

Introduction à l'informatique CM5

Antonio E. Porreca

<https://aeporreca.org/introinfo>

 **Rappel** 

**Vendredi 18 octobre
on a le partiel !**

Algorithmes de tri

Algorithmes de tri pour accélérer la recherche dans un tableau

- La recherche dans un tableau non trié prend temps $O(n)$ avec la **recherche séquentielle** (ou linéaire)
- Par contre, on peut faire une **recherche dichotomique** dans un tableau trié en temps $O(\log_2 n)$
- Donc ça vaut la peine de trier le tableau si on a beaucoup de recherches à faire

Algorithmes de tri dans le commerce électronique

amazonie.fr



amazonie

Chercher :

Le Petit Prince

Résultats 1–20 sur 928572785 pour « Le Petit Prince »

Trier par :

prix croissant

prix décroissant

note moyenne

nouveauté



Le Petit Prince
de Antoine de Saint-Exupéry

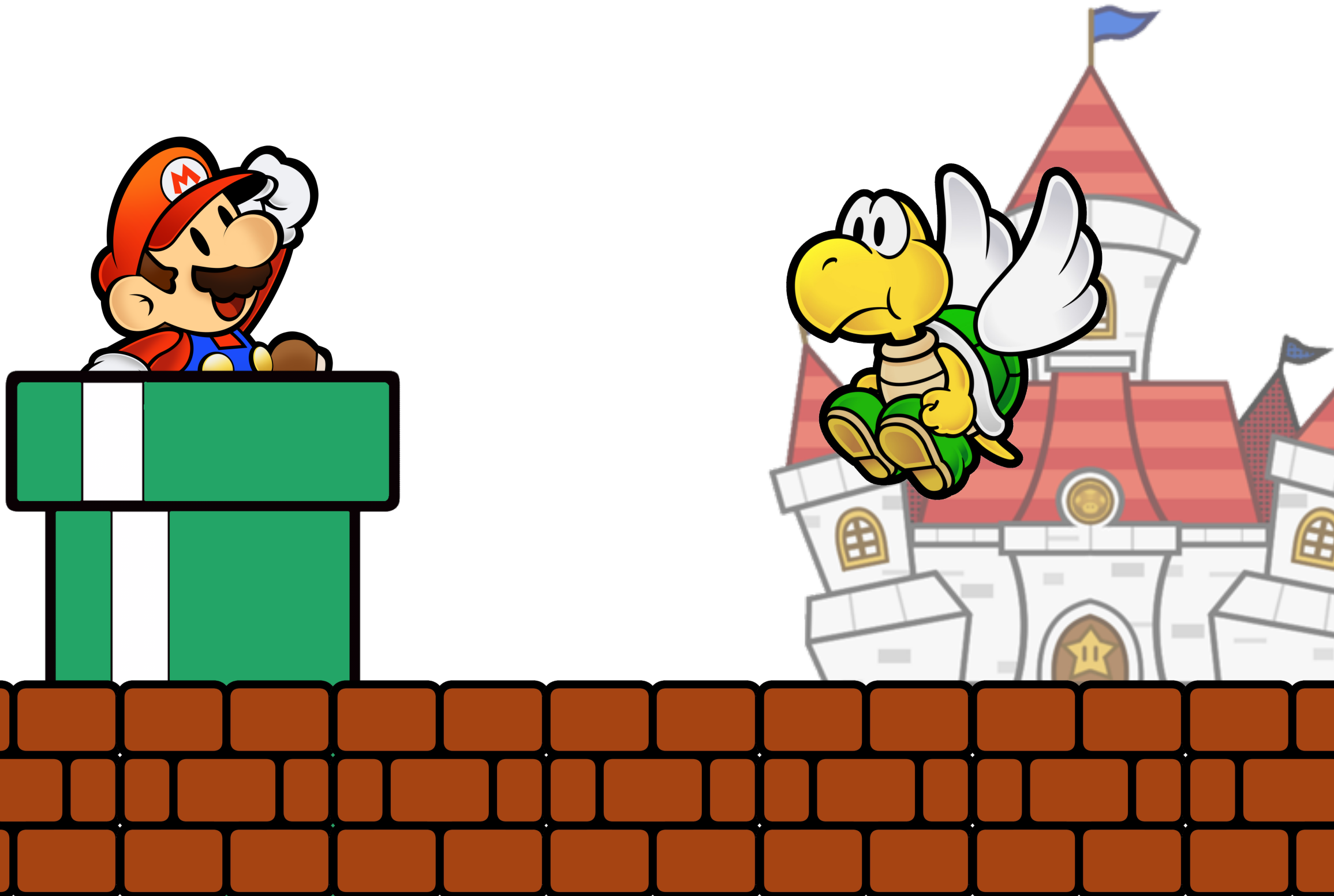
Format poche 6,90 €

Format Kinder 6,49 €



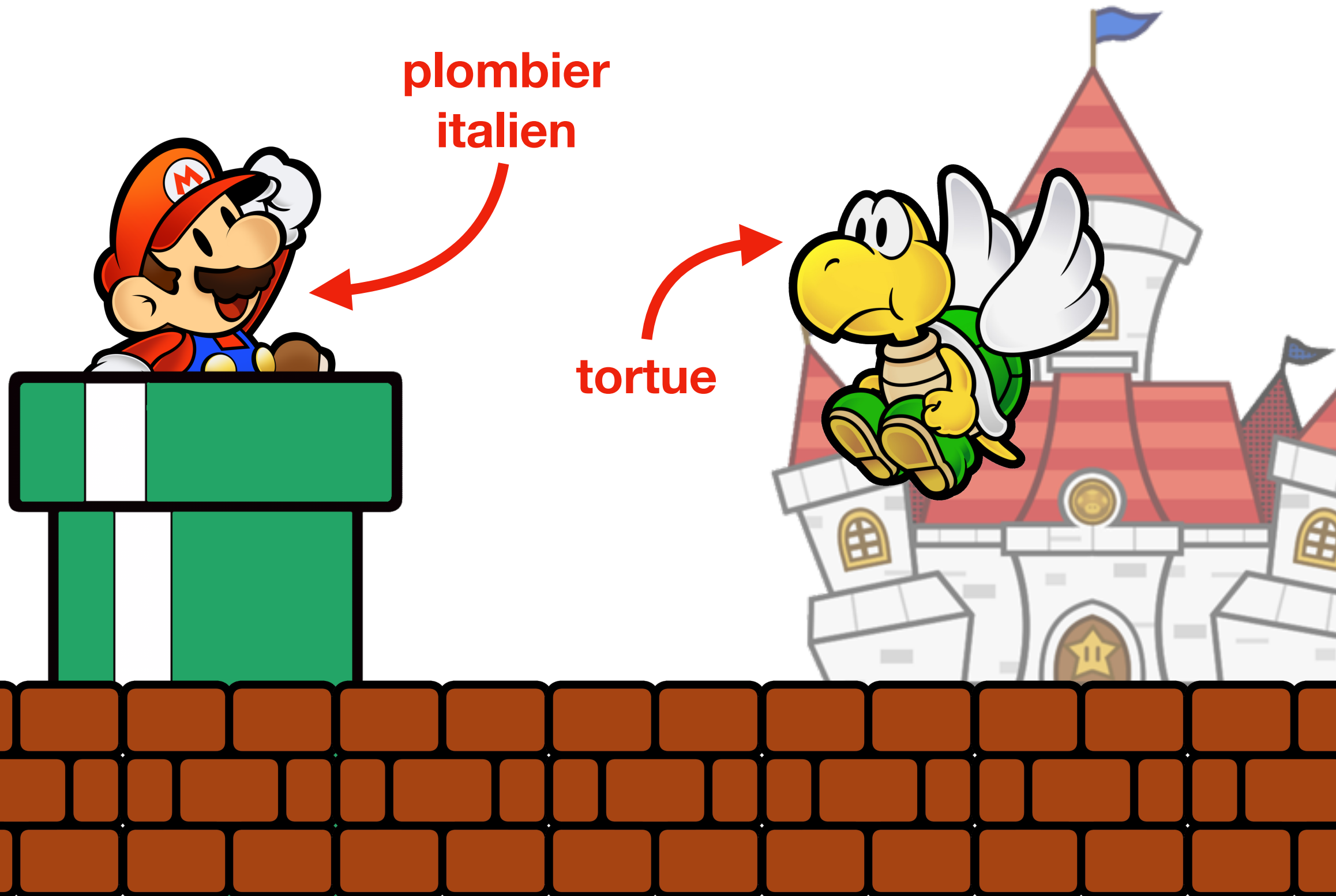
Algos de tri dans le jeux vidéo

« Super Plombiers Italiens »



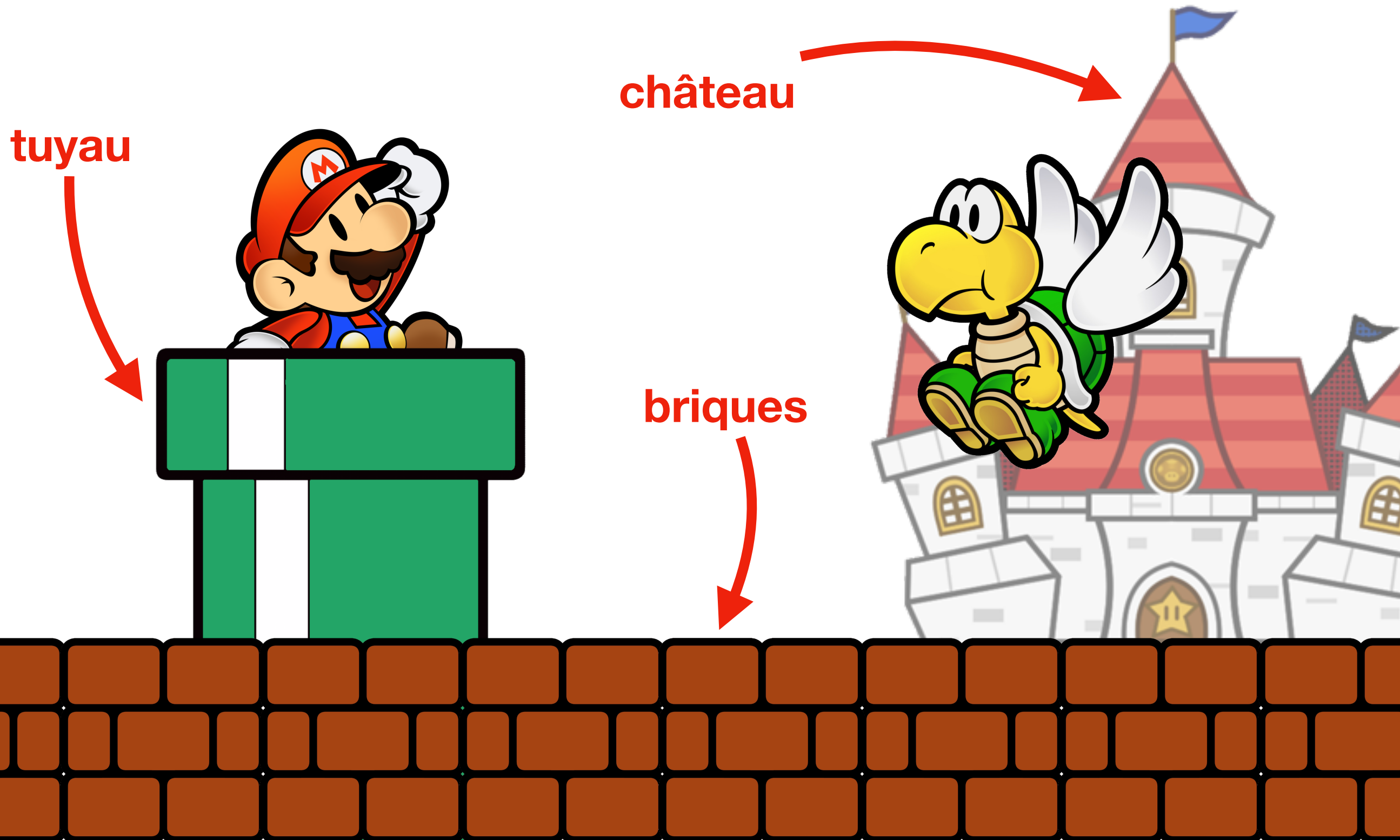
Algos de tri dans le jeux vidéo

« Super Plombiers Italiens »

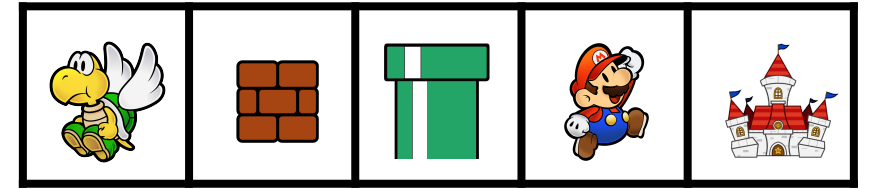


Algos de tri dans le jeux vidéo

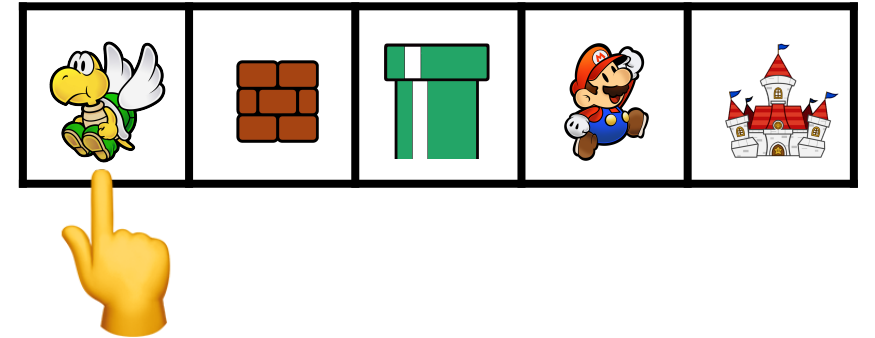
« Super Plombiers Italiens »



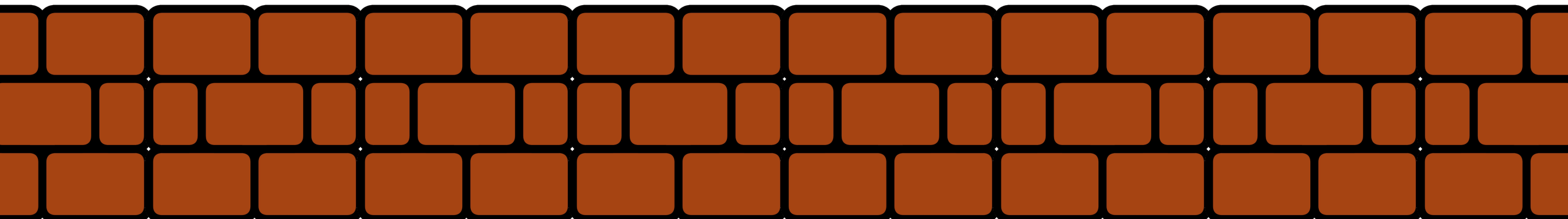
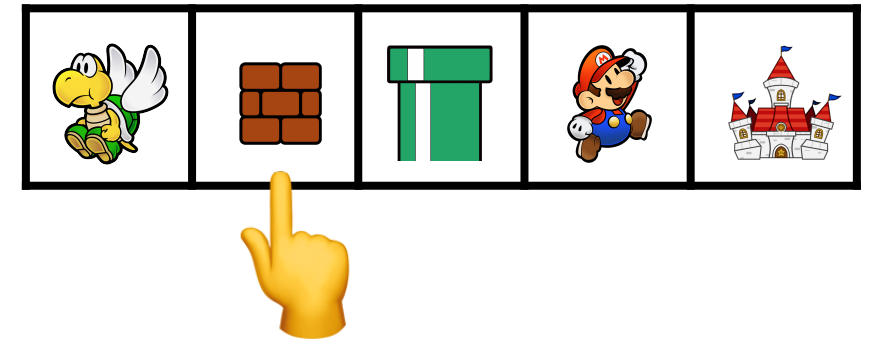
Affichage des objets
dans le **mauvais** ordre



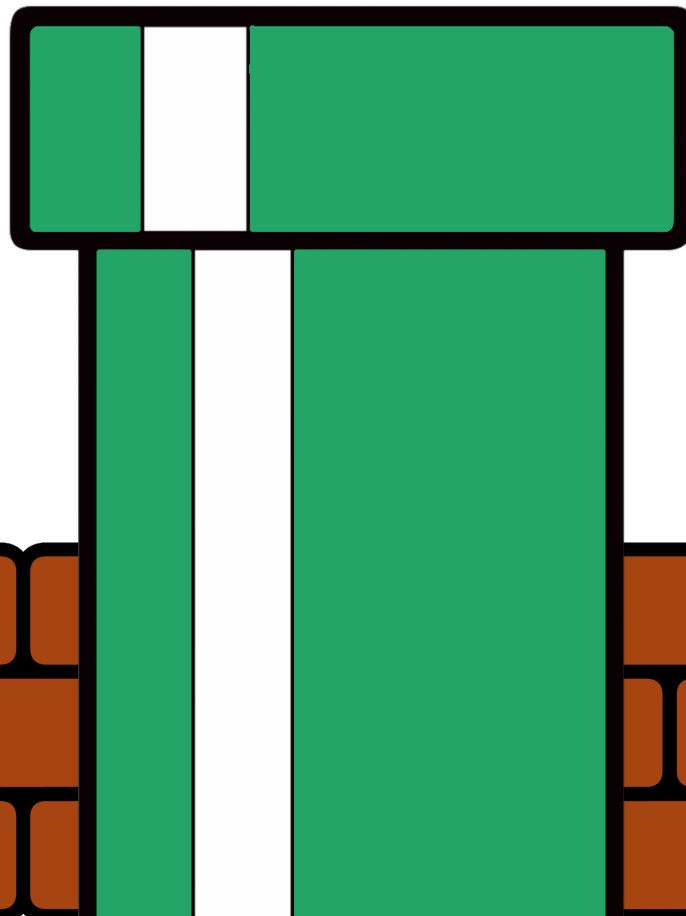
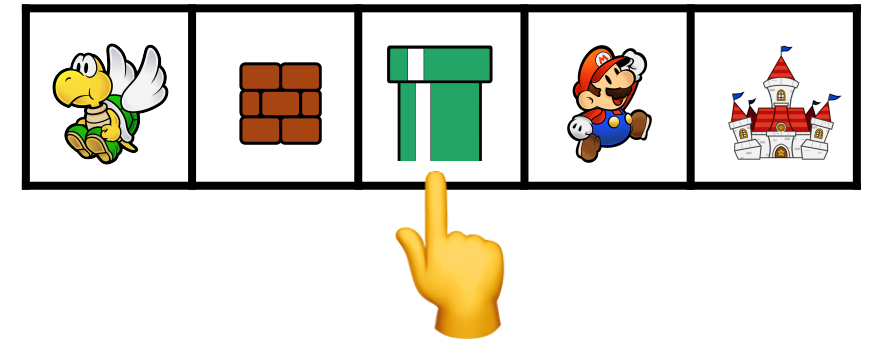
Affichage des objets
dans le **mauvais** ordre



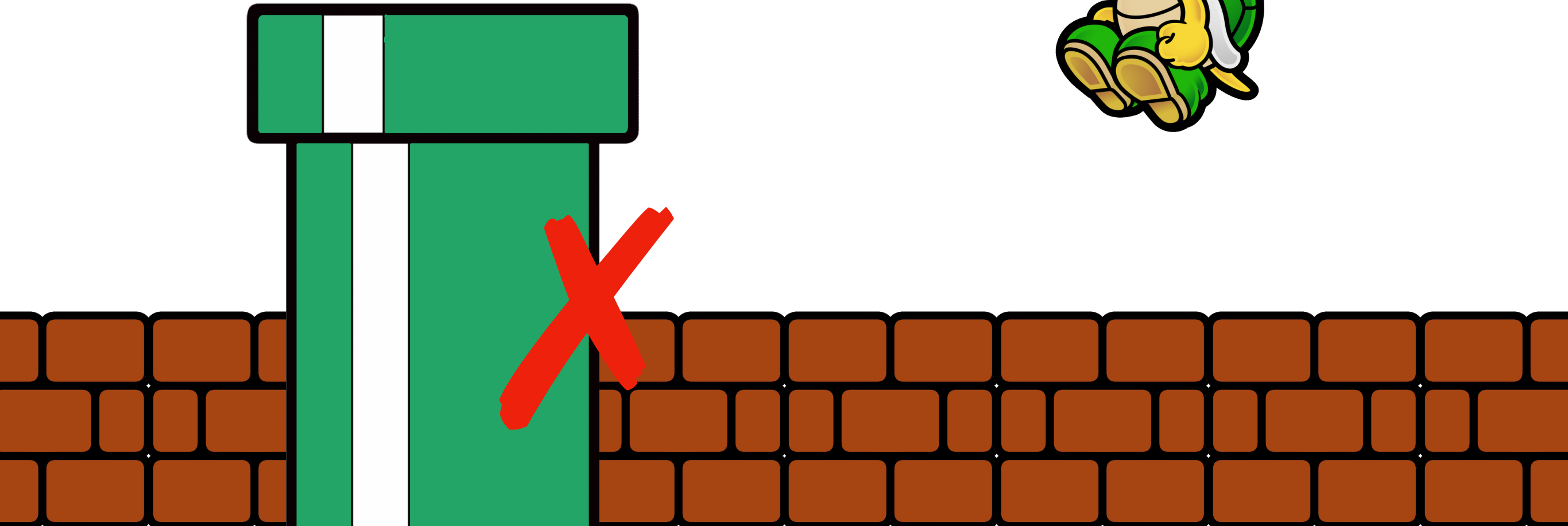
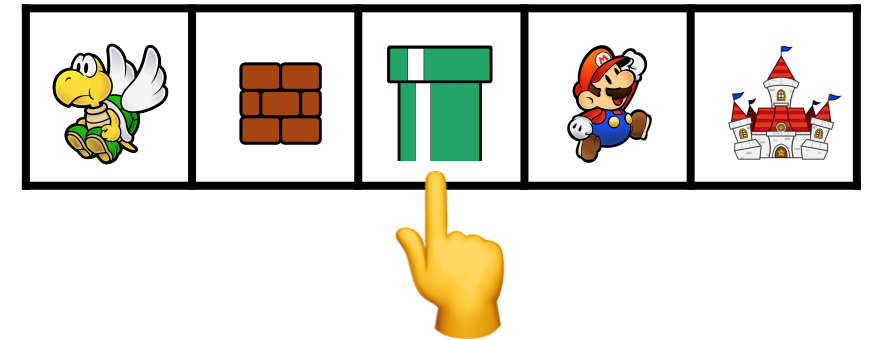
Affichage des objets
dans le **mauvais** ordre



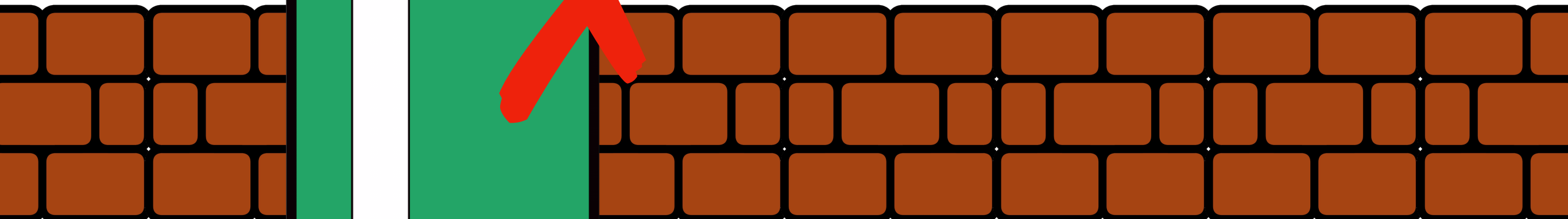
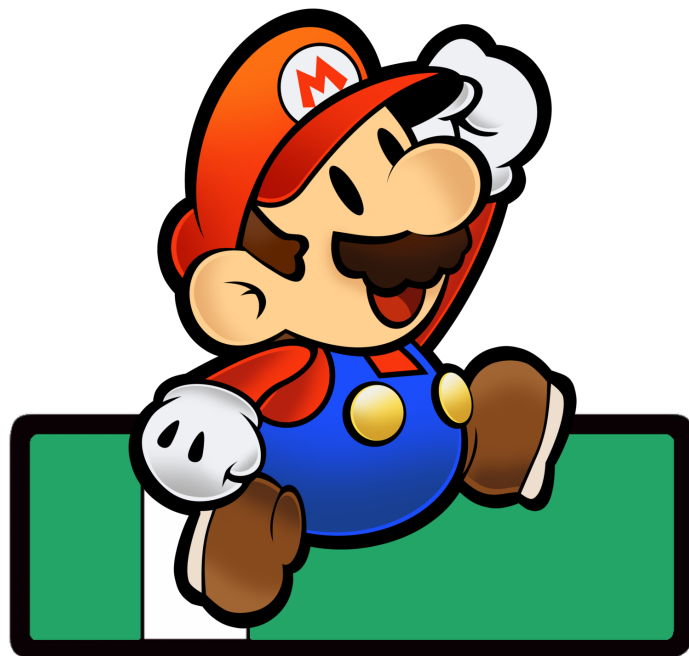
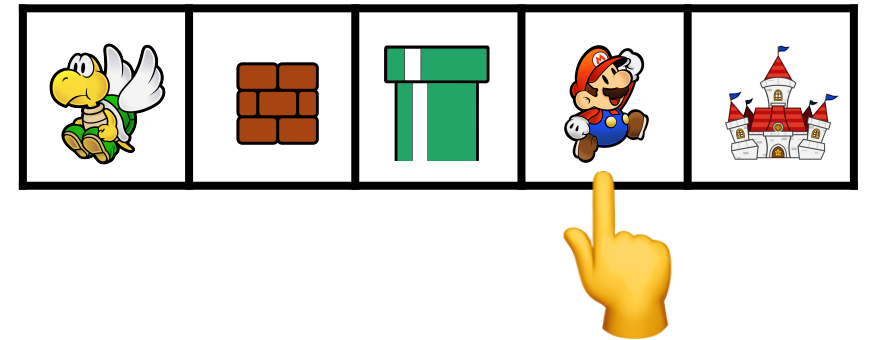
Affichage des objets
dans le **mauvais** ordre



