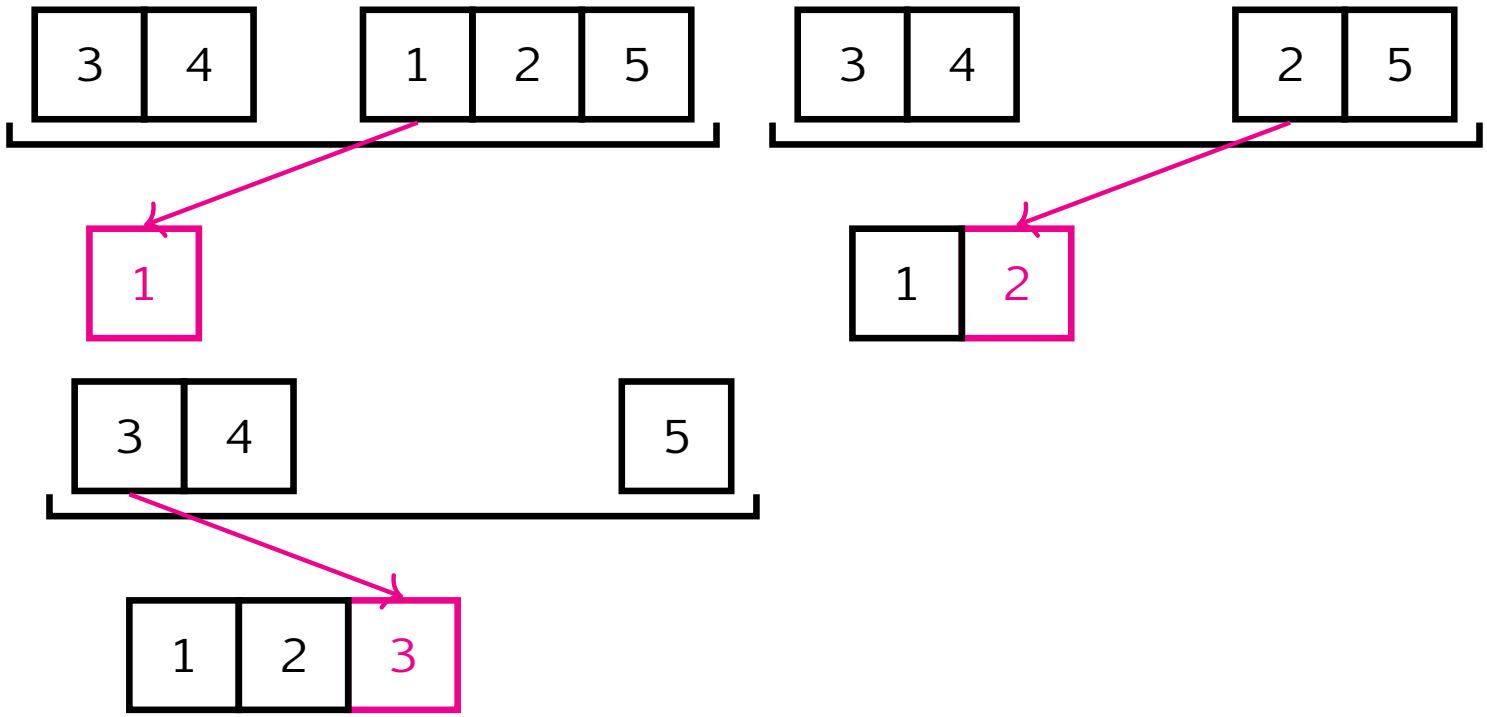
Mise en images de la dernière fusion



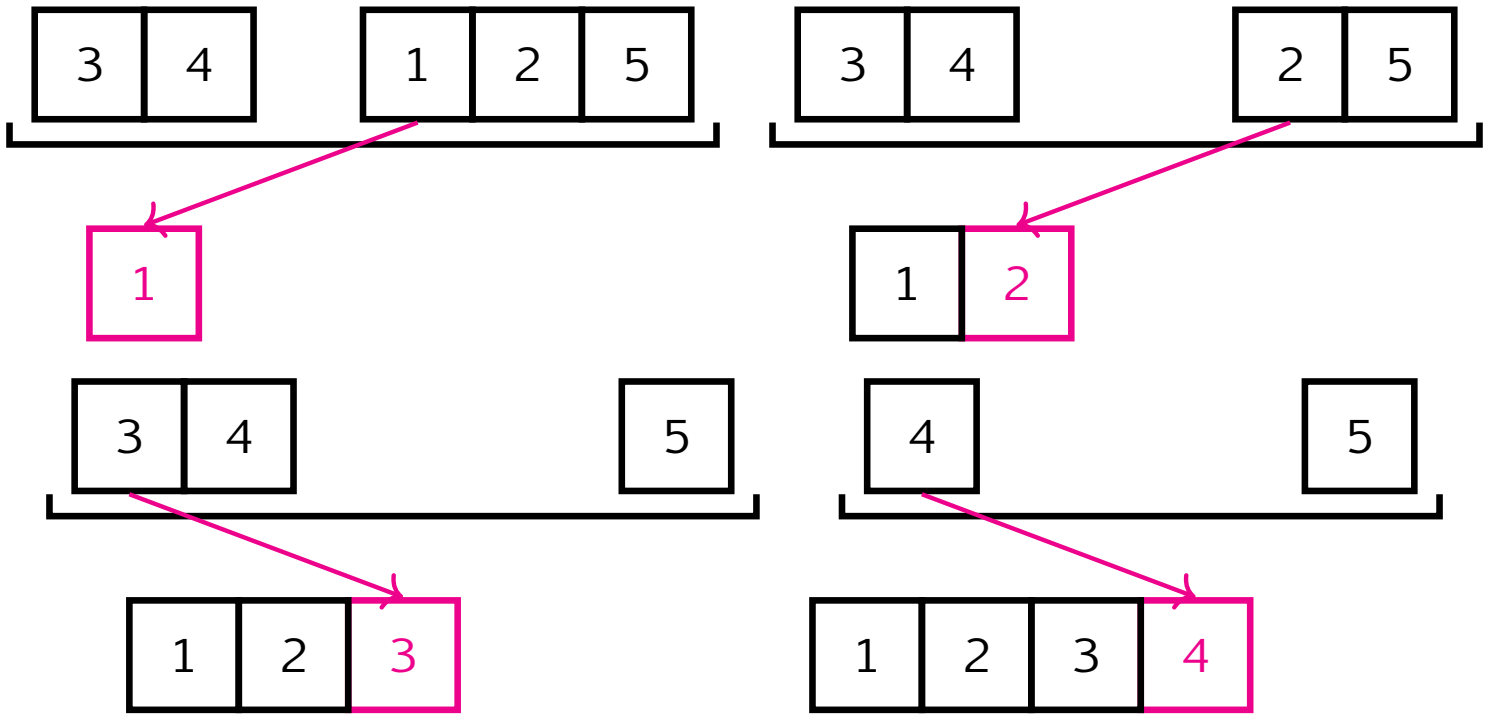
Mise en images de la dernière fusion



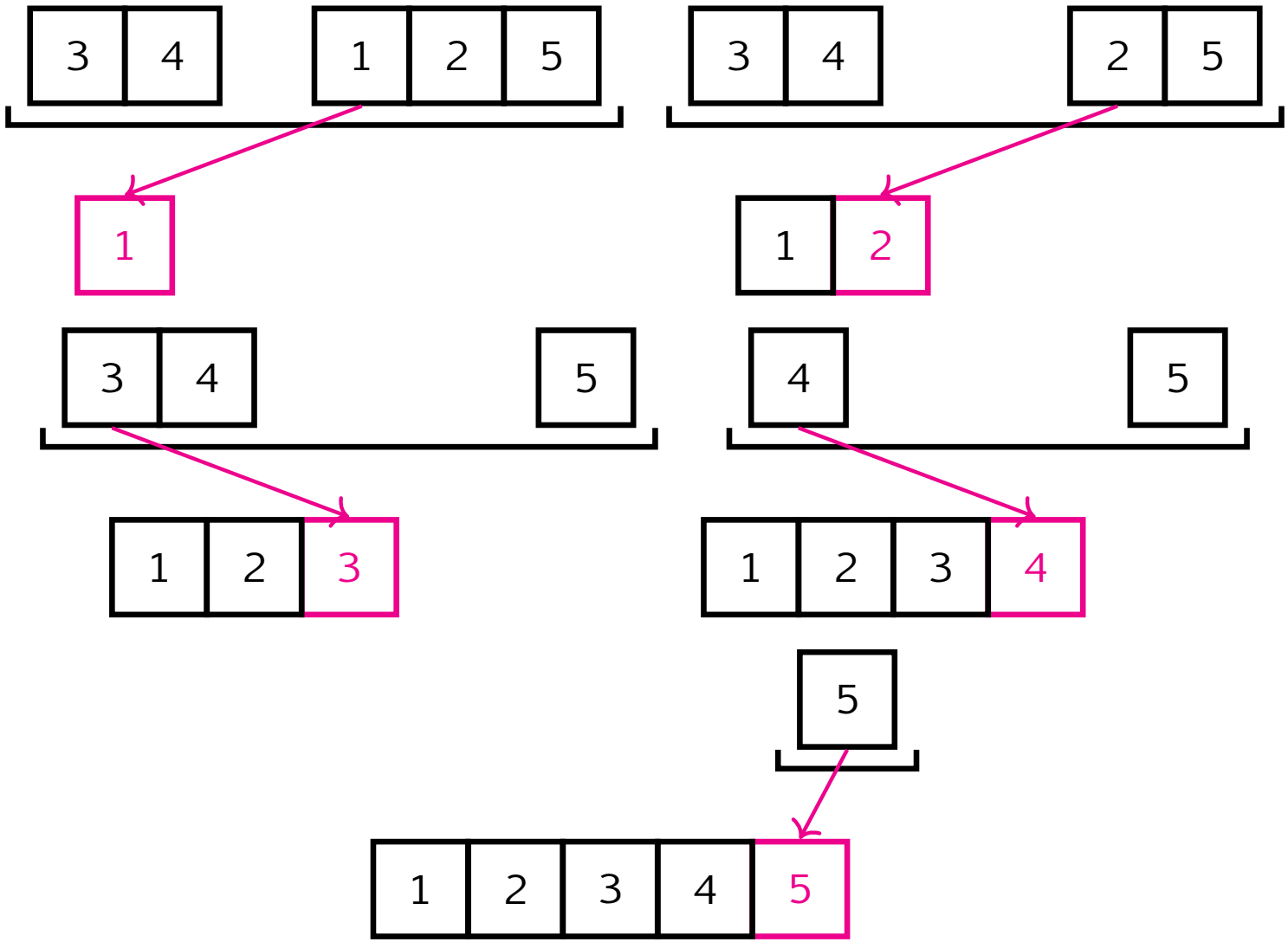
Mise en images de la dernière fusion



Mise en images de la dernière fusion



Mise en images de la dernière fusion



Algorithme

Algorithm 5 Tri fusion

entrée: T un tableau de valeurs ordonnables
résultat: T trié par ordre croissant
tri_fusion(T):

```
1: si taille(T) == 1 alors
2:     renvoi: T
3: fin si
4: mil ← taille(T) // 2
5: gauche ← tri_fusion(T[:mil])
6: droite ← tri_fusion(T[mil:])
7: renvoi: fusion(gauche, droite)
```

Algorithme

Algorithm 6 Fusion

entrée: `P` et `Q` deux tableaux valeurs triés
résultat: `T` la fusion triée des tableaux `P` et `Q`

```
    fusion(P, Q):
 1: T, pi, qi ← [], 0, 0
 2: np, nq ← taille(P), taille(Q)
 3: tant que np > pi et nq > qi faire
 4:     si P[pi] < Q[qi] alors
 5:         T.append(P[pi])
 6:         pi ← pi + 1
 7:     sinon
 8:         T.append(Q[qi])
 9:         qi ← qi + 1
10:     fin si
11: fin tant que
12: si np > pi alors
13:     T ← T + P[pi:]
14: sinon
15:     T ← T + Q[qi:]
16: fin si
17: renvoi: T
```