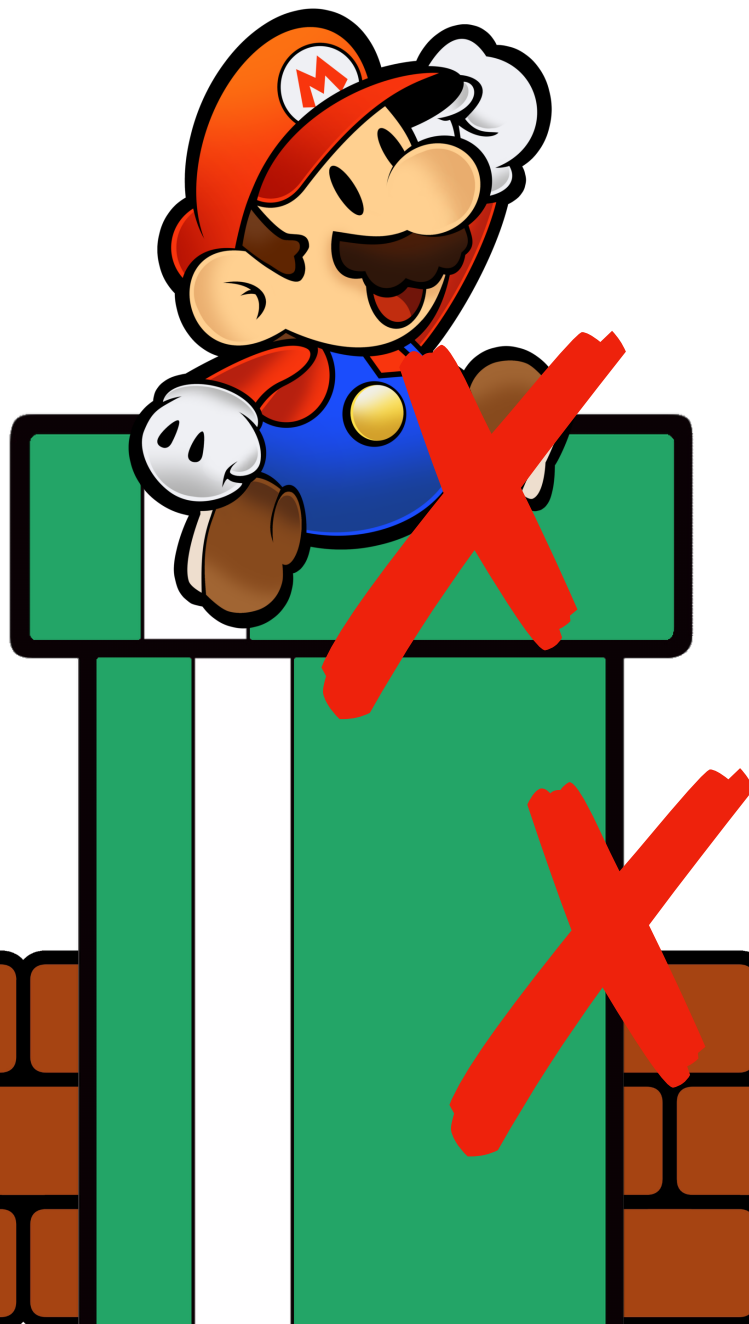
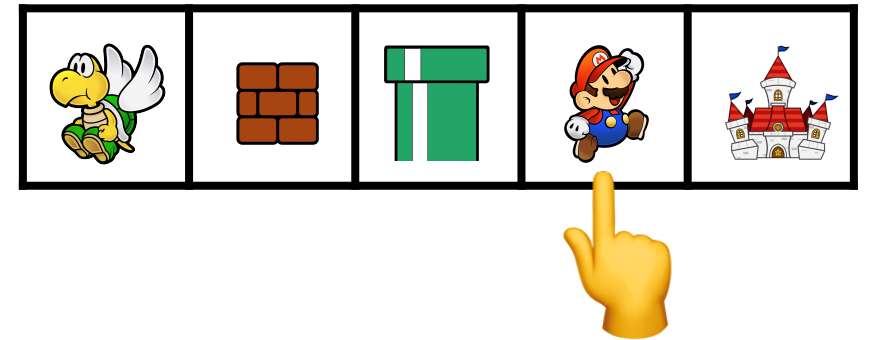
Affichage des objets
dans le **mauvais** ordre



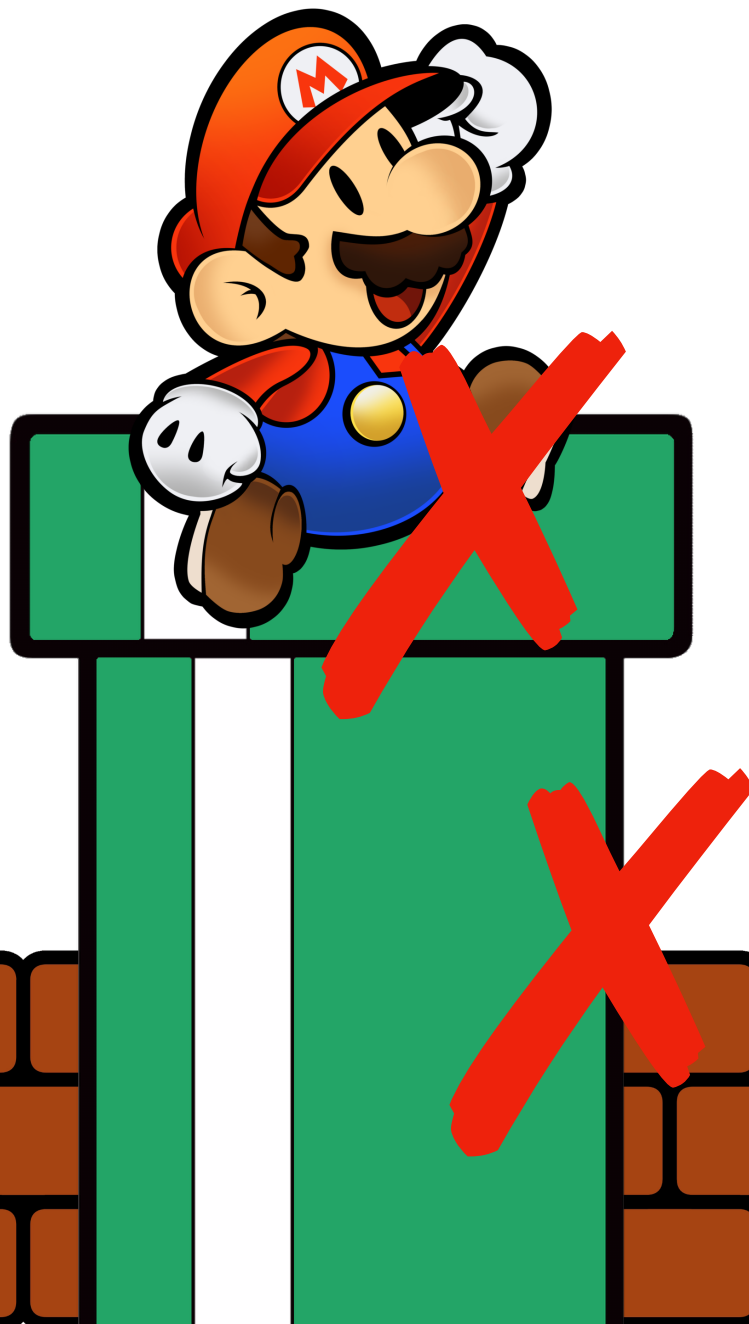
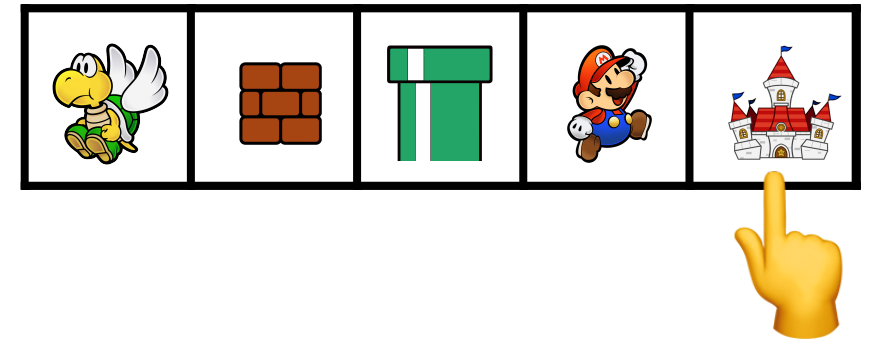
Affichage des objets
dans le **mauvais** ordre



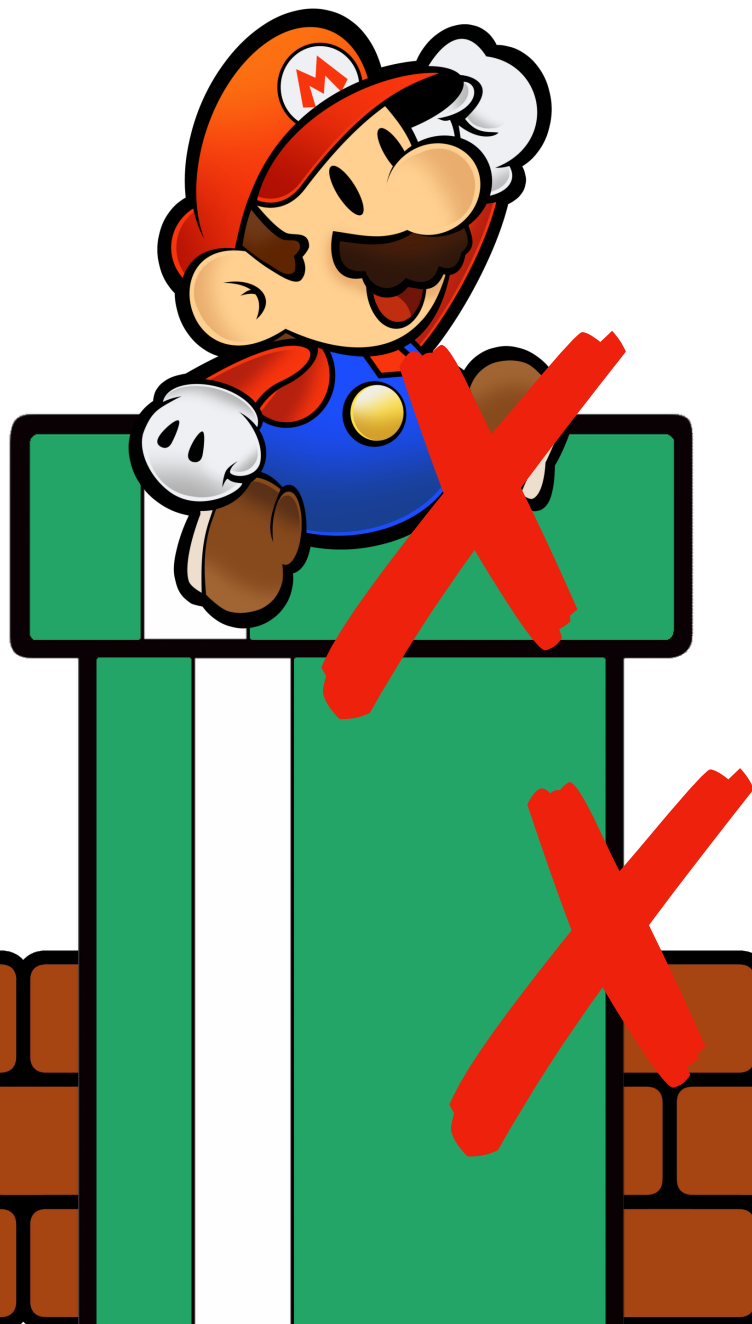
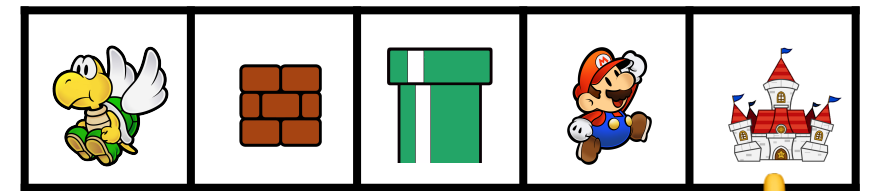
Affichage des objets
dans le **mauvais** ordre



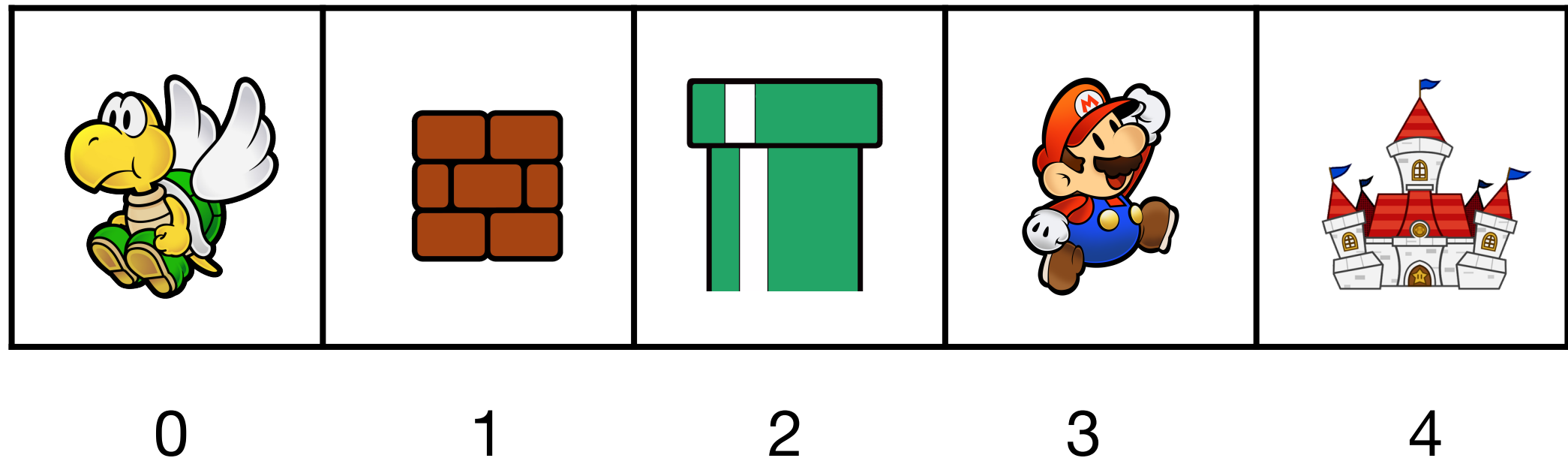
Affichage des objets
dans le **mauvais** ordre



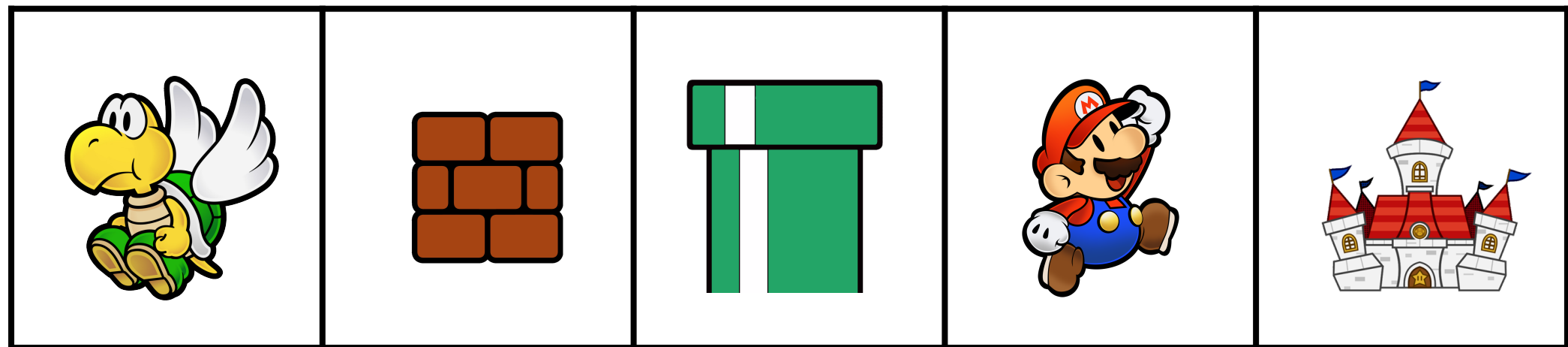
Affichage des objets
dans le **mauvais** ordre



Affichage des objets dans le mauvais ordre



Affichage des objets dans le **mauvais** ordre



0

1

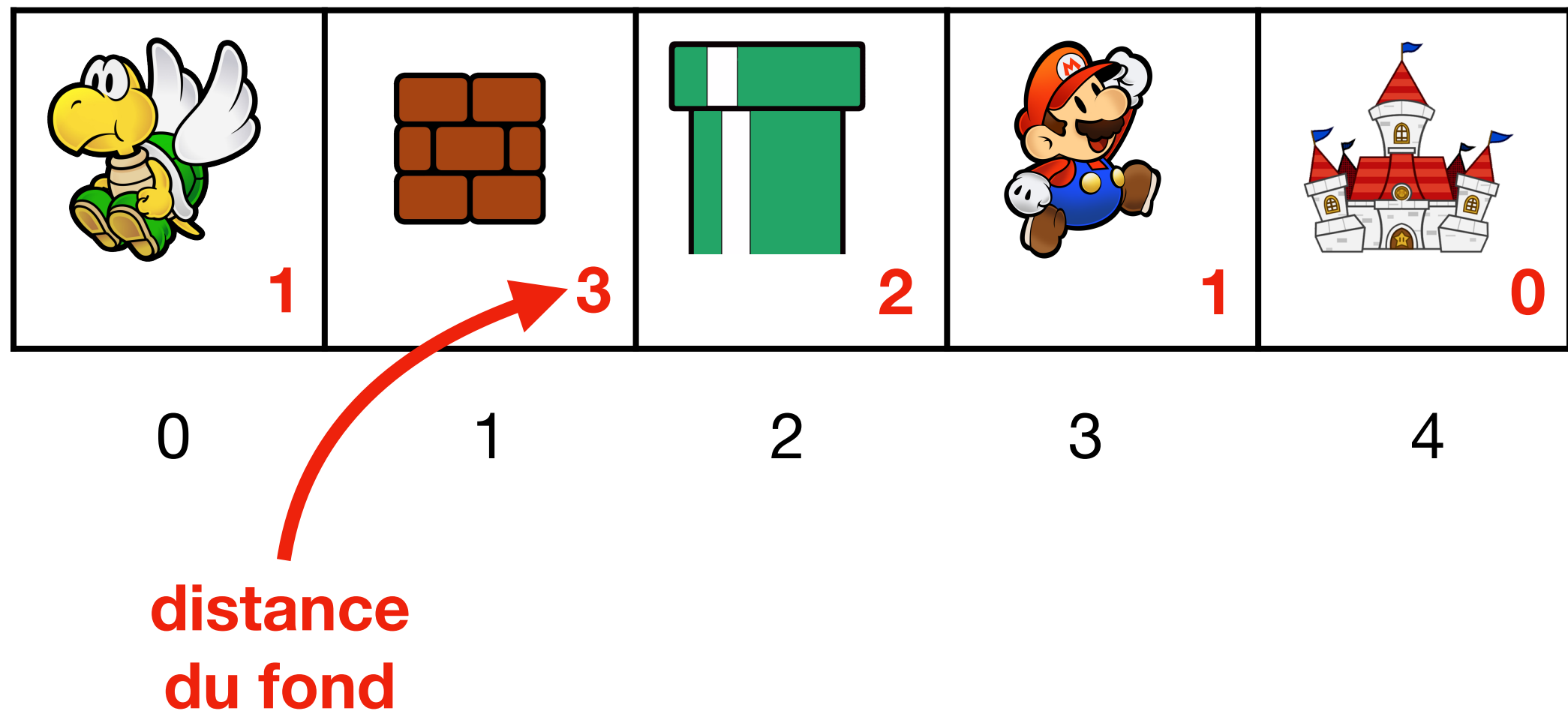
2

3

4


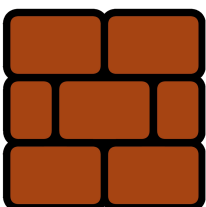
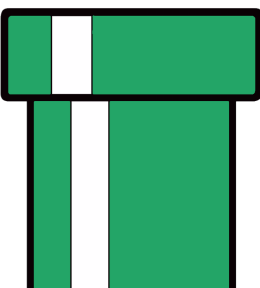


**ordre
d'affichage**

Affichage des objets dans le mauvais ordre




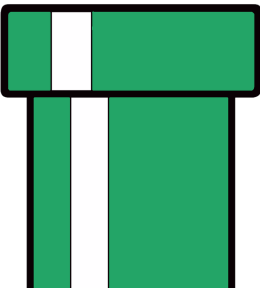
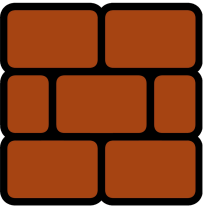


Tri ! 

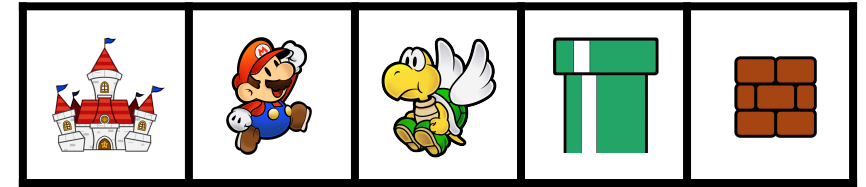
Affichage des objets dans le mauvais ordre

 1	 3	 2	 1	 0
0	1	2	3	4

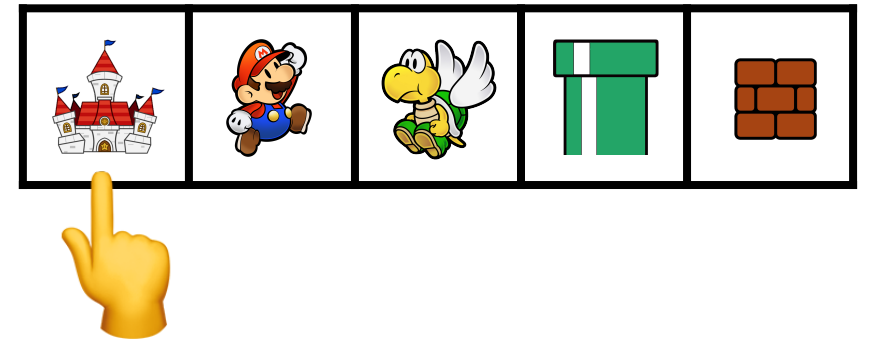
Affichage des objets dans le bon ordre

 0	 1	 1	 2	 3
0	1	2	3	4

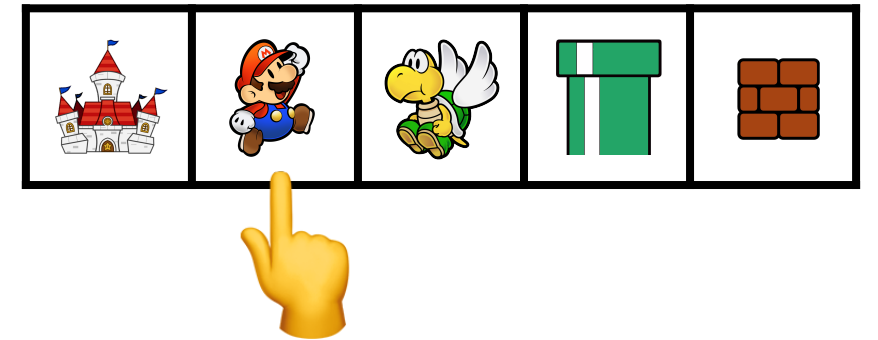
Affichage des objets dans le bon ordre



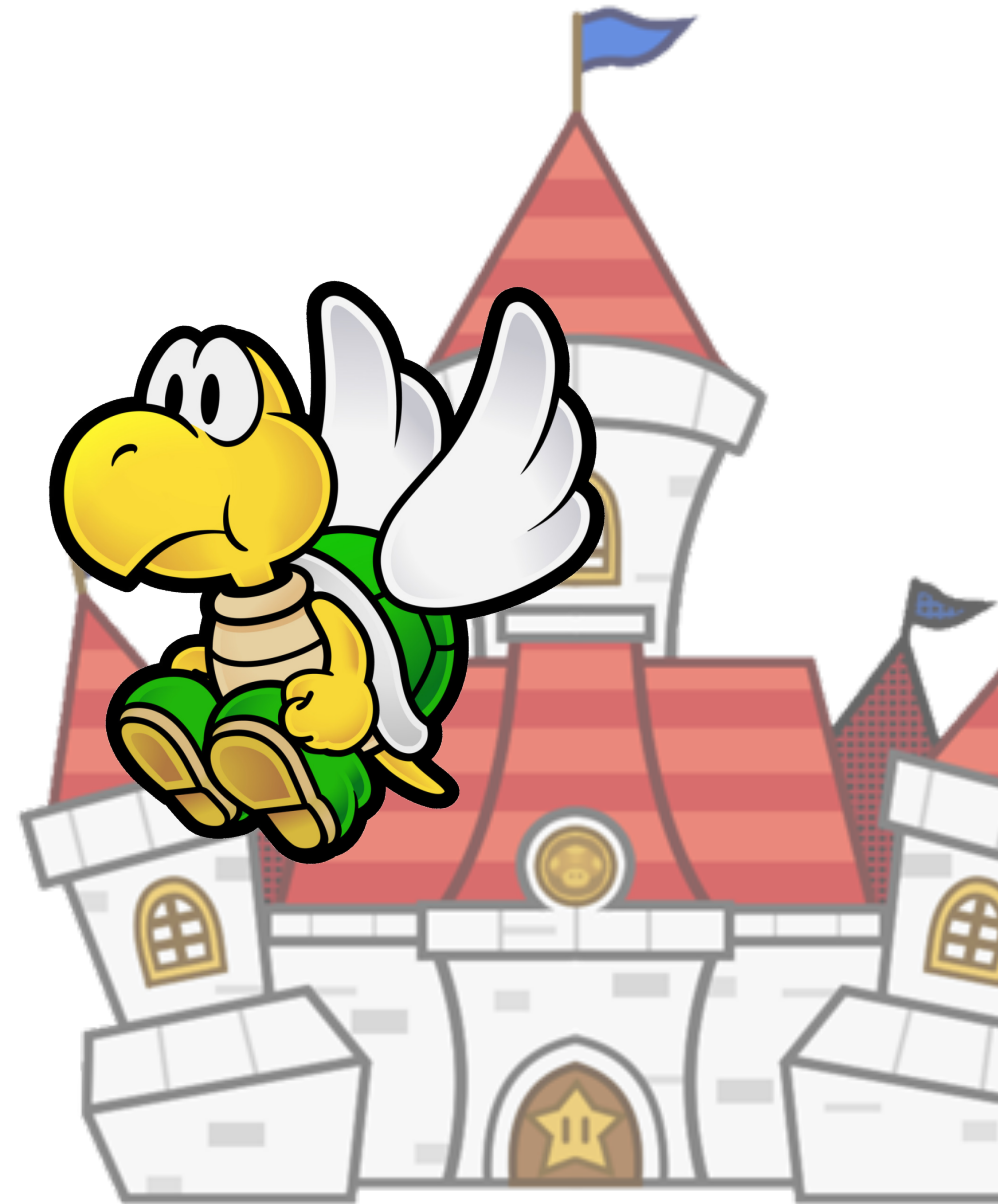
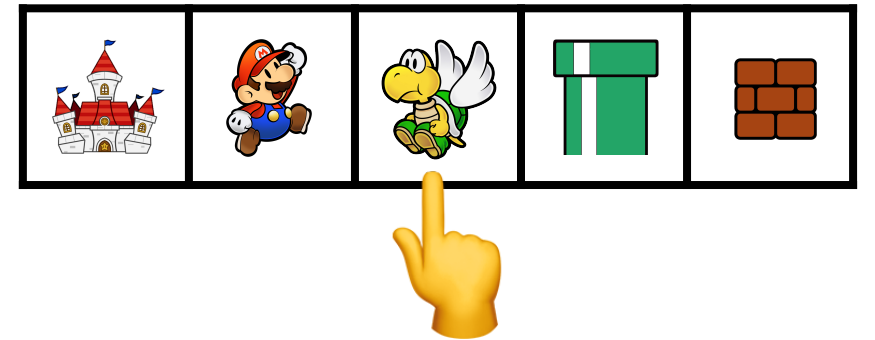
Affichage des objets dans le **bon** ordre



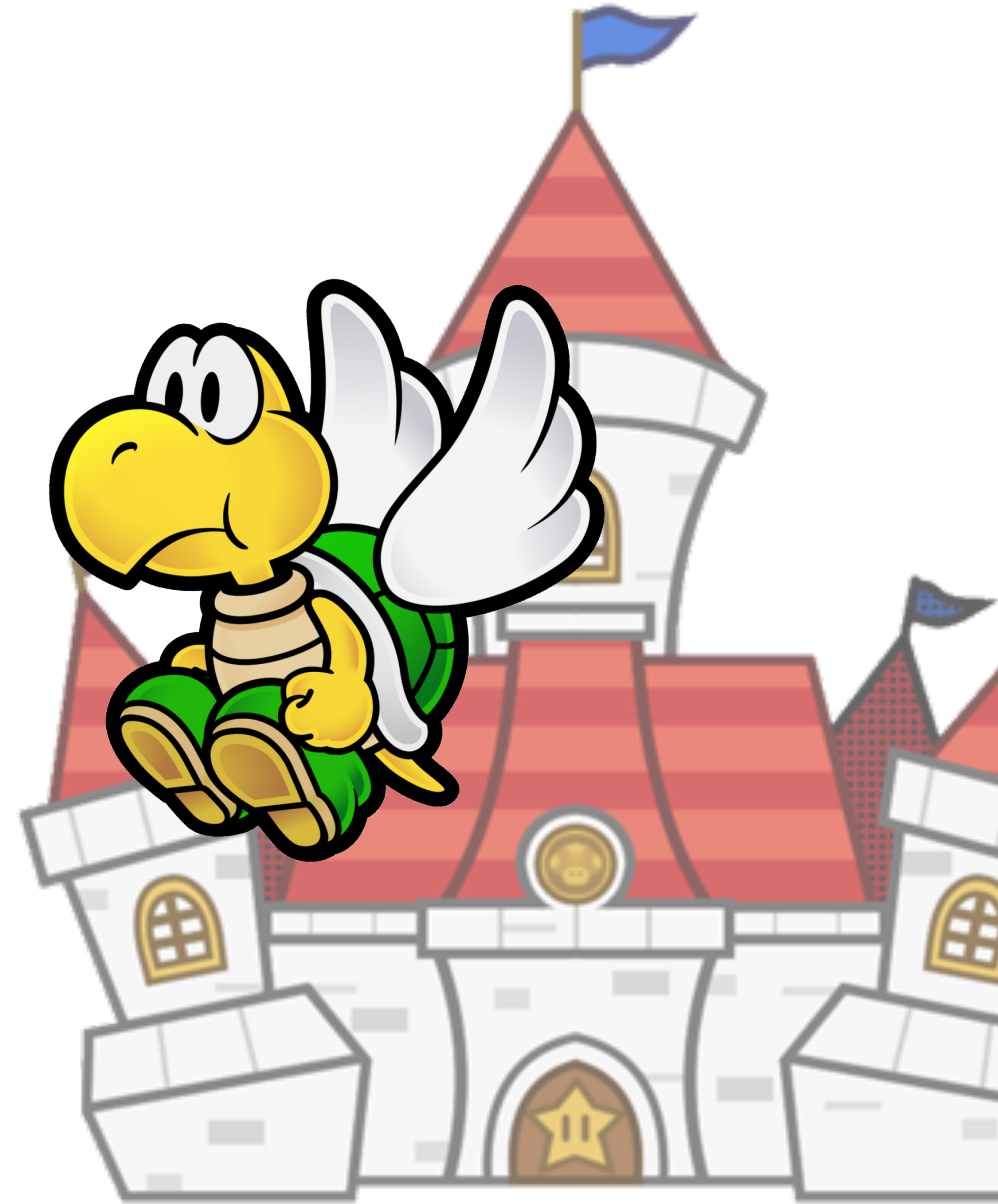
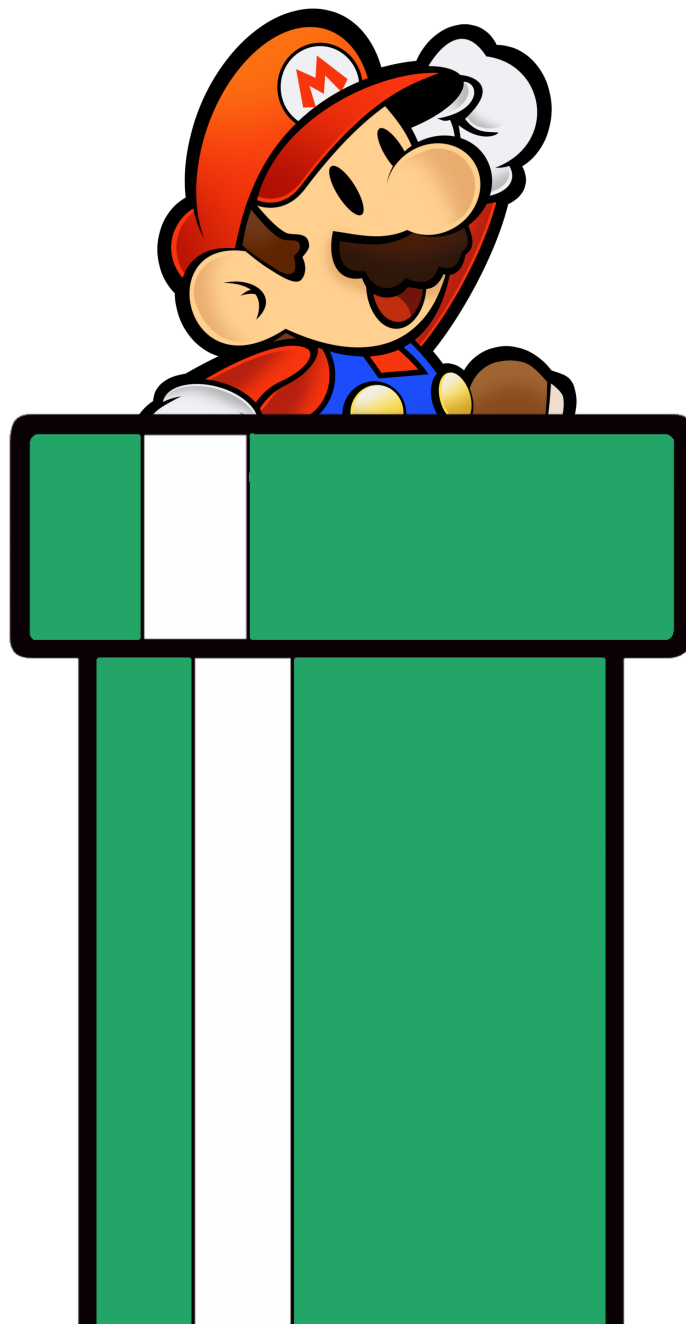
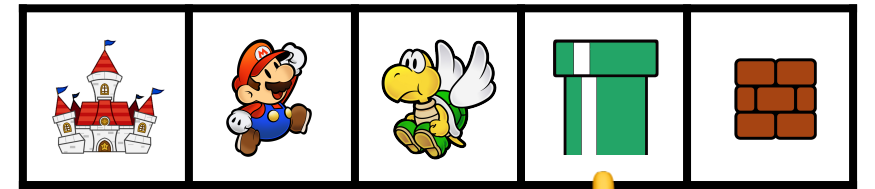
Affichage des objets dans le bon ordre



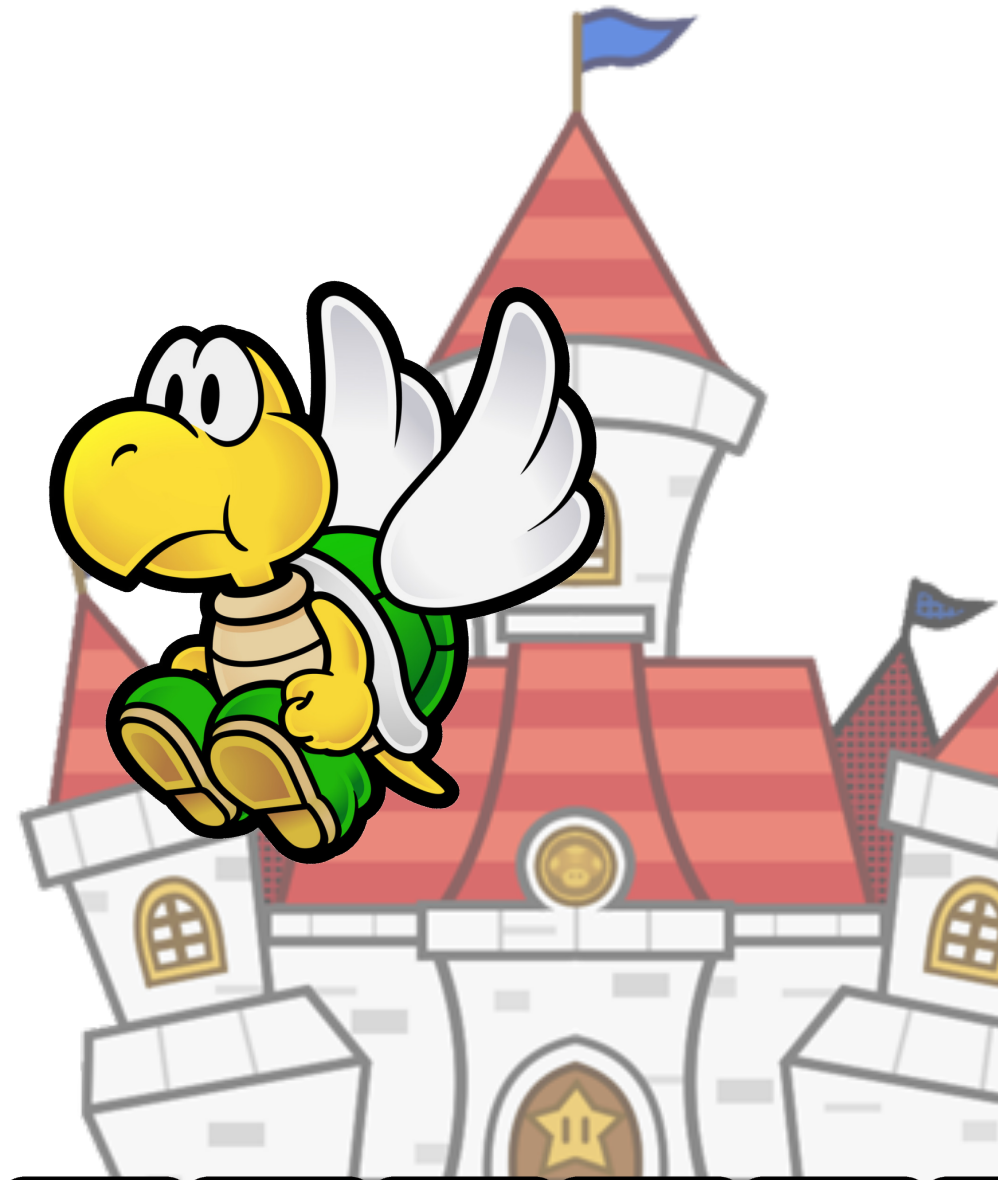
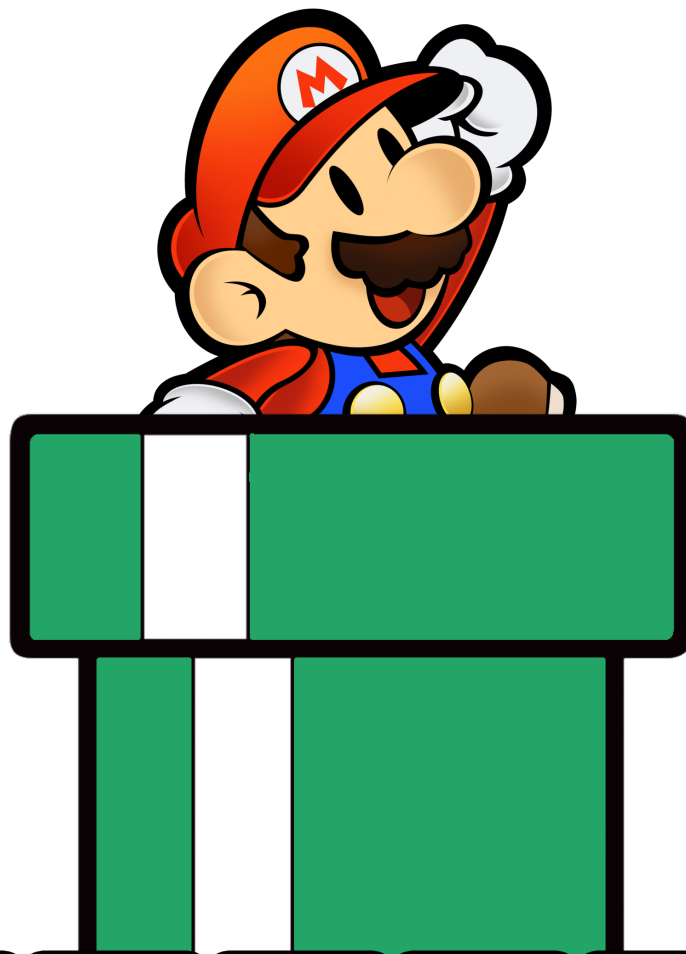
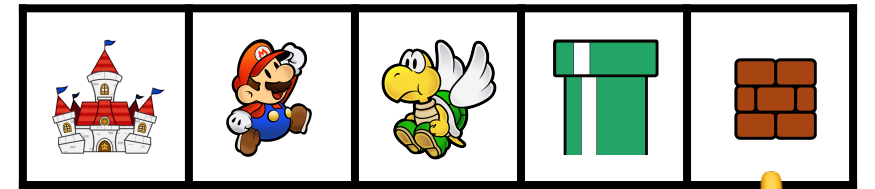
Affichage des objets dans le bon ordre



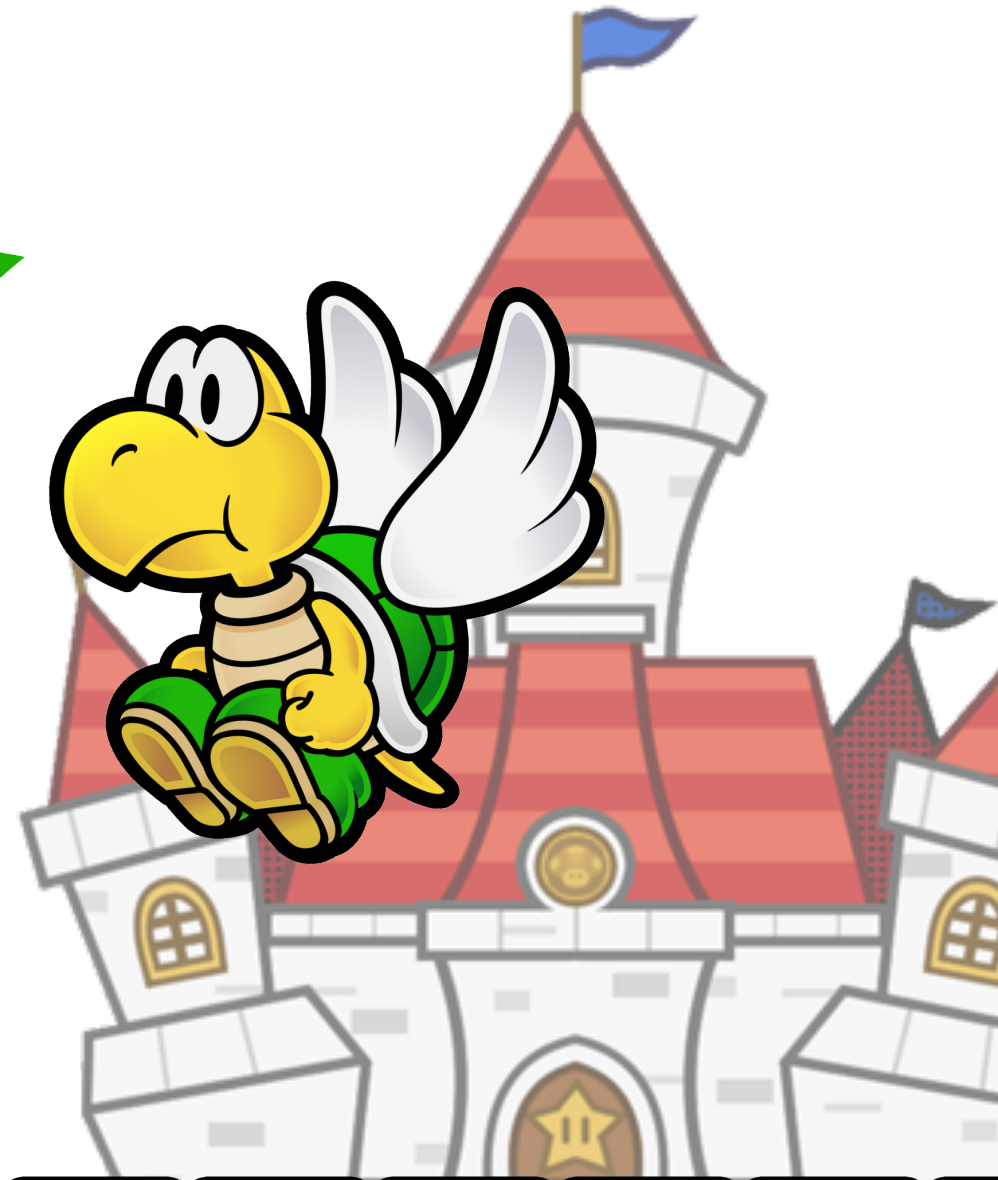
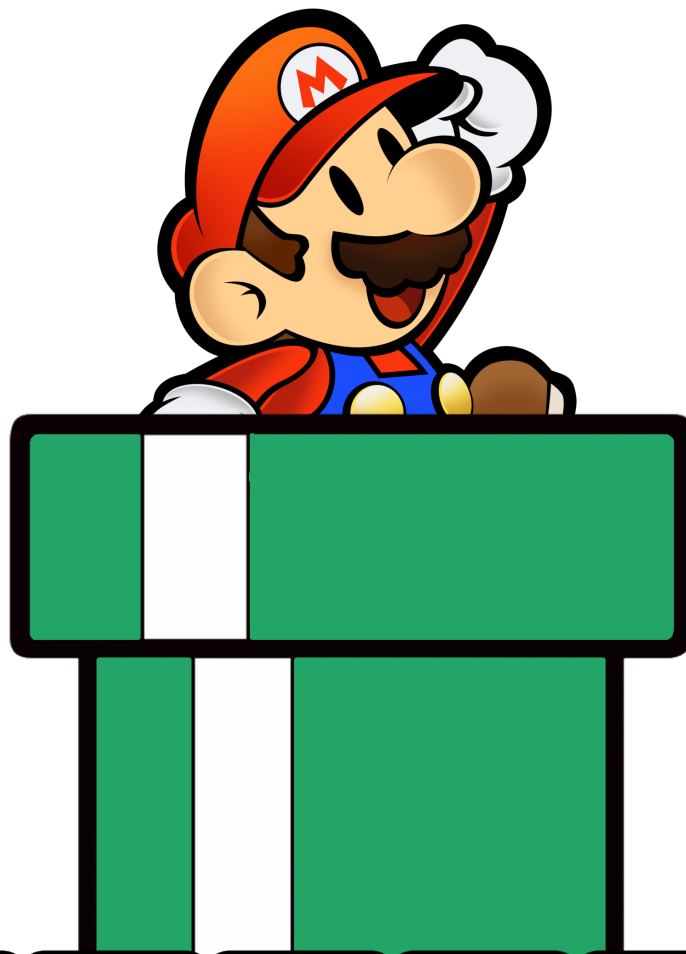
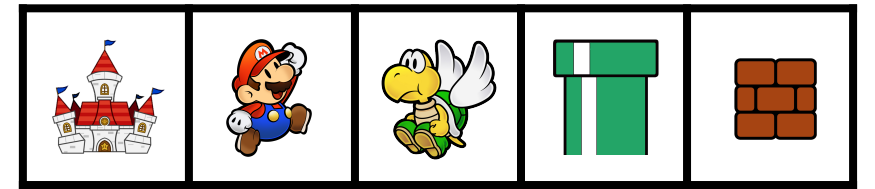
Affichage des objets dans le bon ordre



Affichage des objets dans le bon ordre

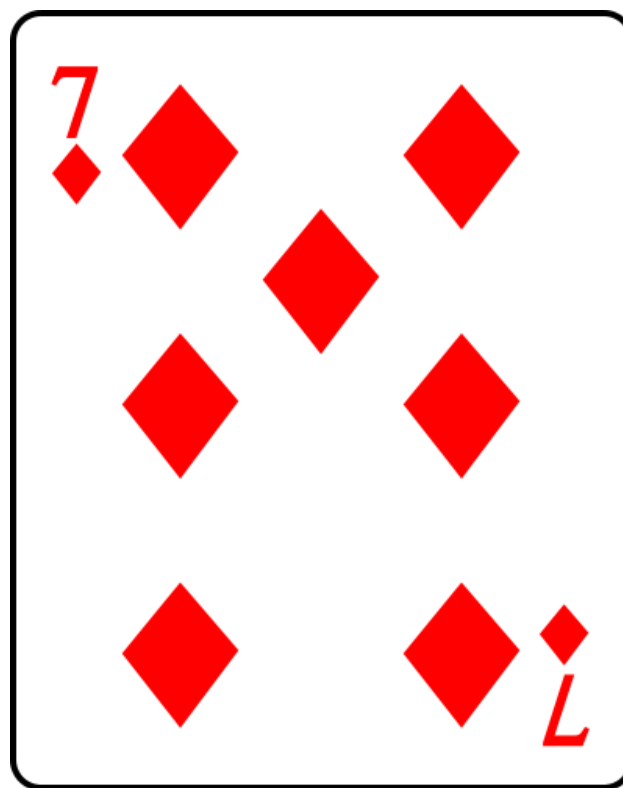


Affichage des objets dans le bon ordre

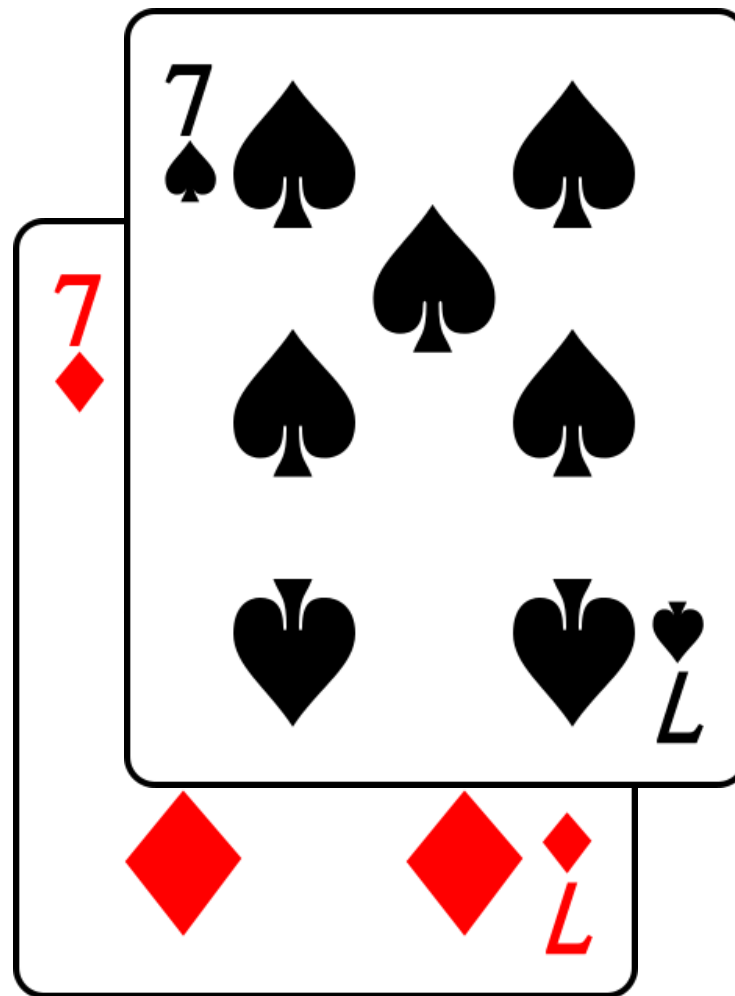


Tri par insertion

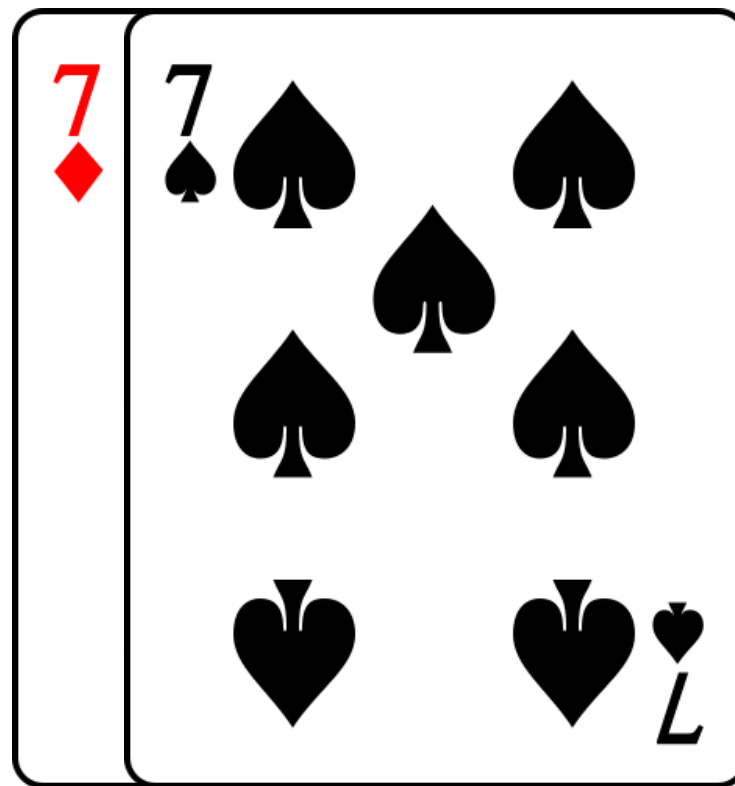
Tri par insertion



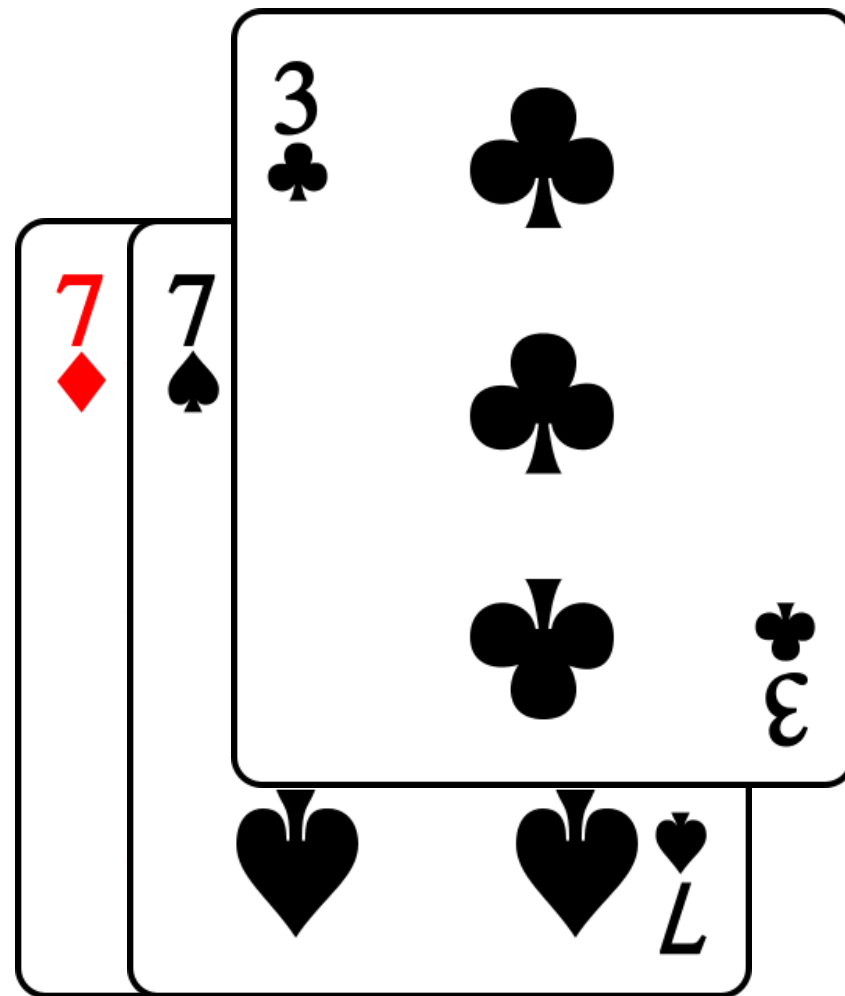
Tri par insertion



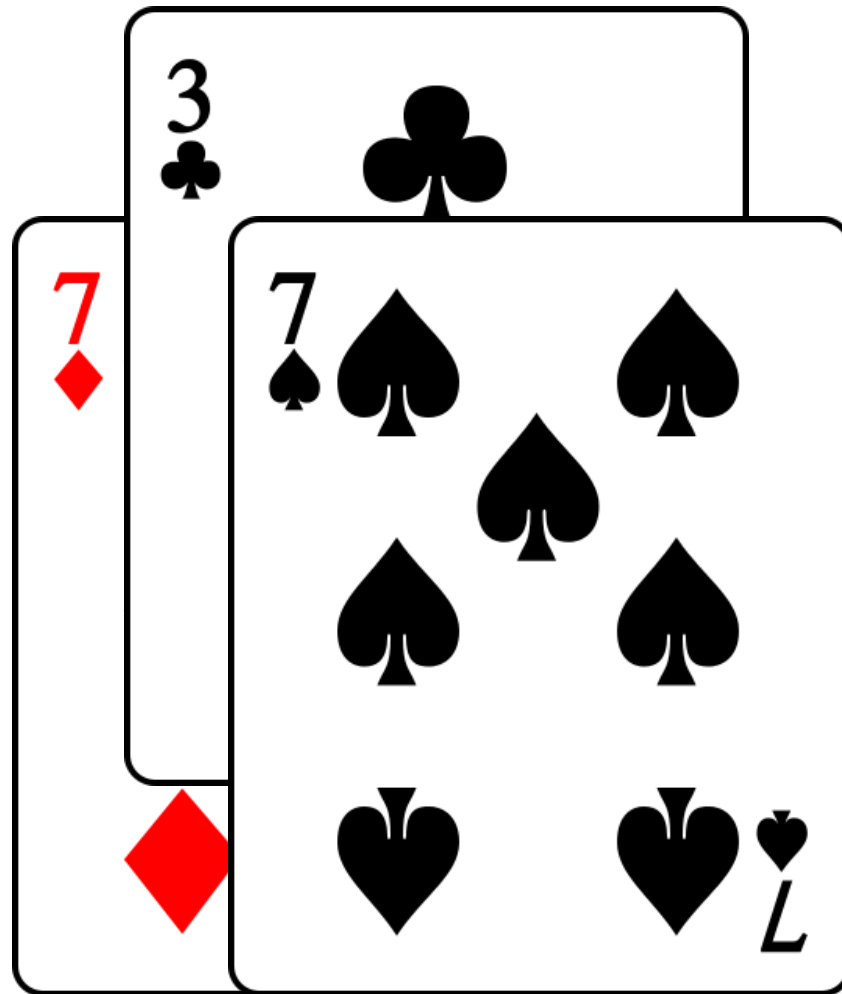
Tri par insertion



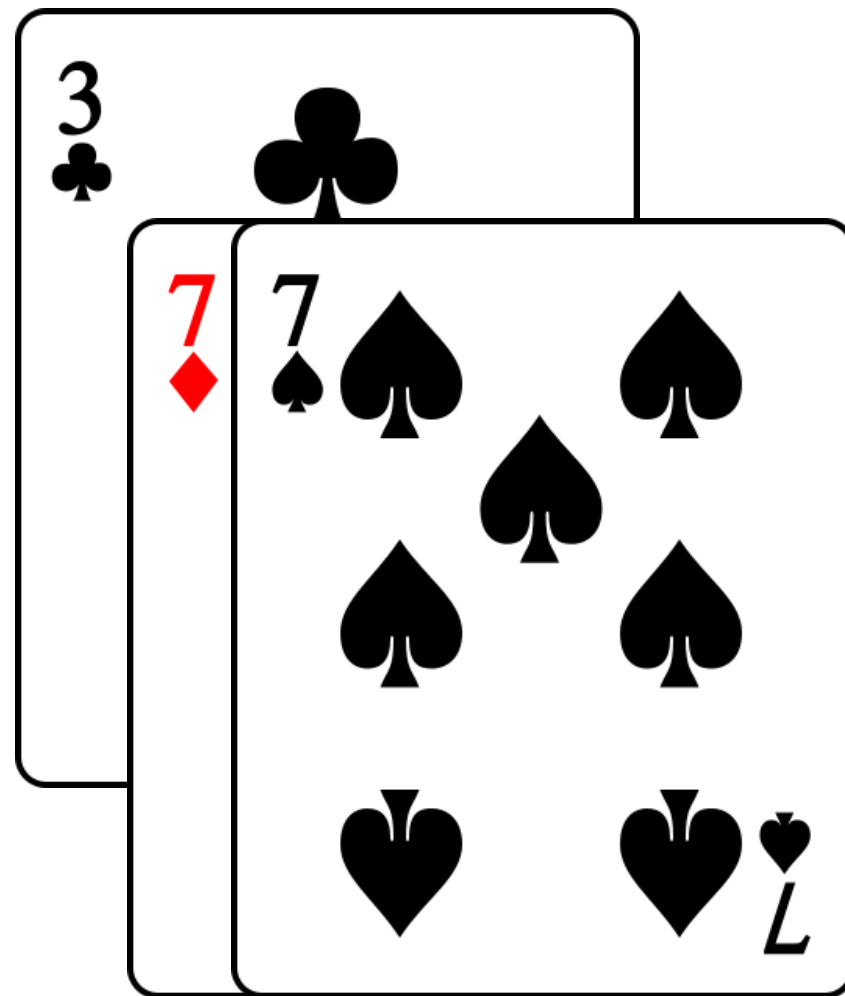
Tri par insertion



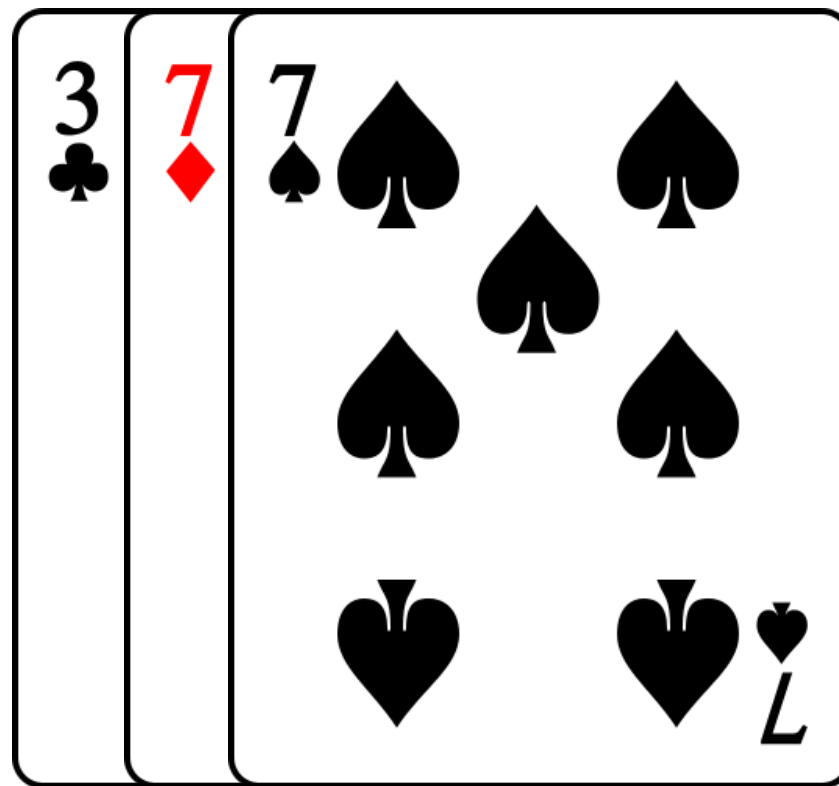
Tri par insertion



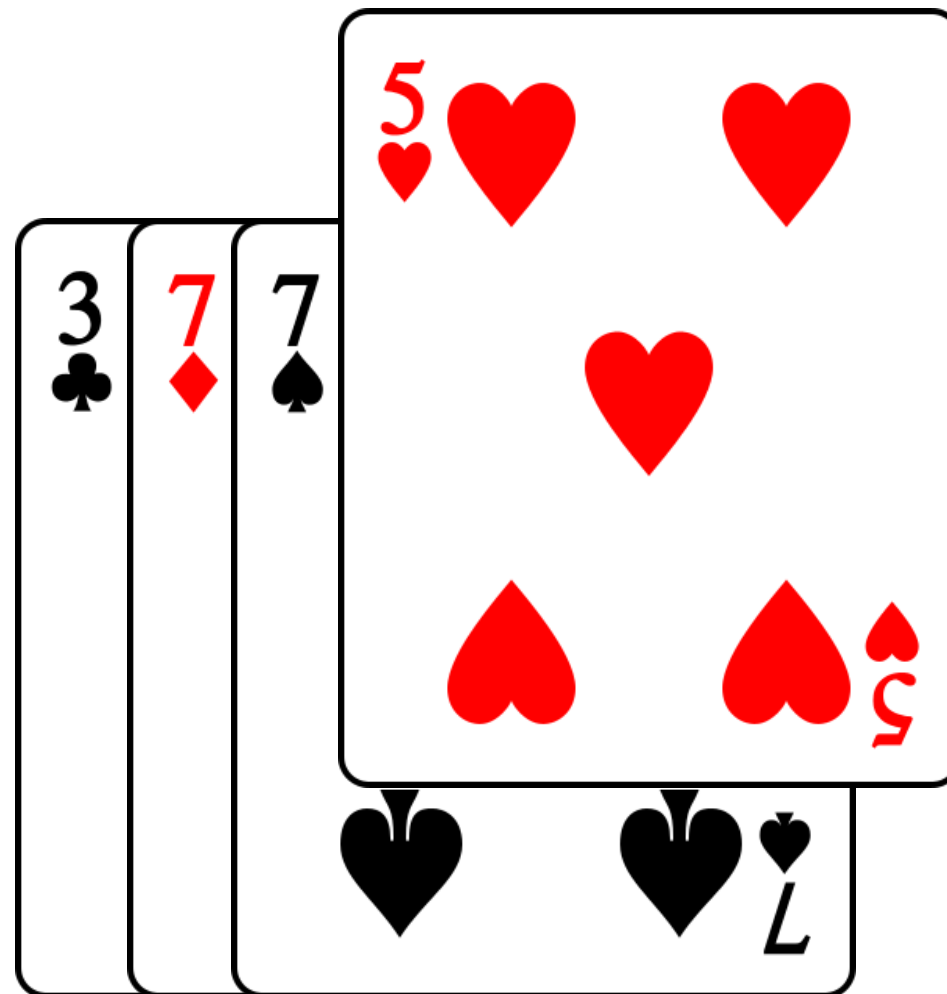
Tri par insertion



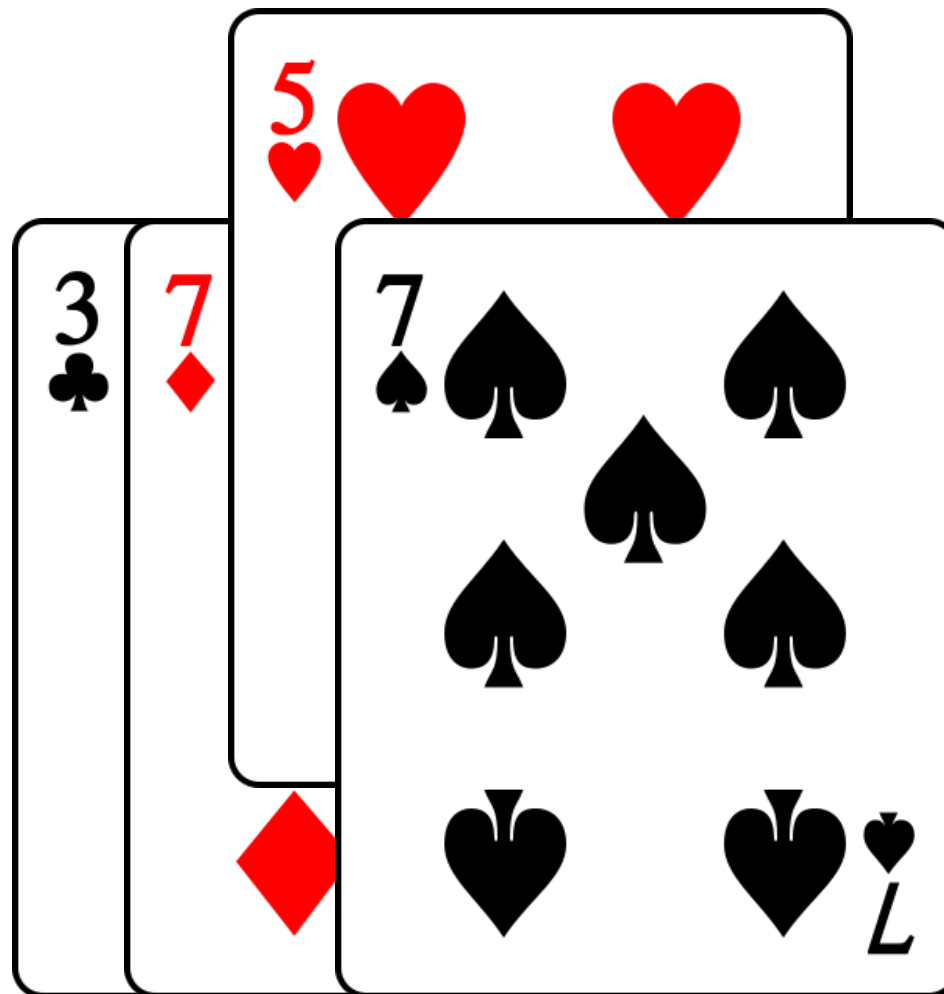
Tri par insertion



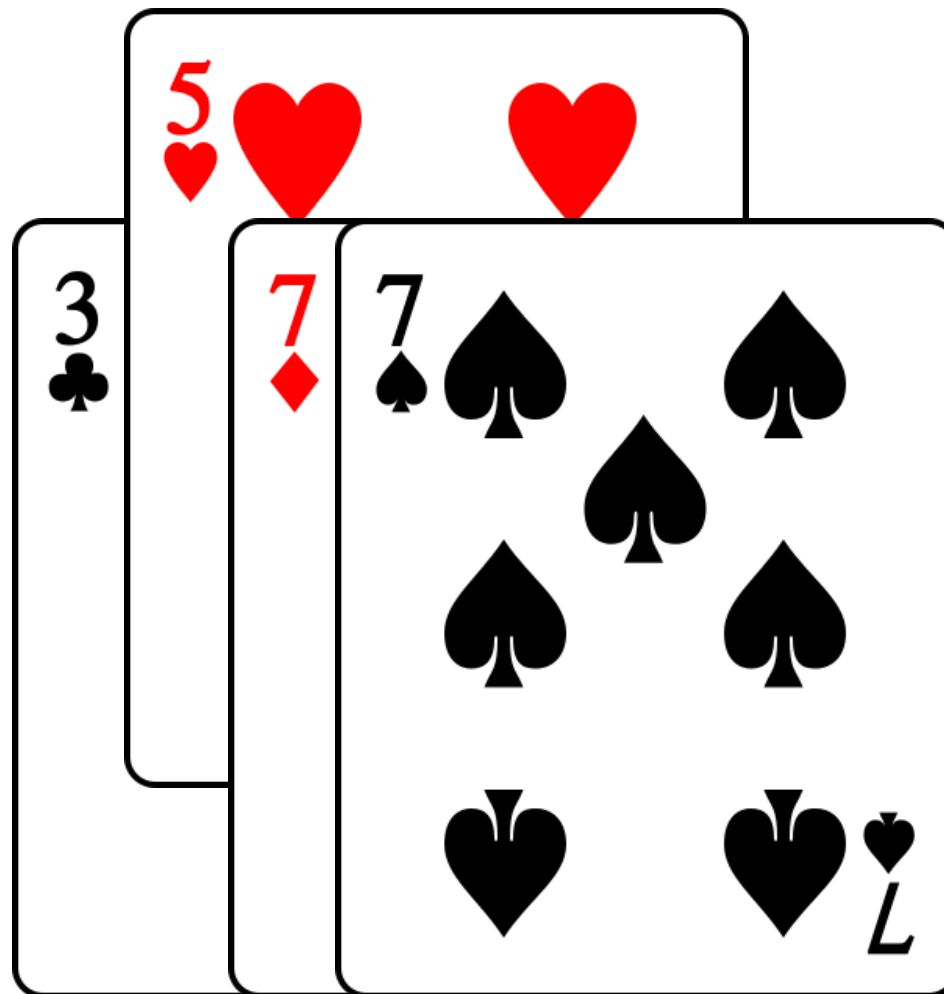
Tri par insertion



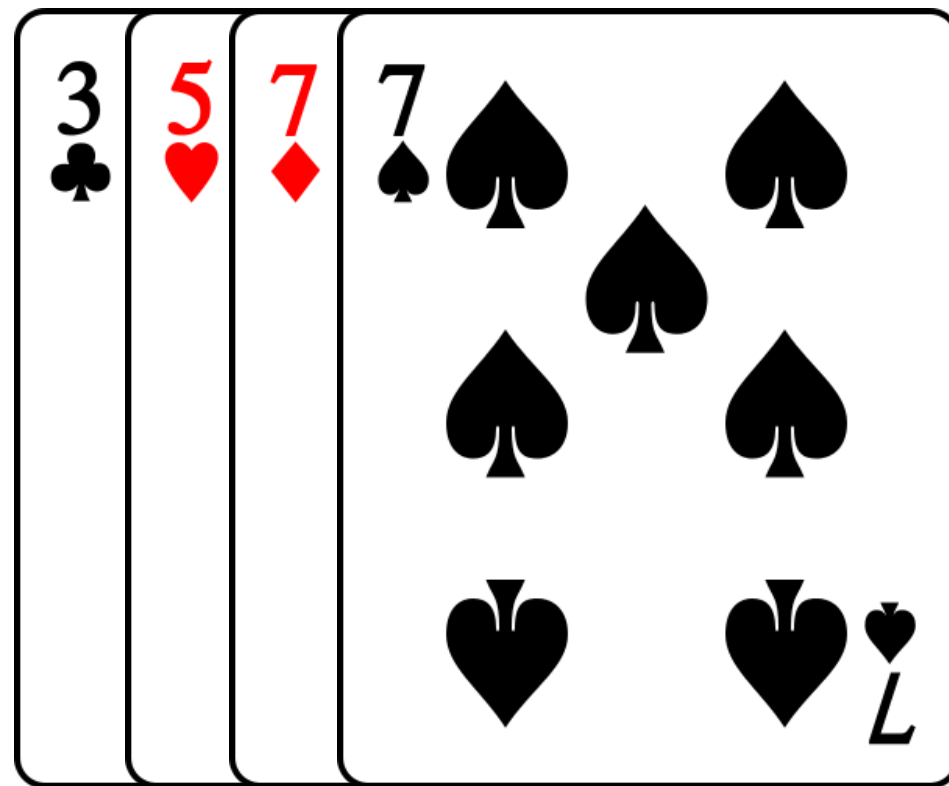
Tri par insertion



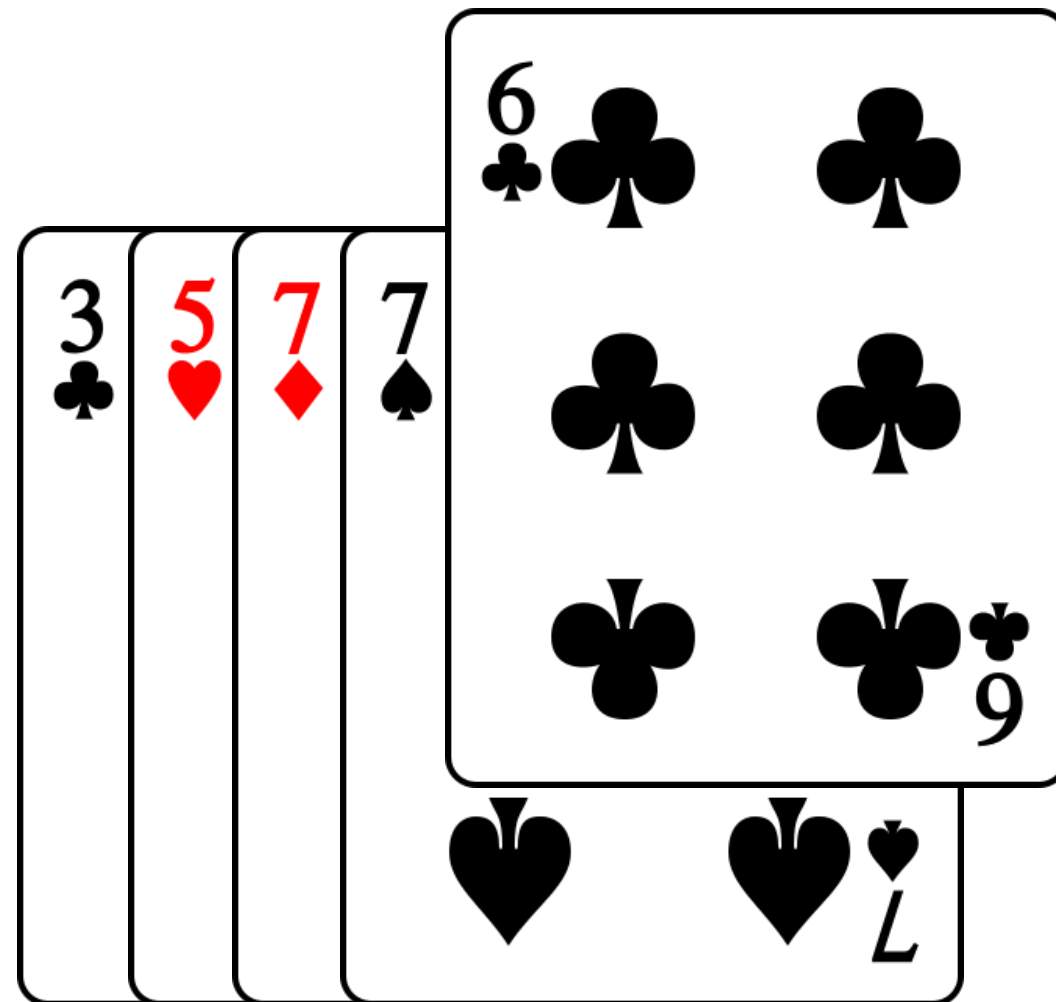
Tri par insertion



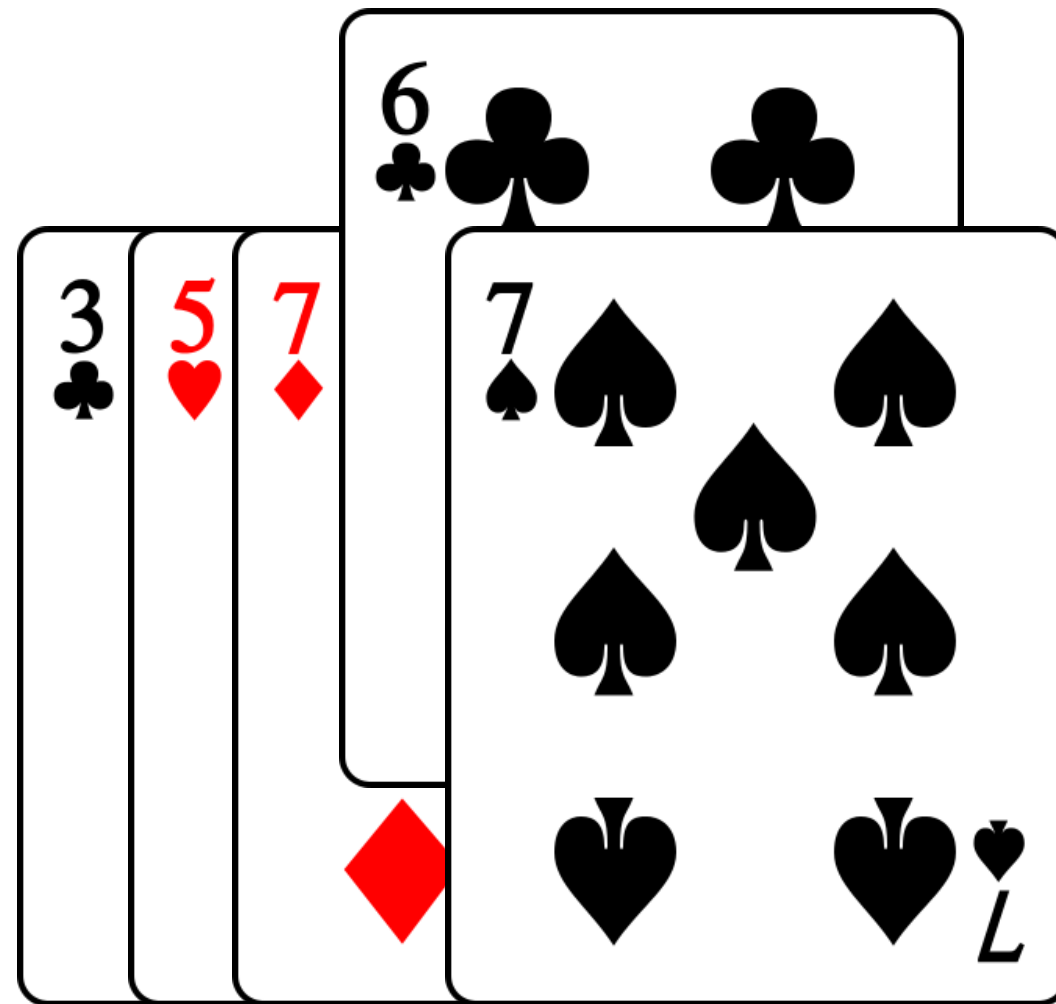
Tri par insertion



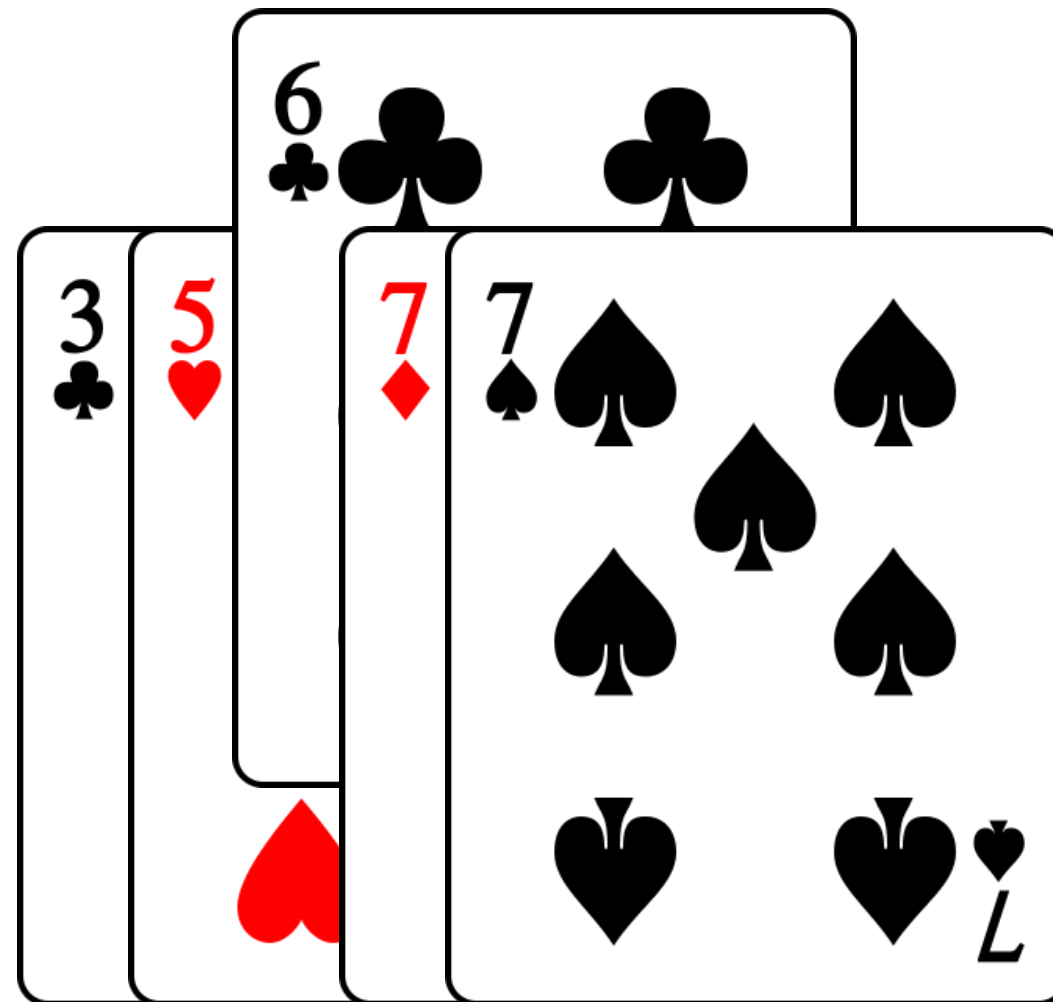
Tri par insertion



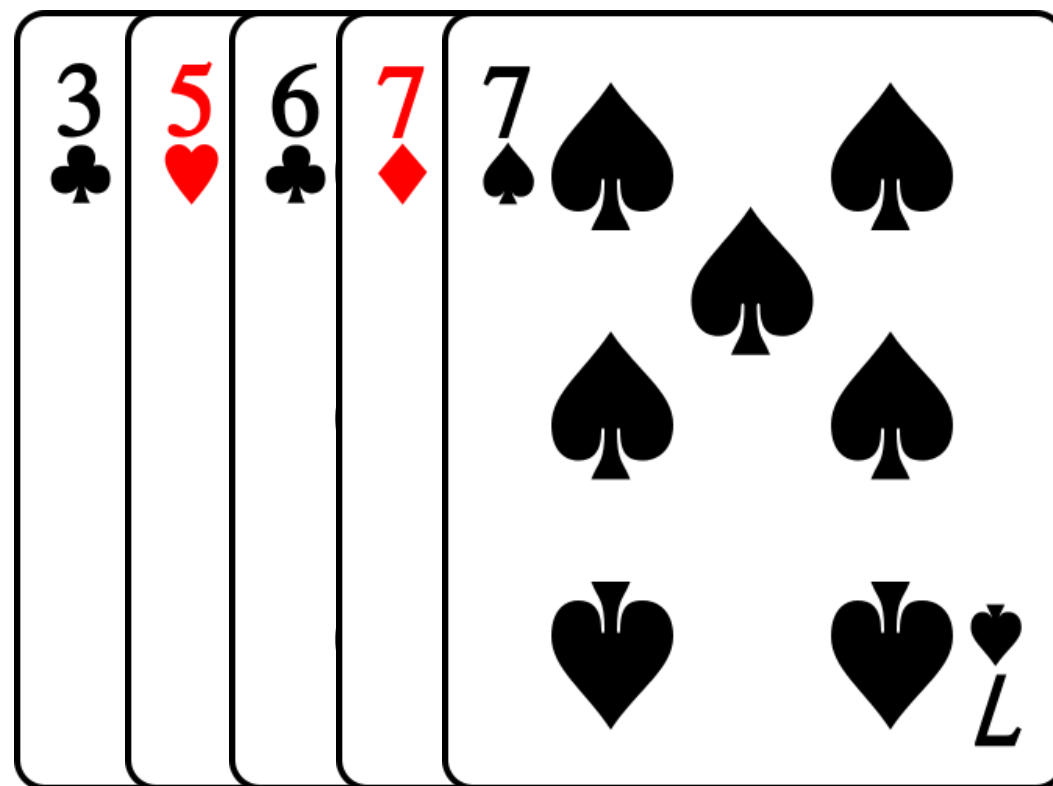
Tri par insertion



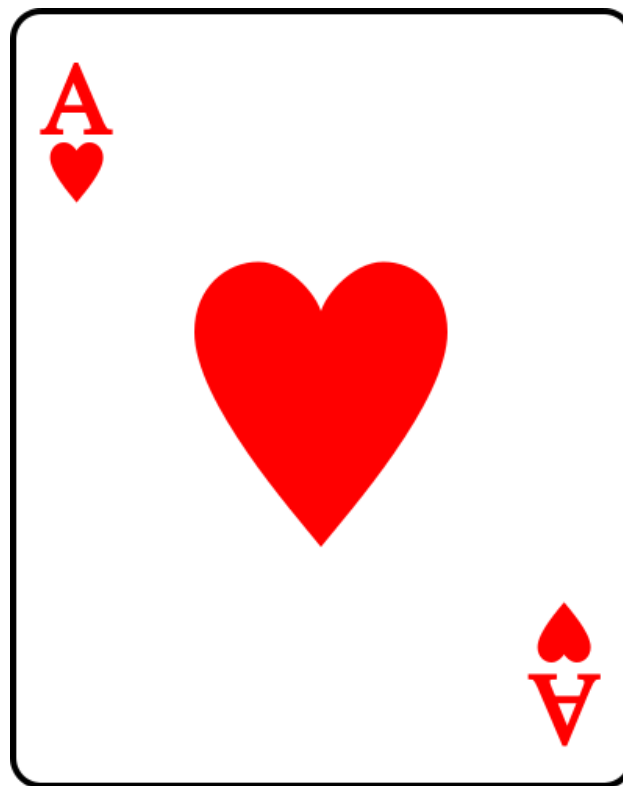
Tri par insertion



Tri par insertion

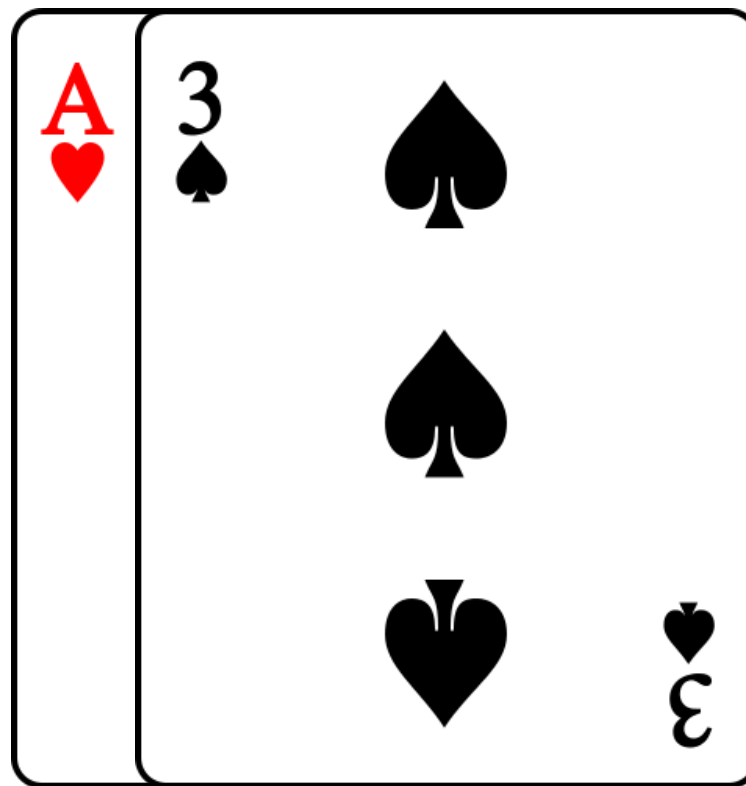


Le meilleur des cas



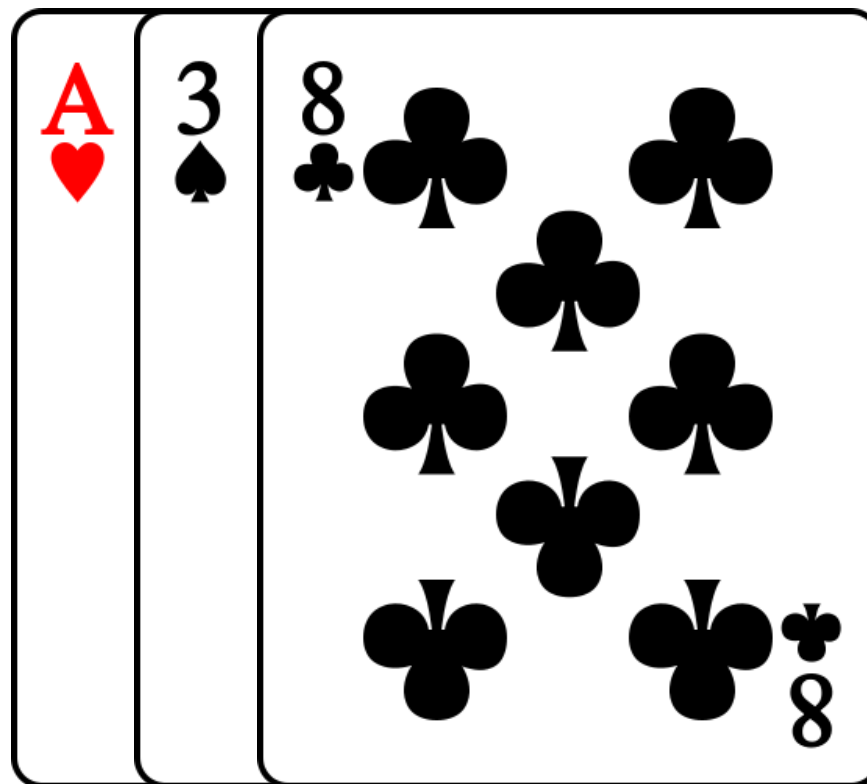
Nº operations = 1

Le meilleur des cas



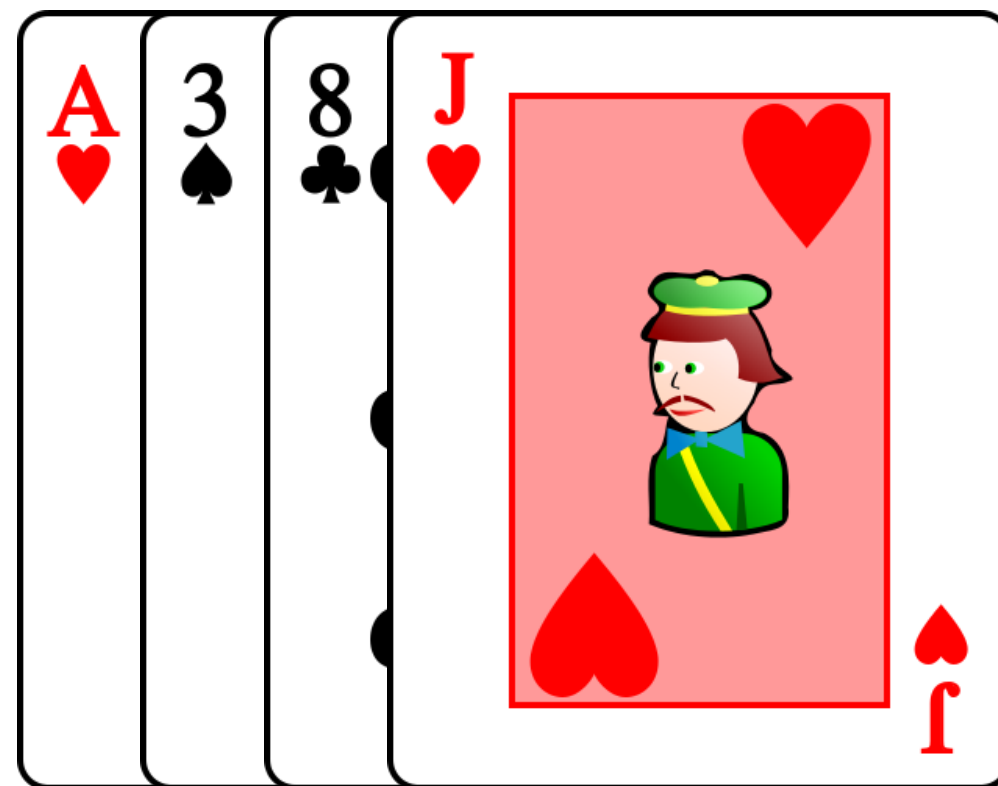
Nº operations = 2

Le meilleur des cas



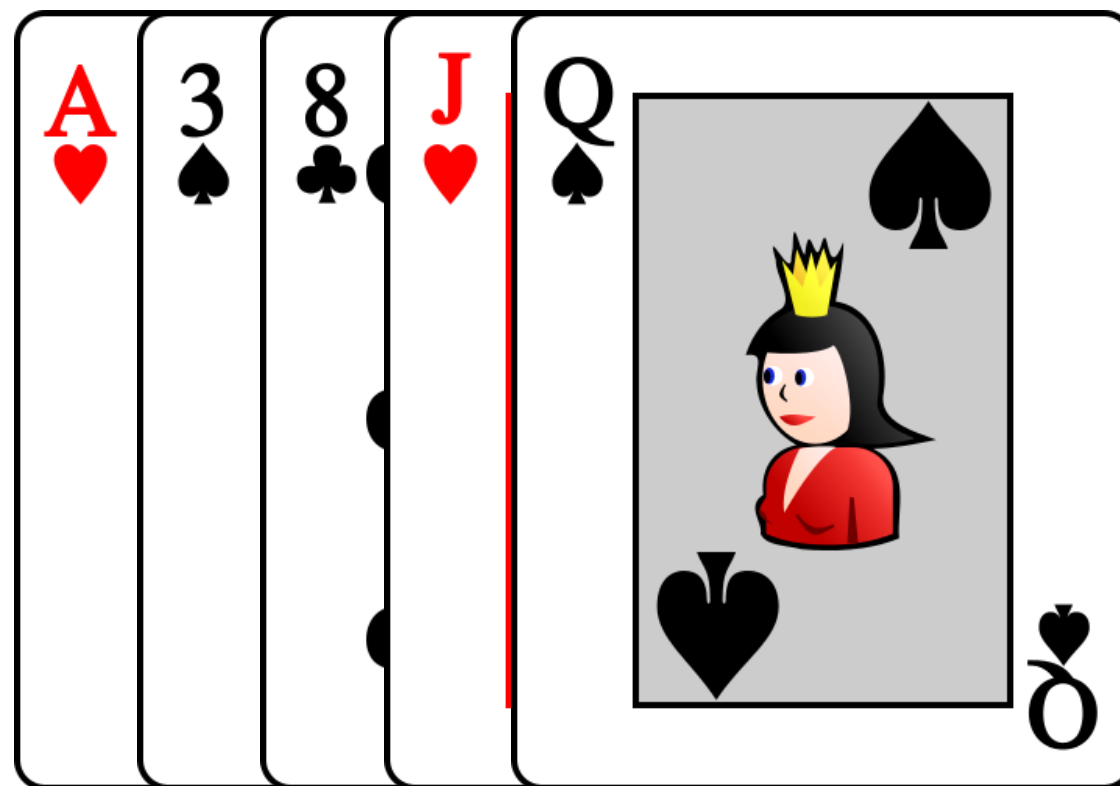
Nº operations = 3

Le meilleur des cas



Nº operations = 4

Le meilleur des cas

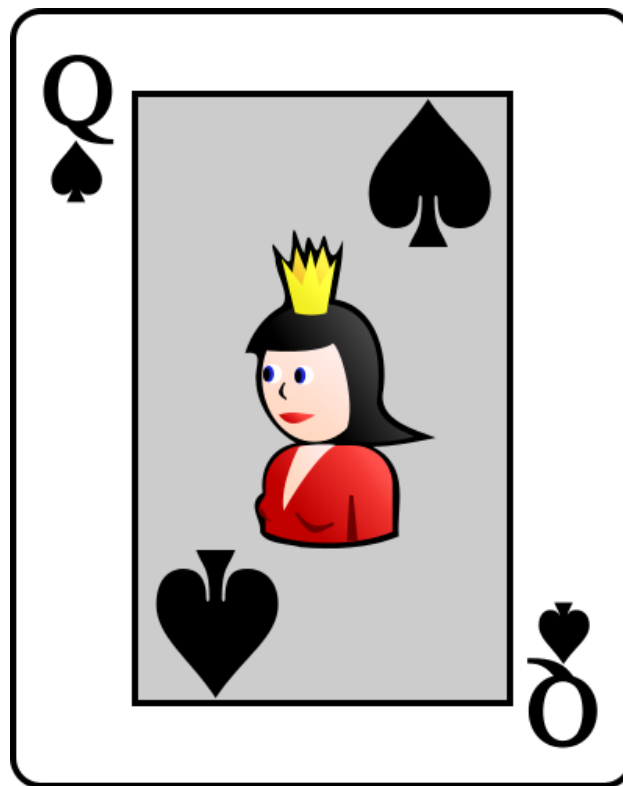


Nº operations = 5

Le meilleur des cas

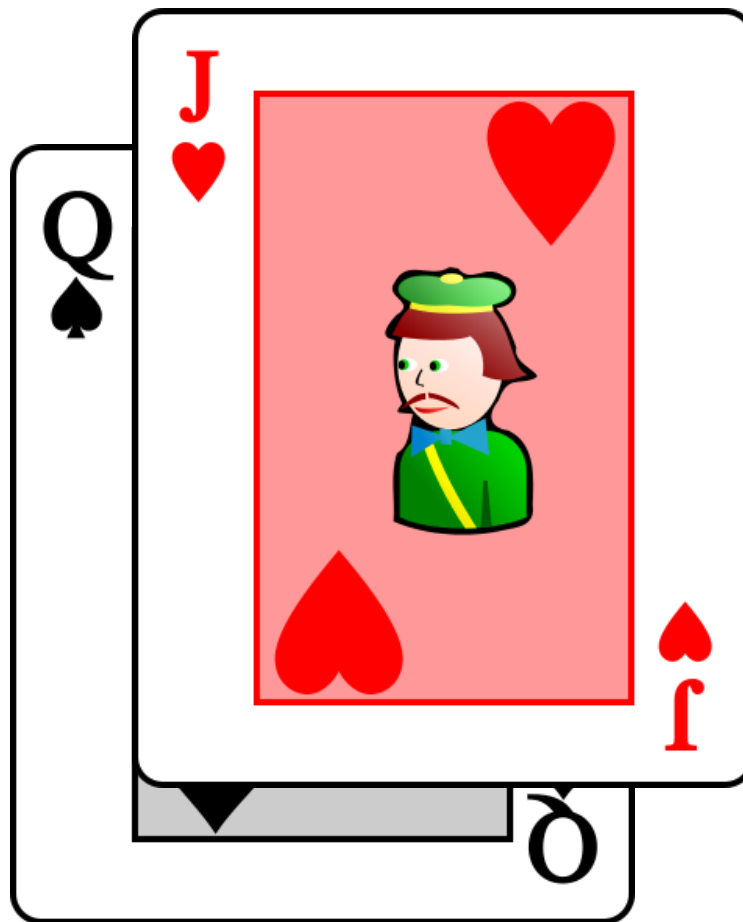
- Les cartes arrivent déjà triées
- On fait n opérations (déplacements de cartes)

Le pire des cas



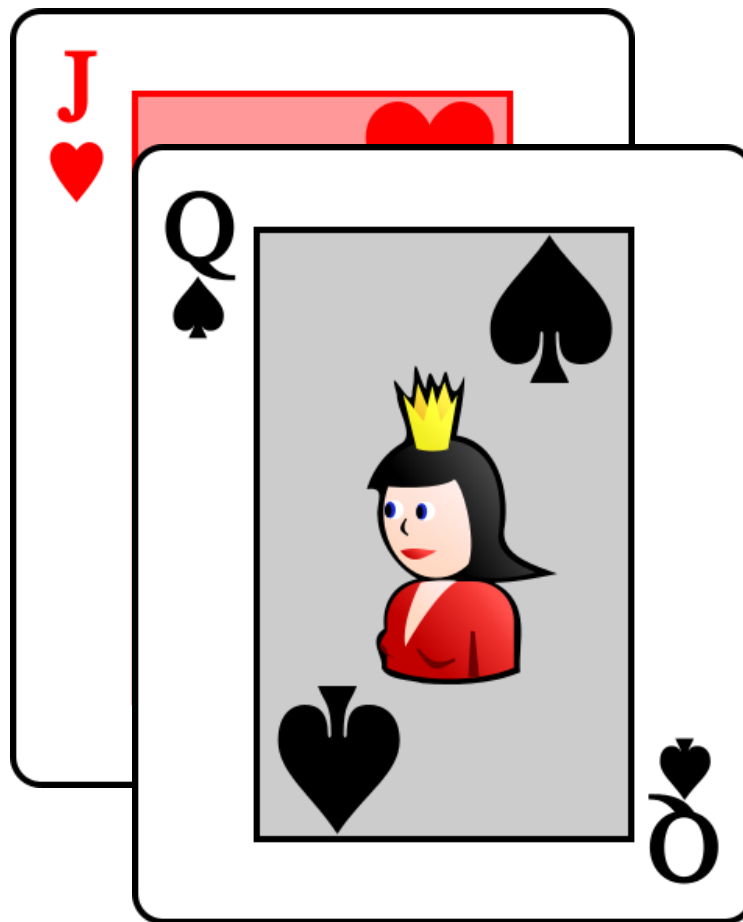
Nº operations = 1

Le pire des cas



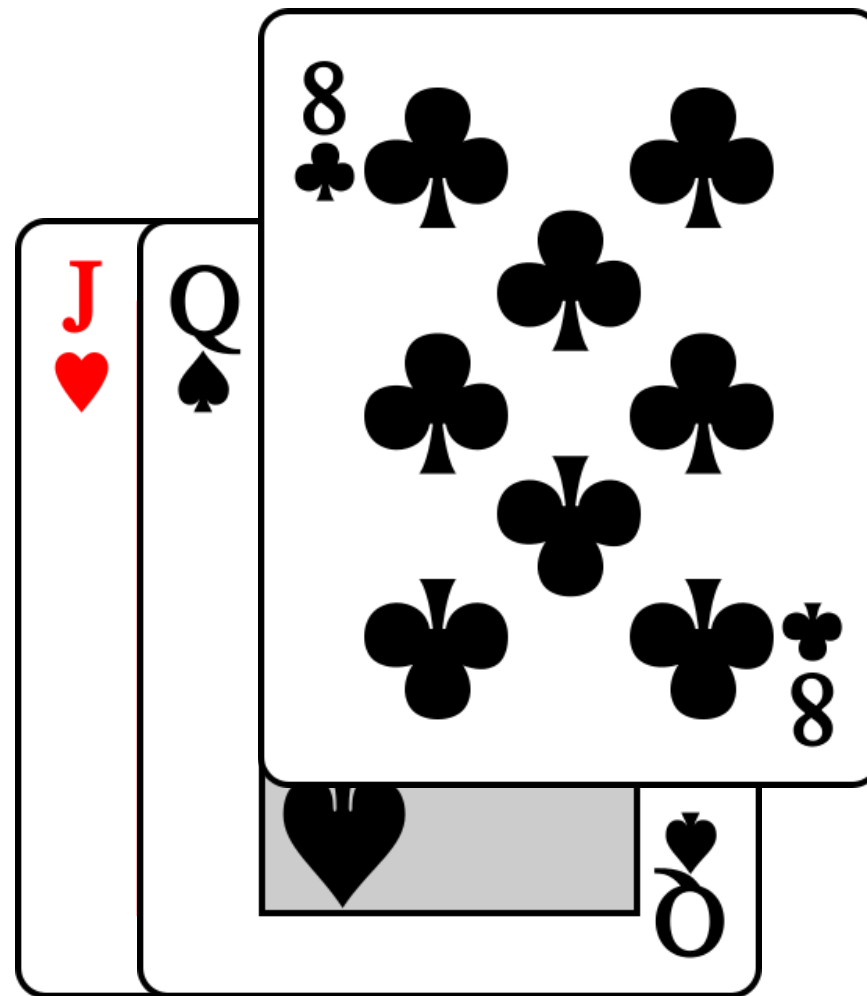
Nº operations = 1 + 1

Le pire des cas



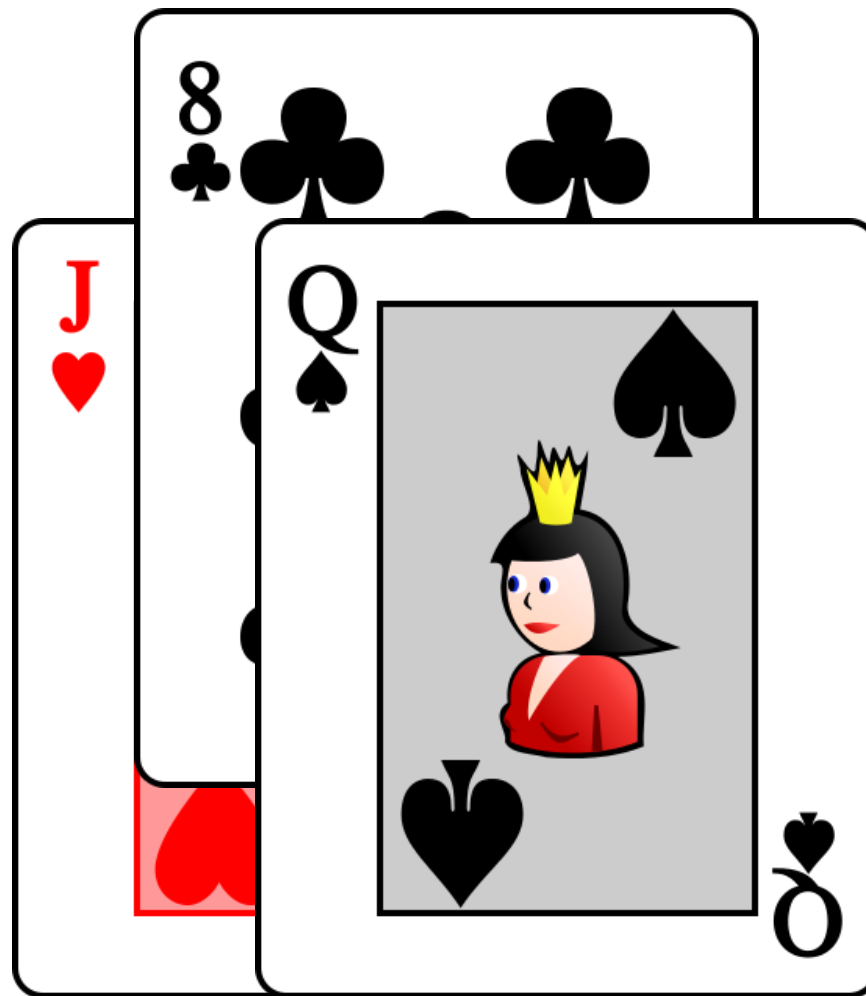
Nº operations = 1 + 2

Le pire des cas



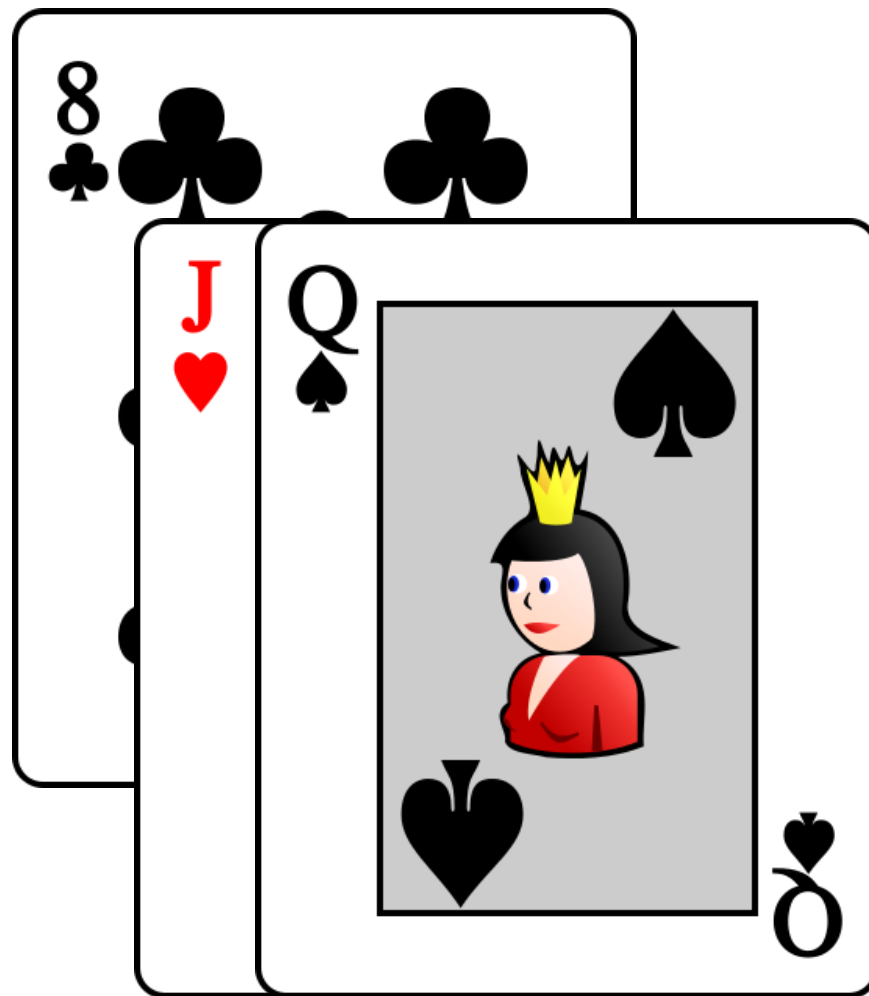
Nº operations = 1 + 2 + 1

Le pire des cas



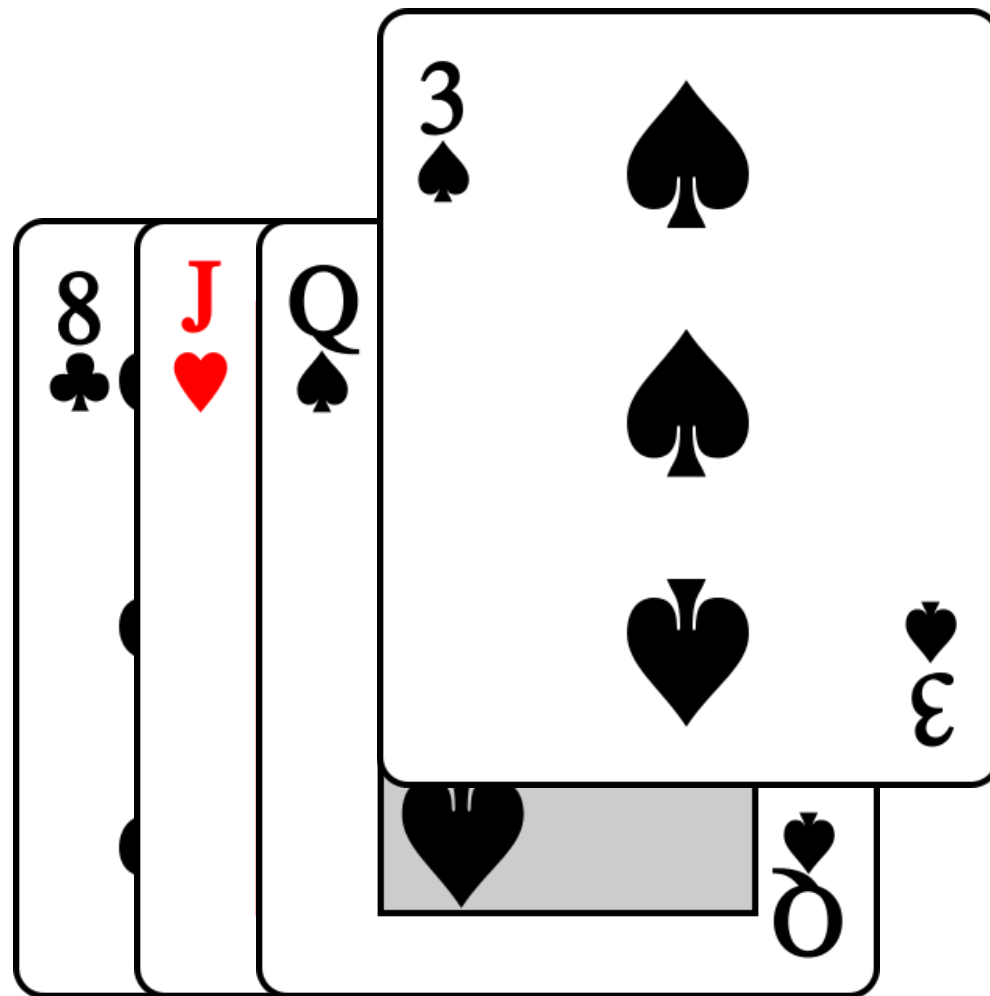
Nº operations = 1 + 2 + 2

Le pire des cas



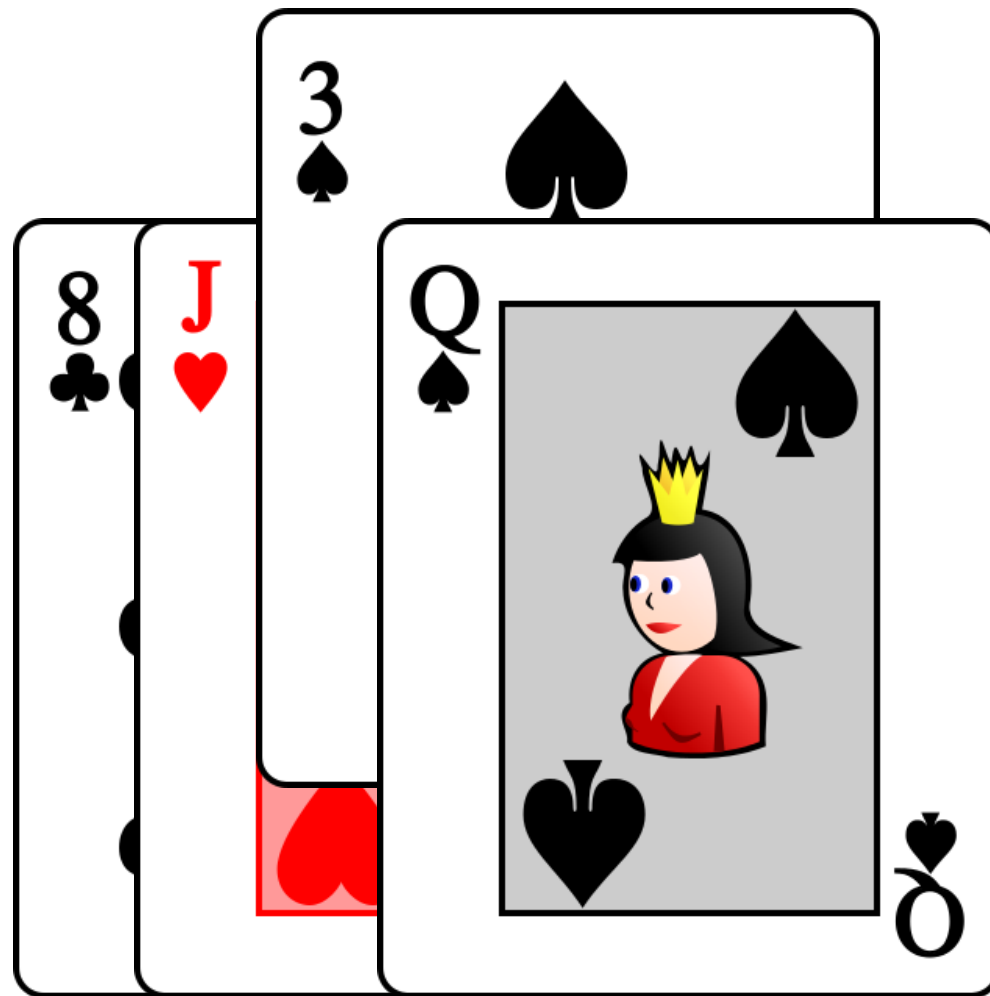
Nº operations = 1 + 2 + 3

Le pire des cas



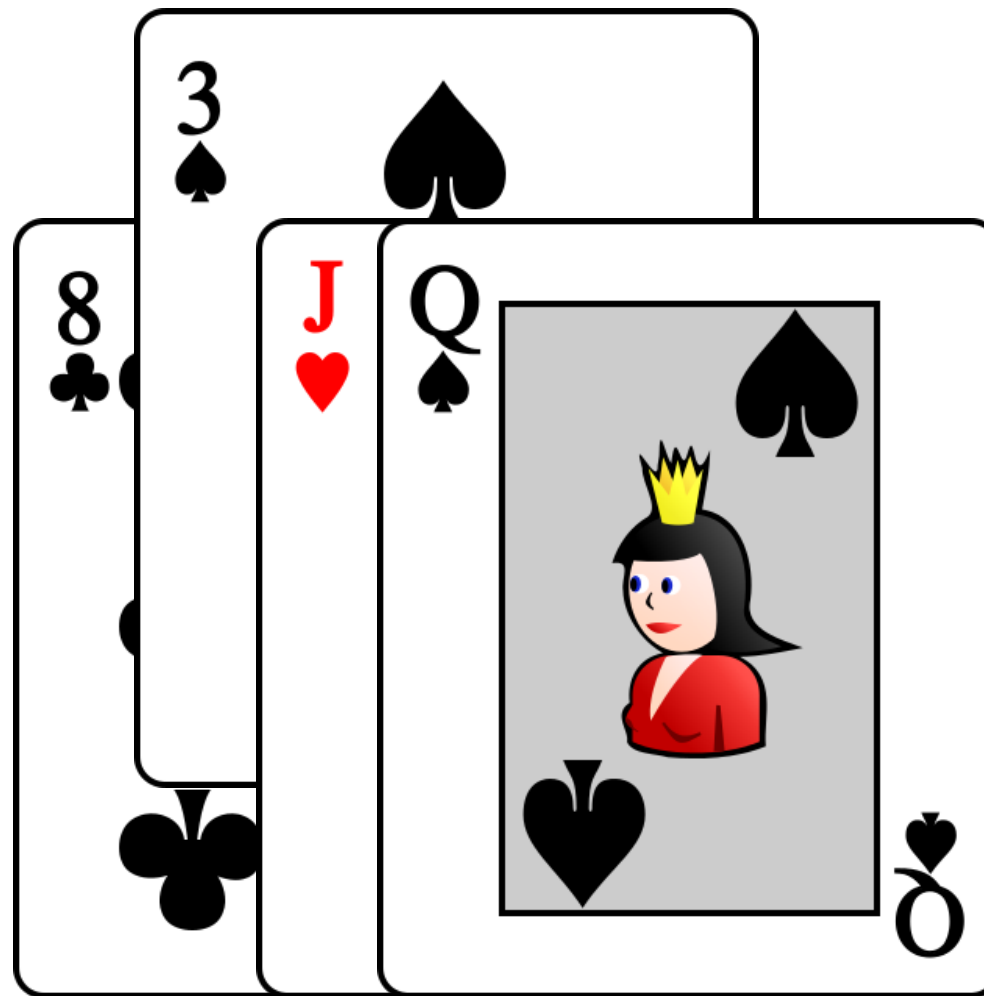
Nº operations = 1 + 2 + 3 + 1

Le pire des cas



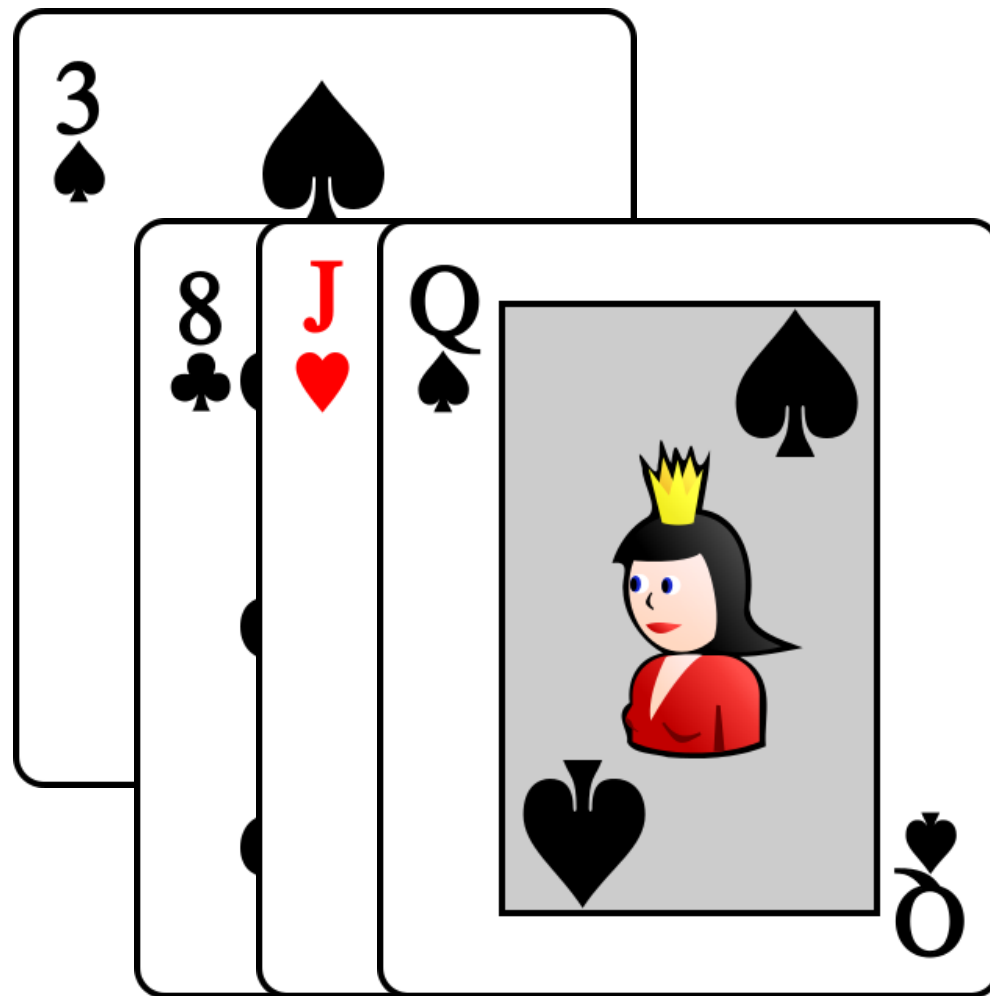
Nº operations = 1 + 2 + 3 + 2

Le pire des cas



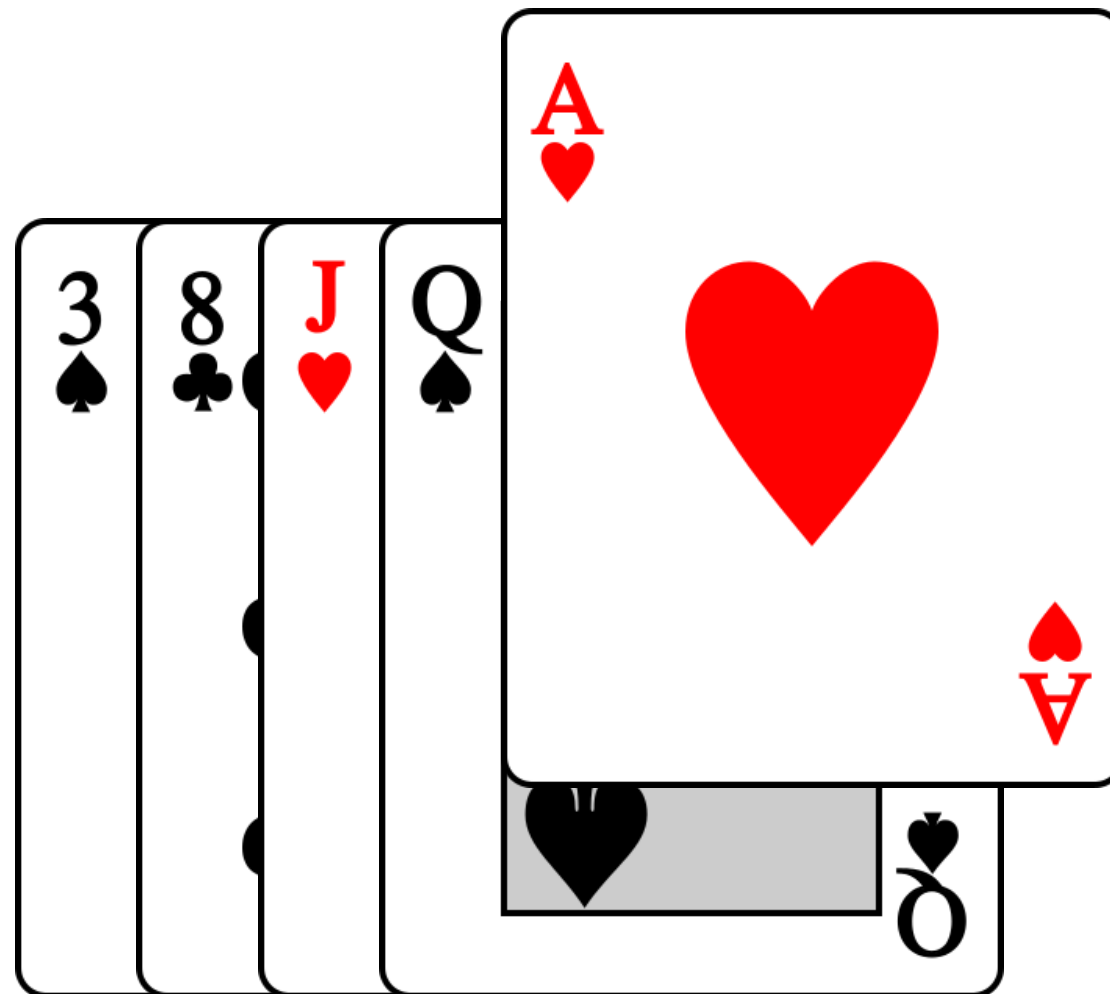
Nº operations = 1 + 2 + 3 + 3

Le pire des cas



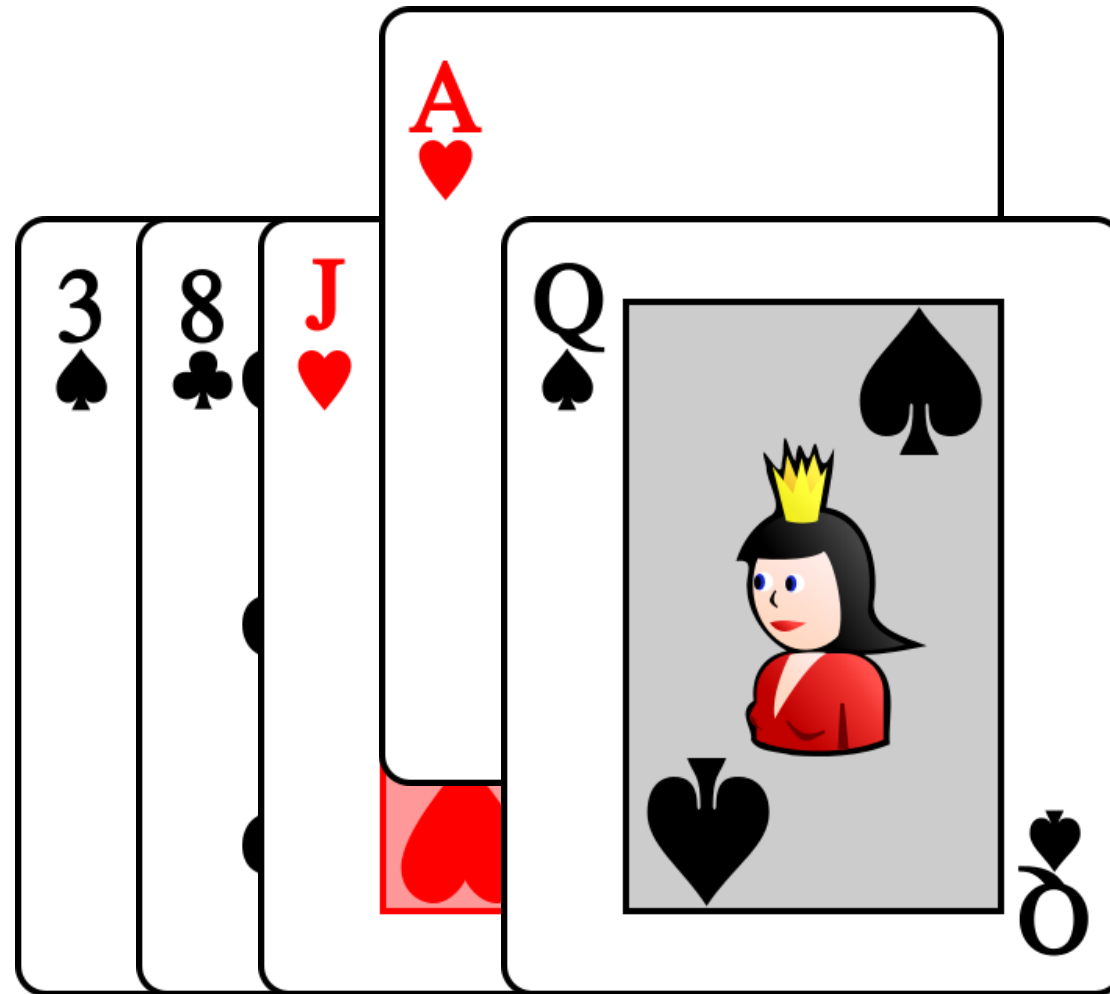
Nº operations = 1 + 2 + 3 + 4

Le pire des cas



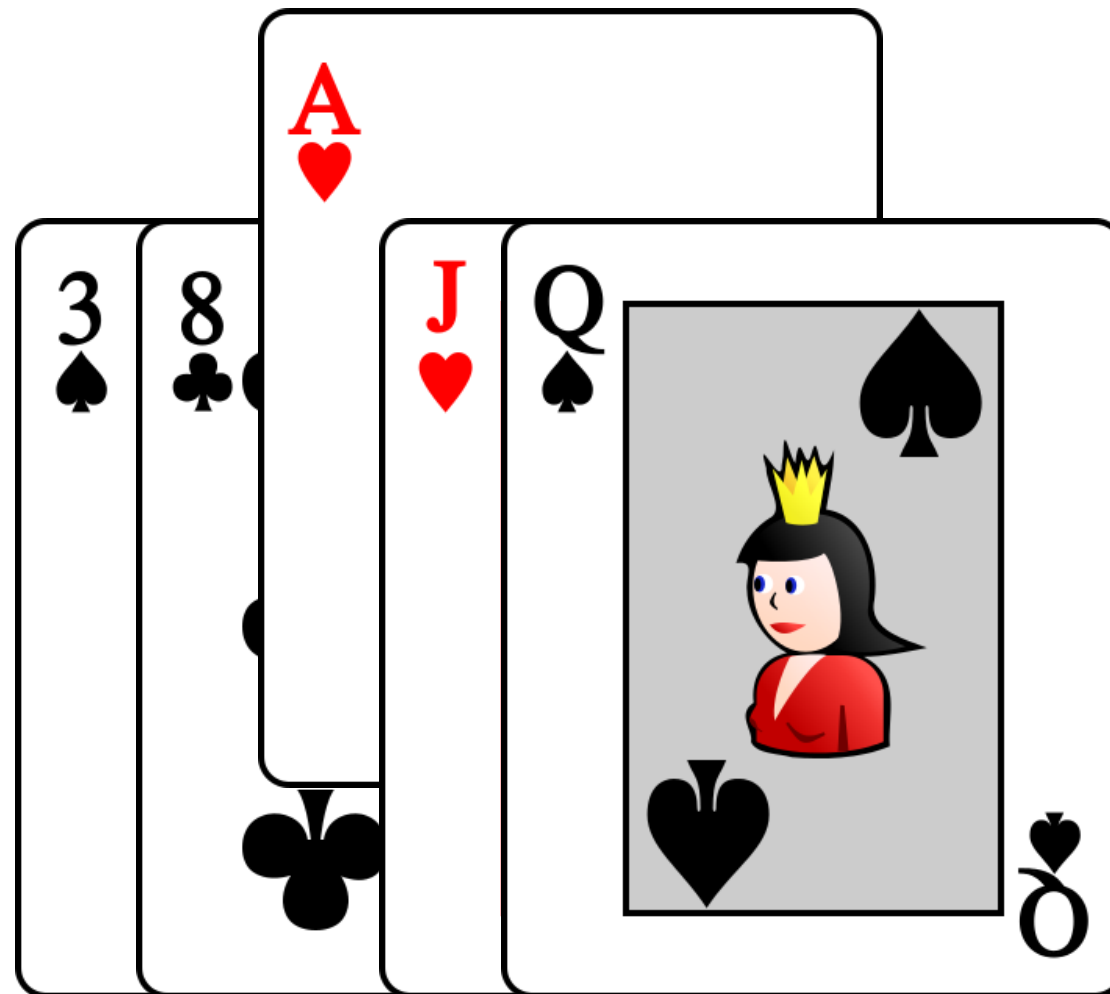
Nº operations = 1 + 2 + 3 + 4 + 1

Le pire des cas



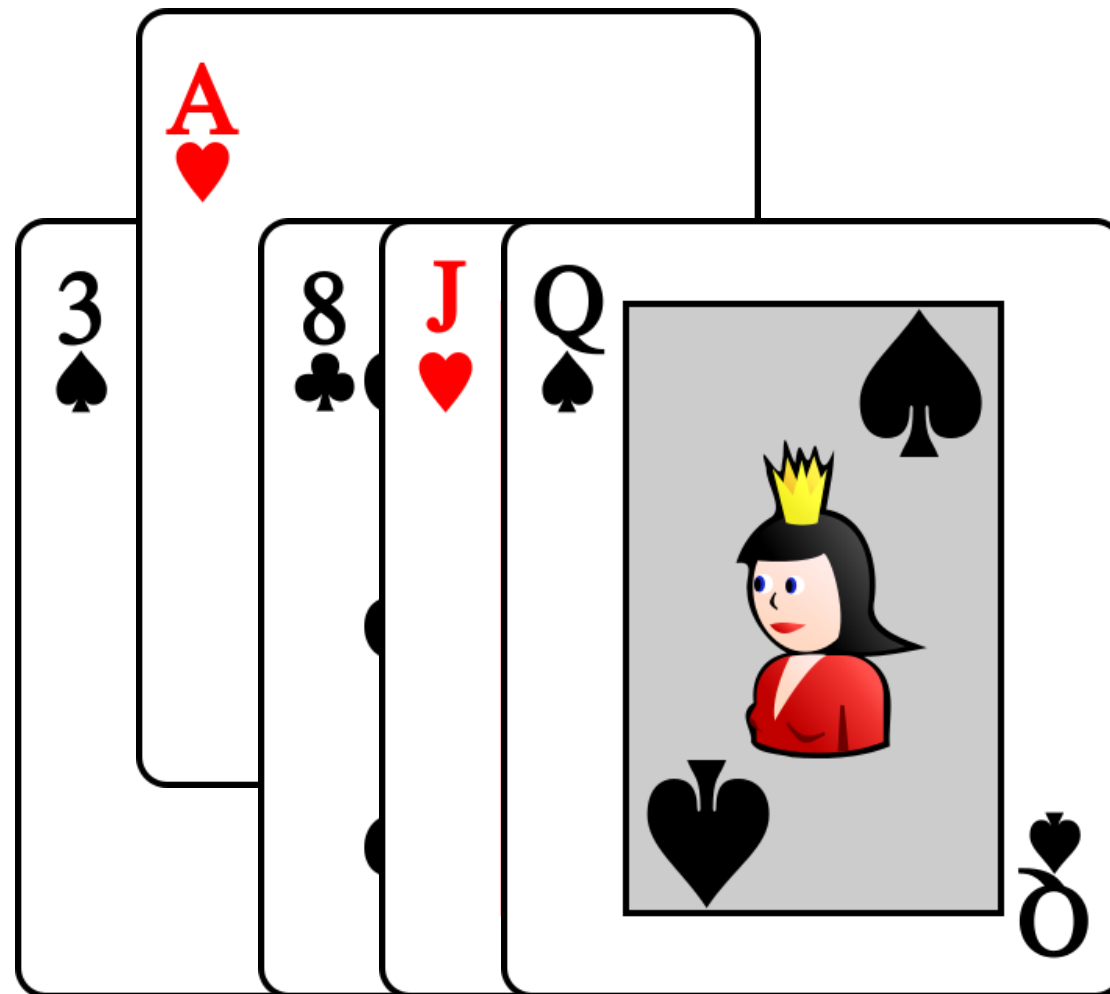
Nº operations = $1 + 2 + 3 + 4 + 2$

Le pire des cas



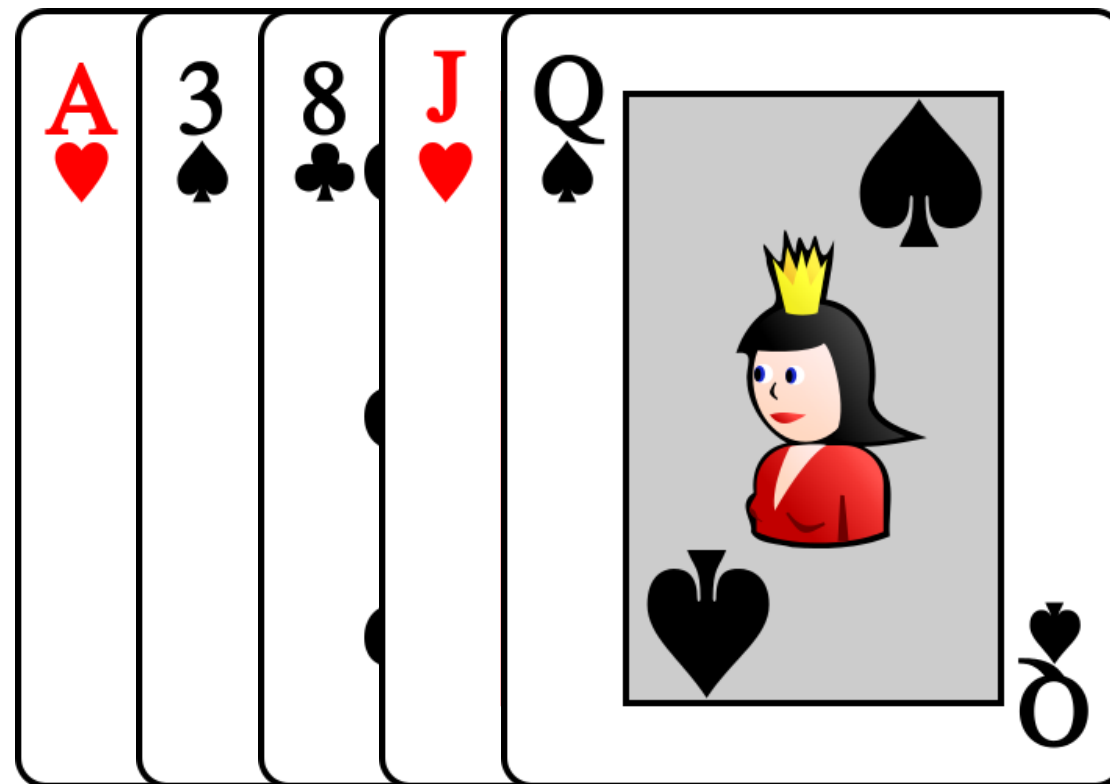
Nº operations = 1 + 2 + 3 + 4 + 3

Le pire des cas



Nº operations = 1 + 2 + 3 + 4 + 4

Le pire des cas



Nº operations = 1 + 2 + 3 + 4 + 5

Le pire des cas

- Les cartes arrivent en ordre décroissant
- On fait i opérations pour la i -ème carte
- Le nombre totale est $1 + 2 + 3 + \cdots + n$

$$\sum_{i=1}^n i = \frac{1}{2}n(n+1) = \frac{1}{2}(n^2 + n) \in O(n^2)$$

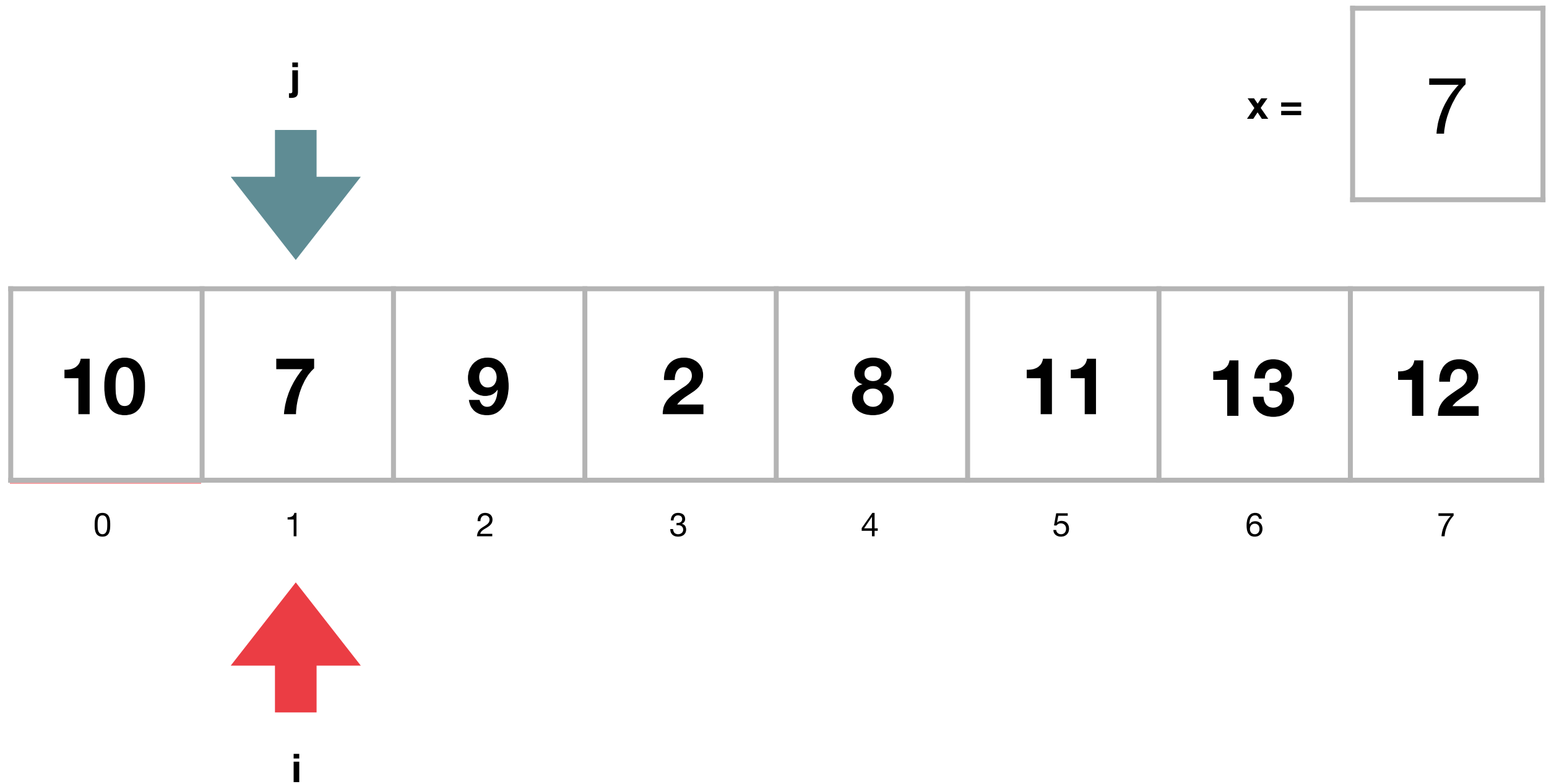
Tri d'un tableau par insertion

```
procedure trier-par-insertion(T)
  n := longueur(T)
  pour i := 1 à n - 1 faire (on saute le 0)
    x := T[i]
    j := i
    tant que j > 0 et x < T[j - 1] faire
      (décaler d'un élément)
      T[j] := T[j - 1]
      j := j - 1
    fin tant que
    (ici x ≥ T[j - 1] ou bien j = 0)
    T[j] := x
  fin pour
fin procedure
```

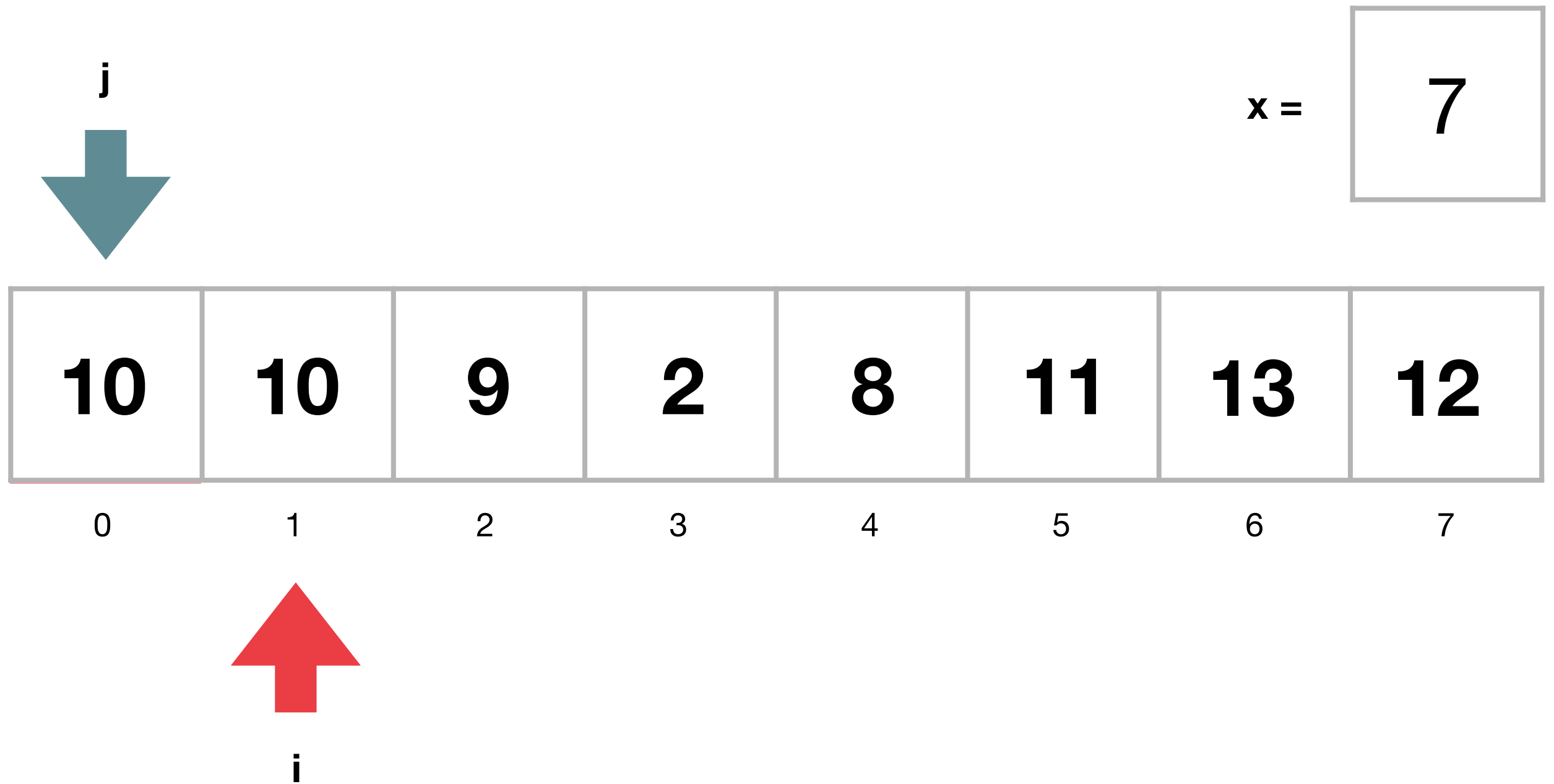
Tri par insertion

10	7	9	2	8	11	13	12
0	1	2	3	4	5	6	7

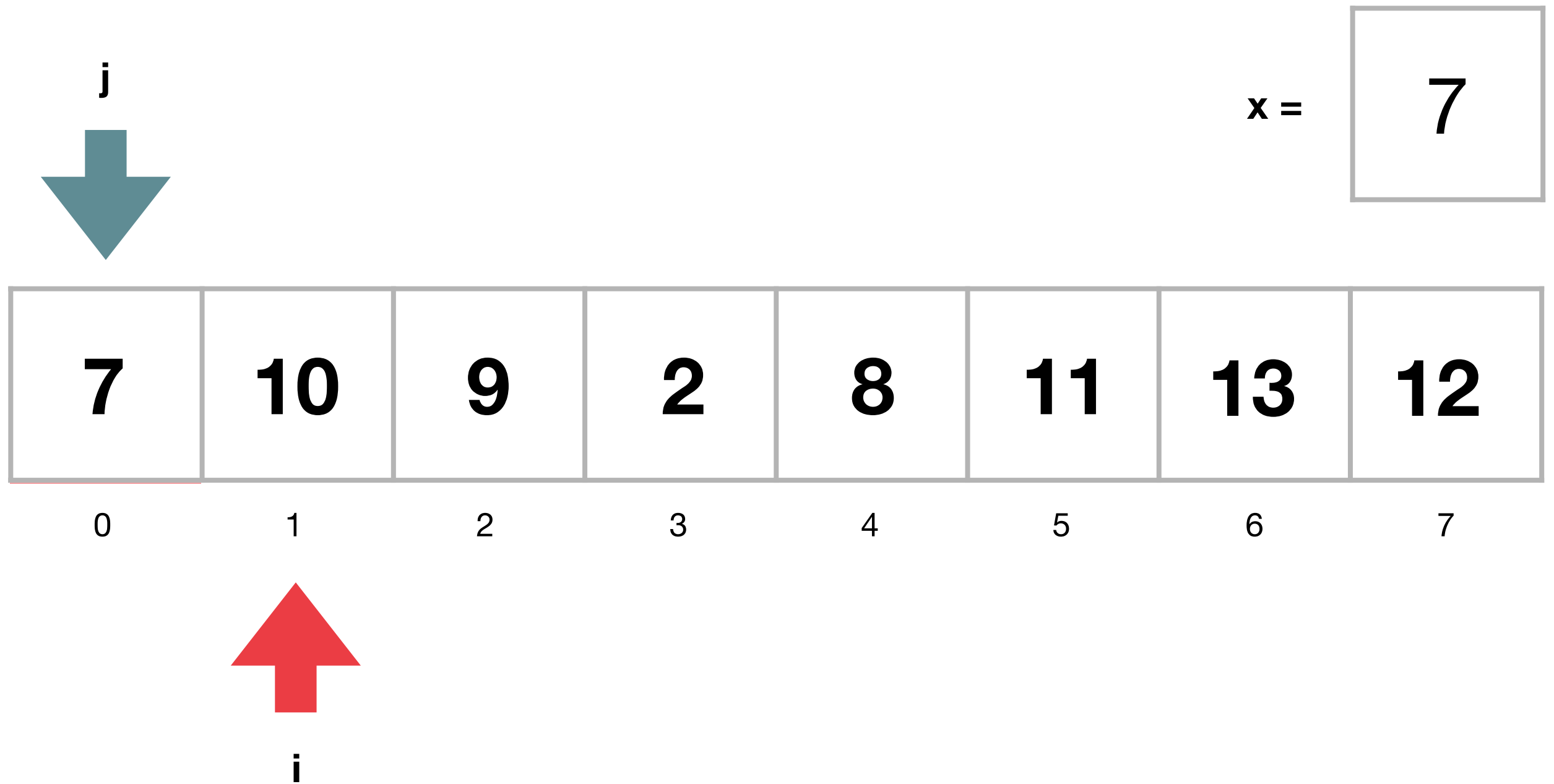
Tri par insertion



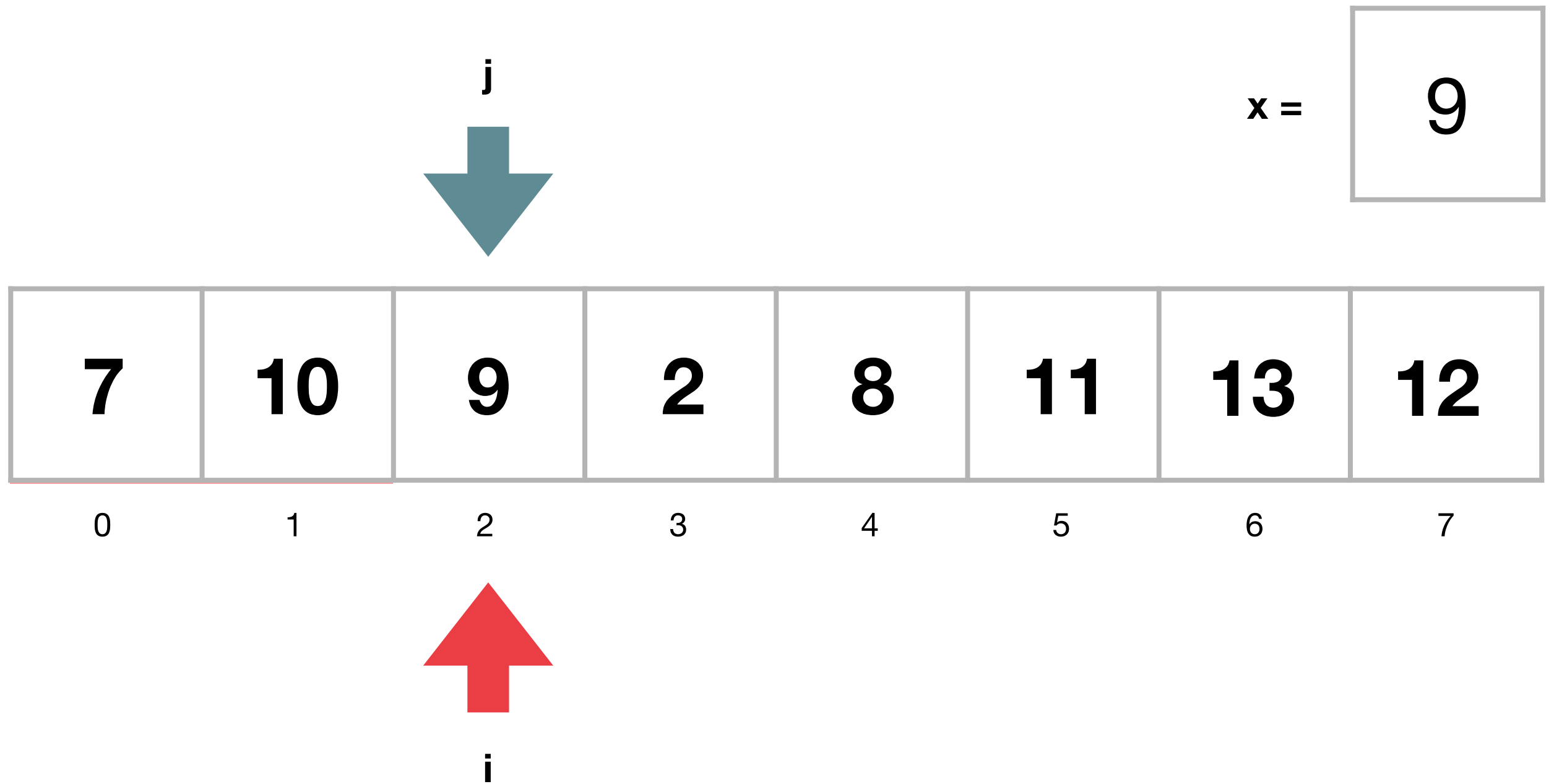
Tri par insertion



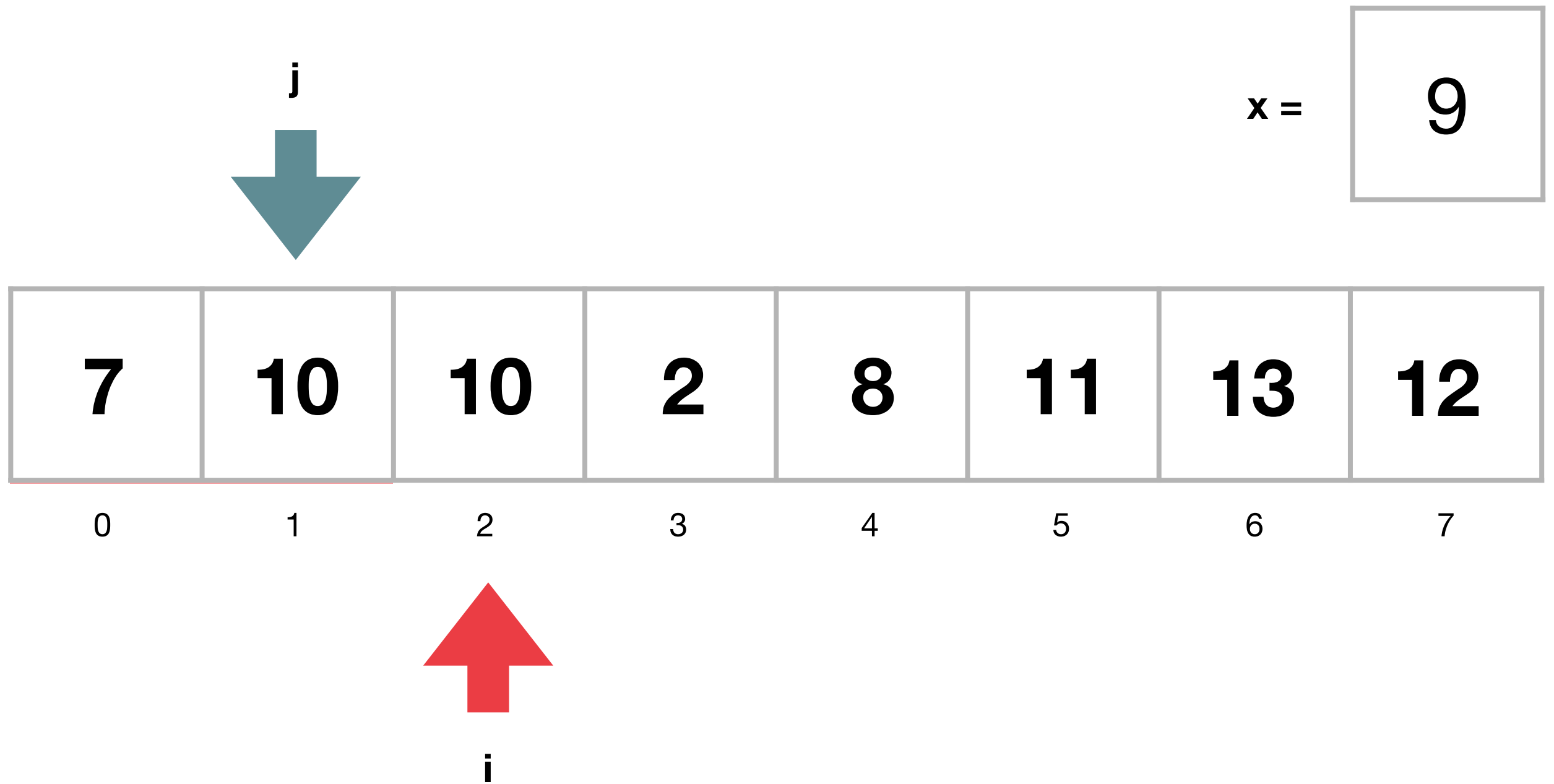
Tri par insertion



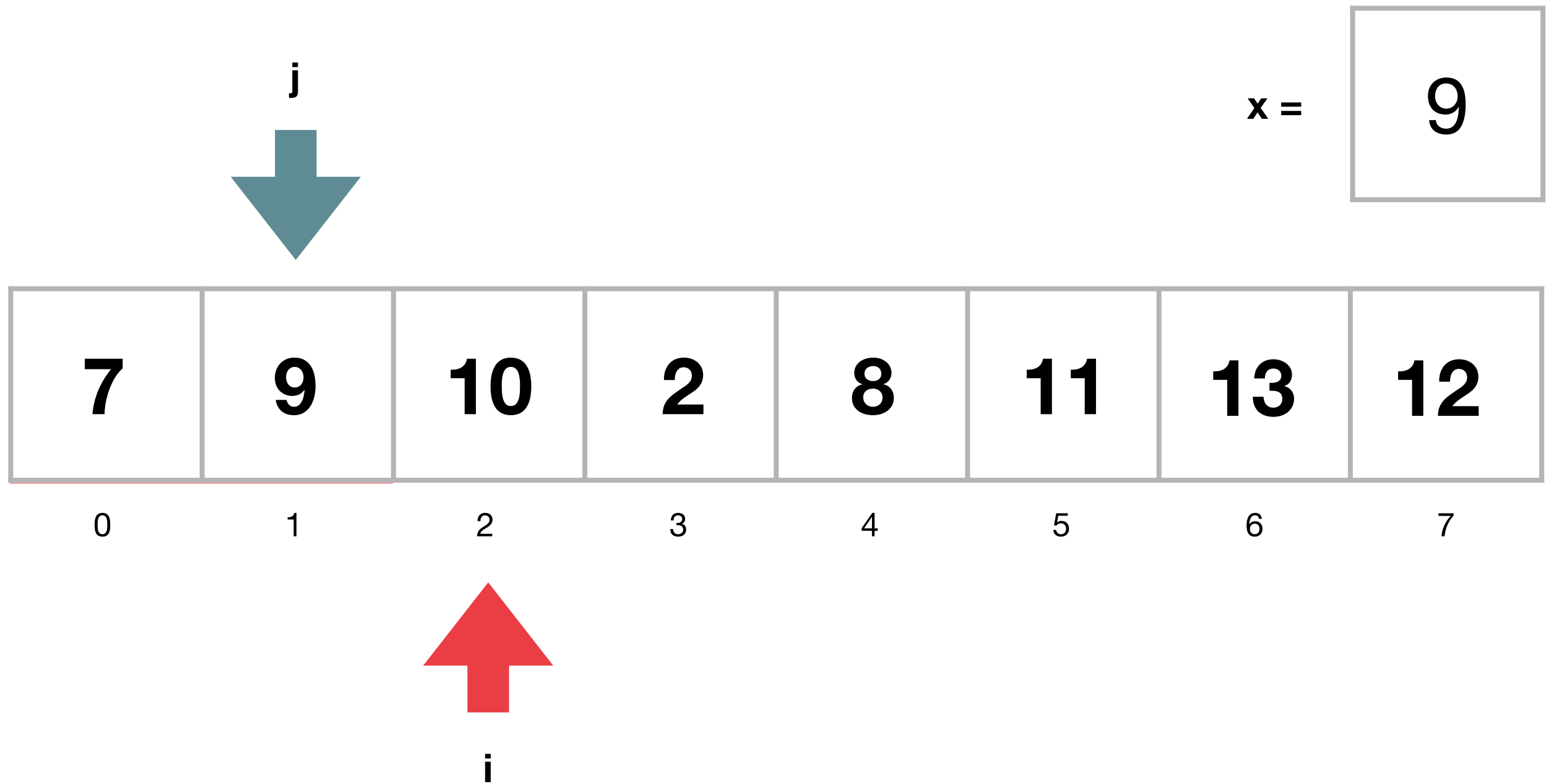
Tri par insertion



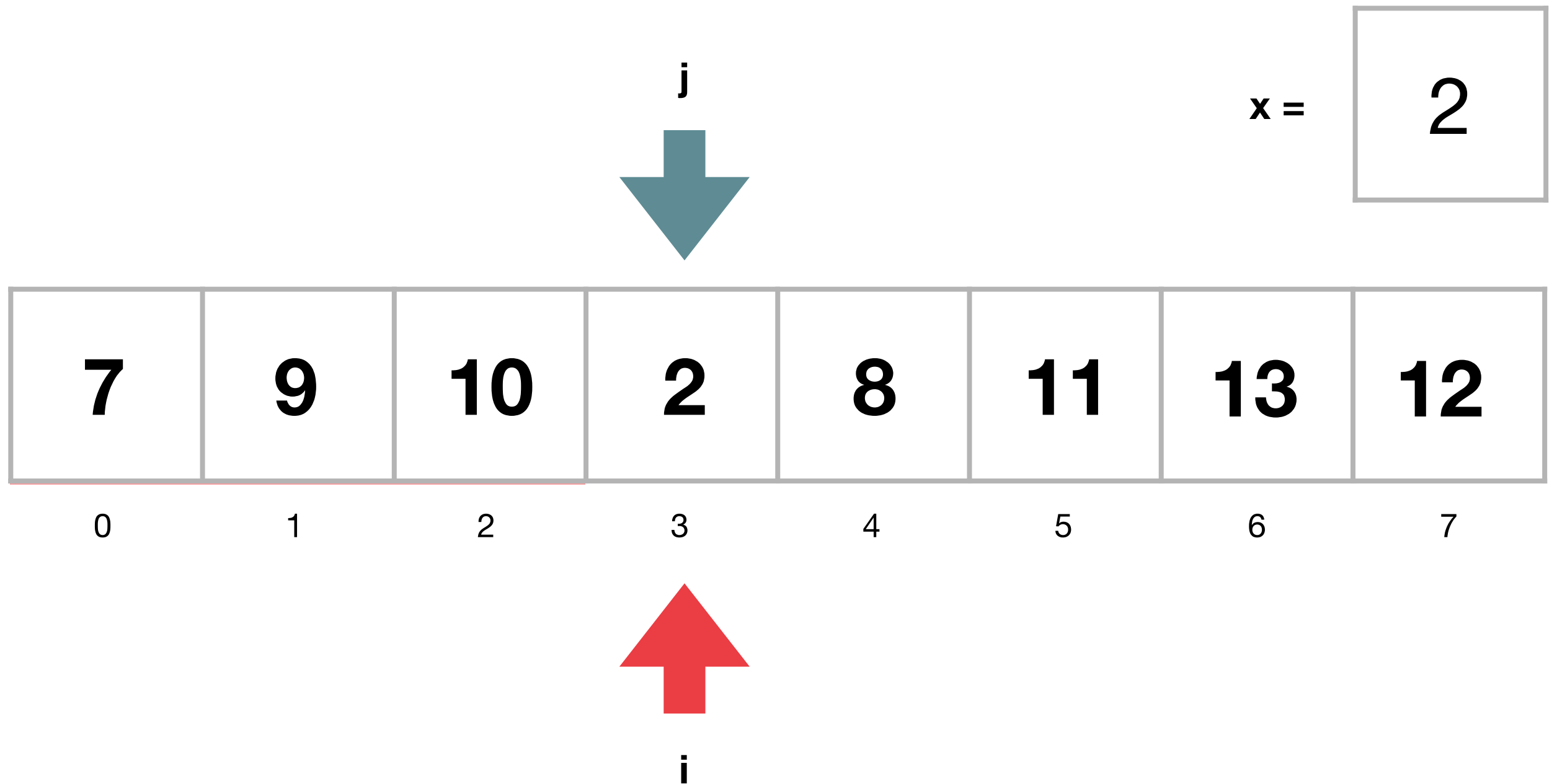
Tri par insertion



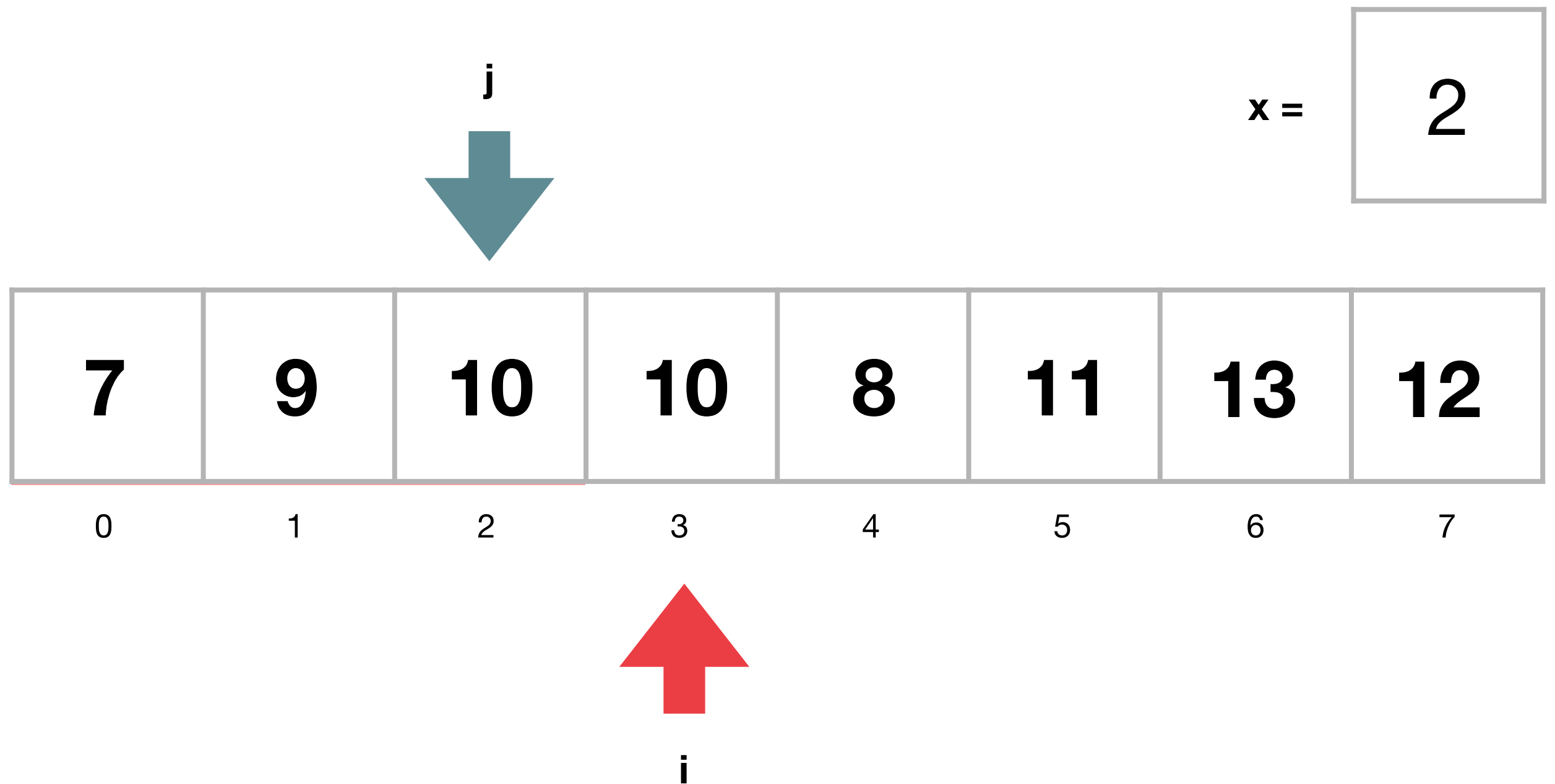
Tri par insertion



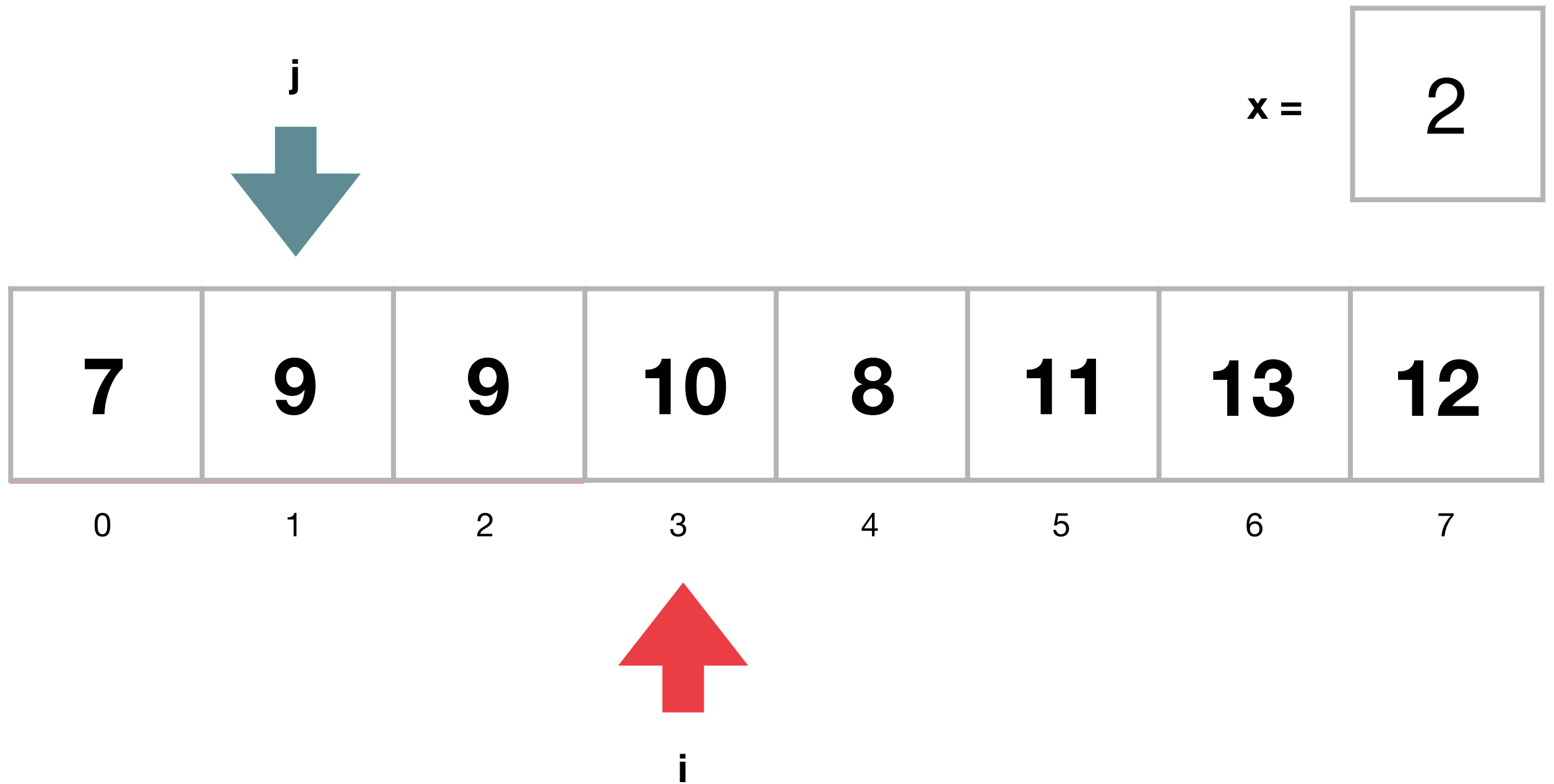
Tri par insertion



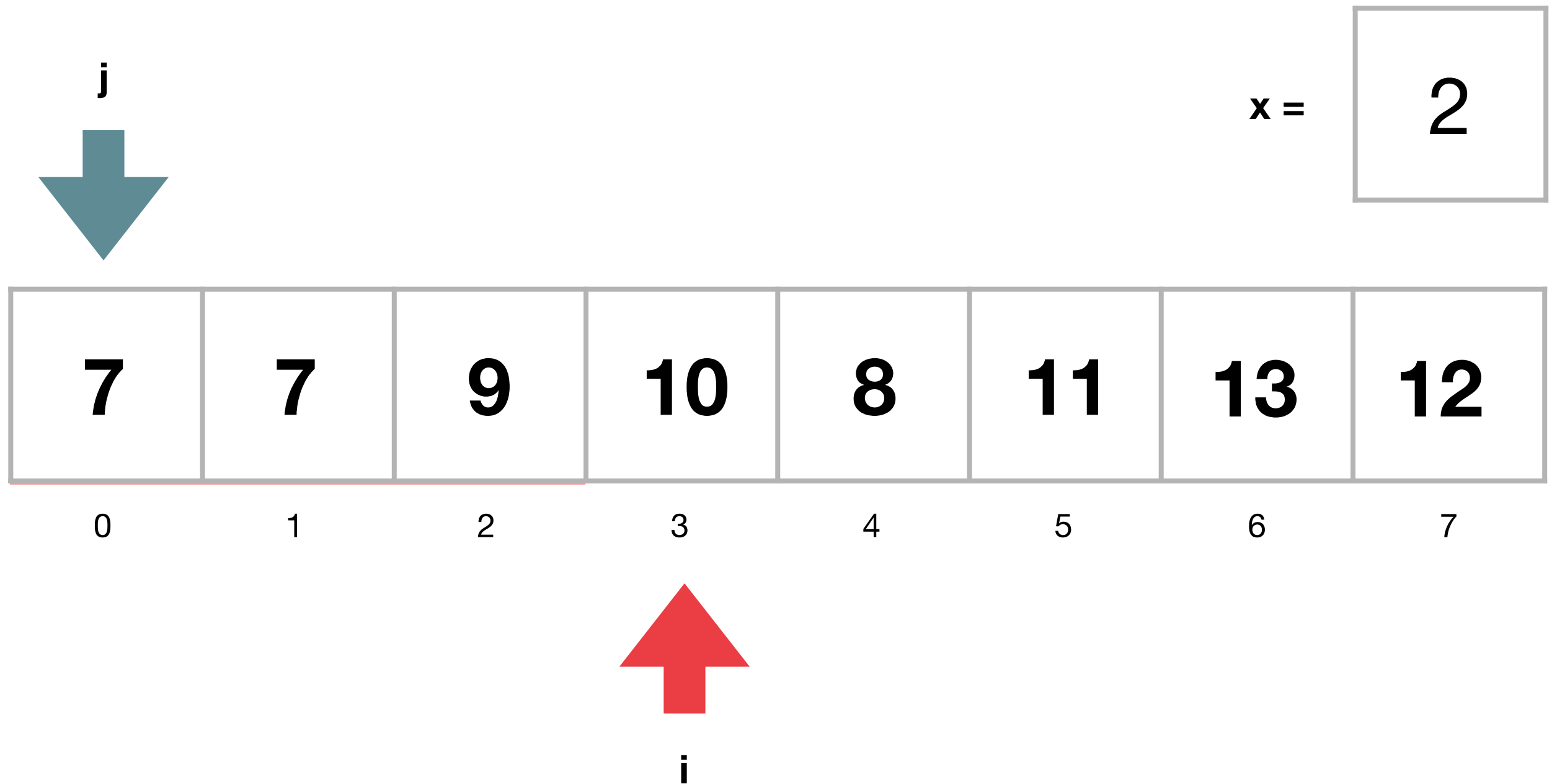
Tri par insertion



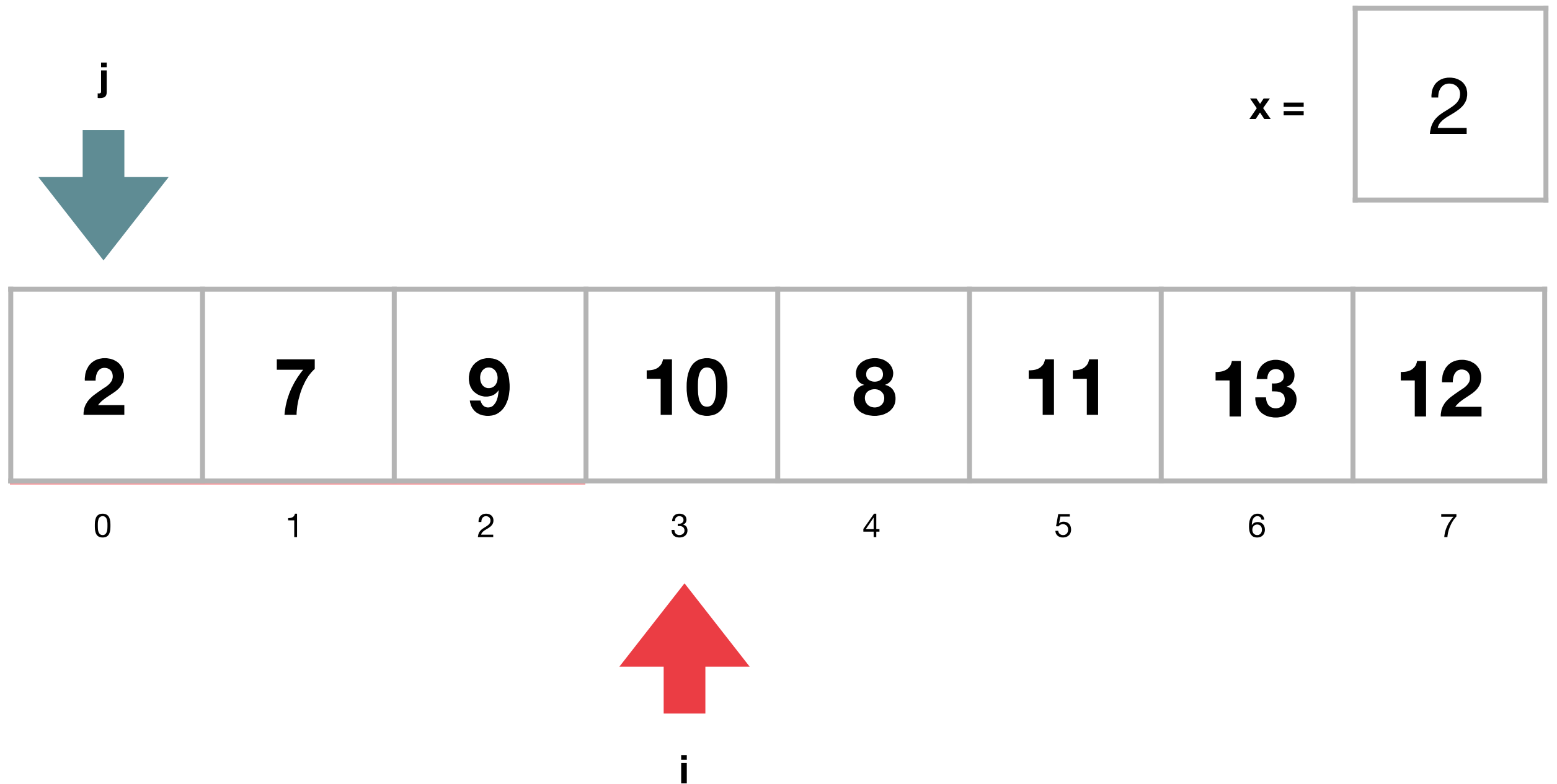
Tri par insertion



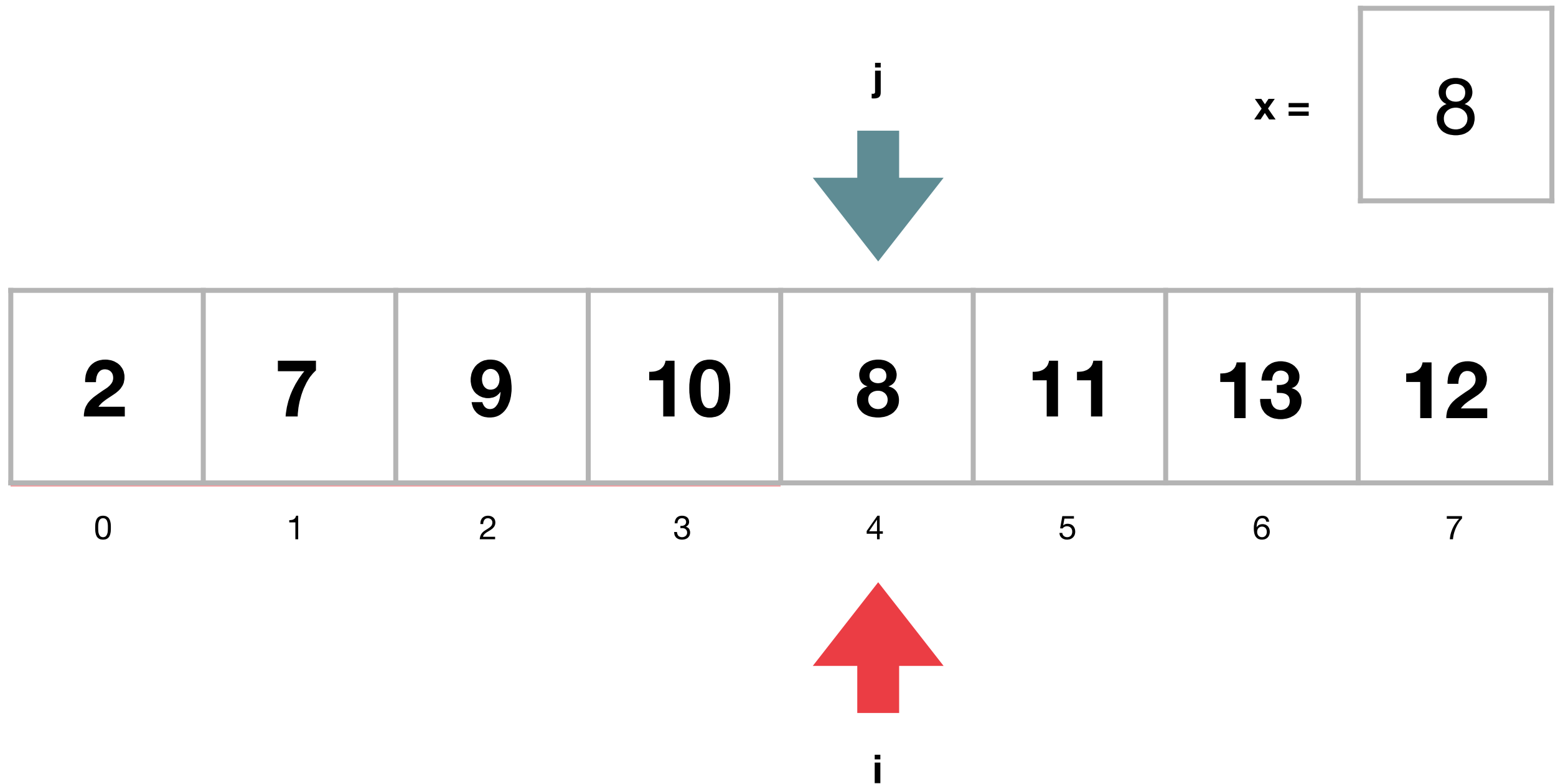
Tri par insertion



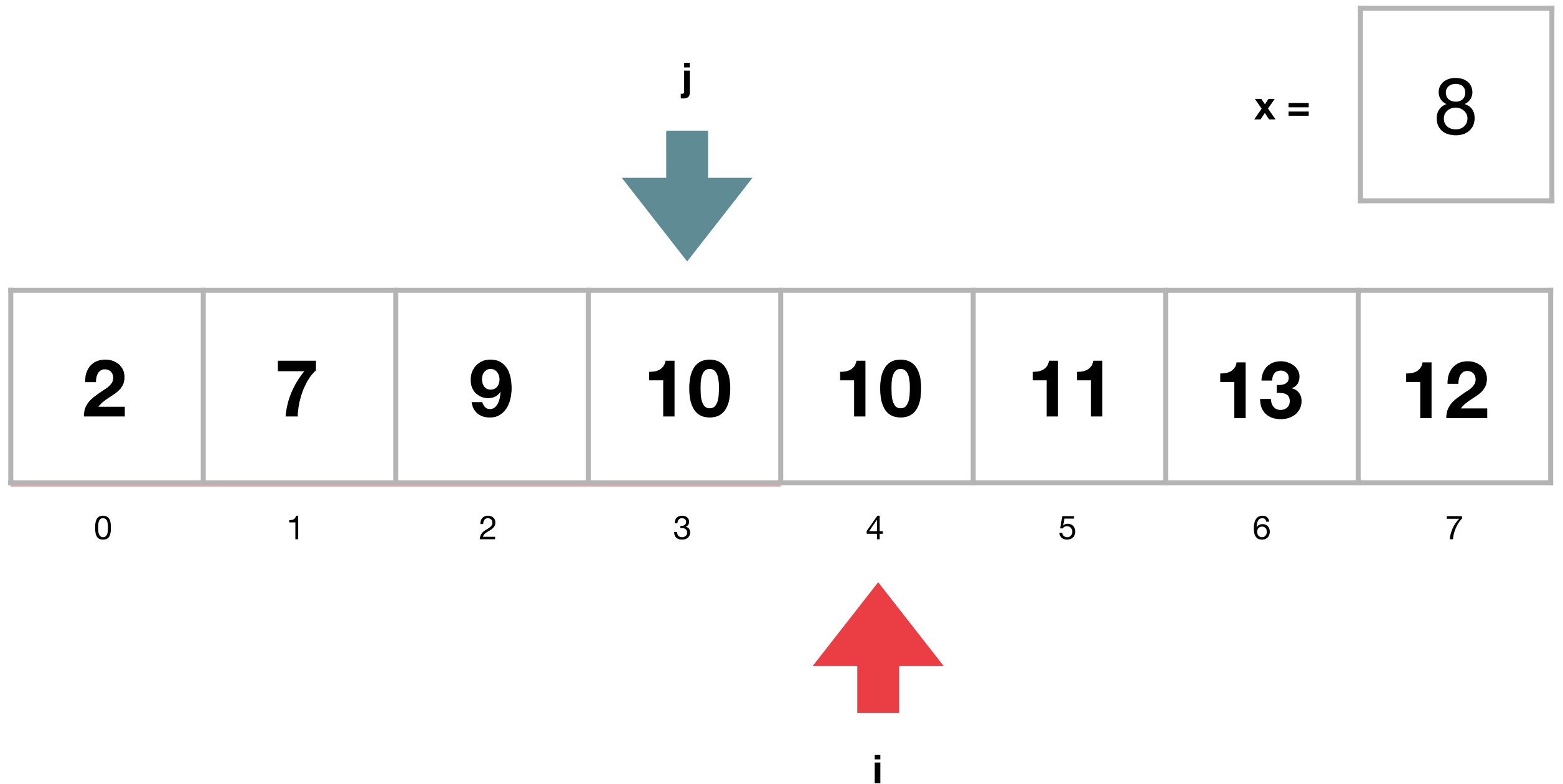
Tri par insertion



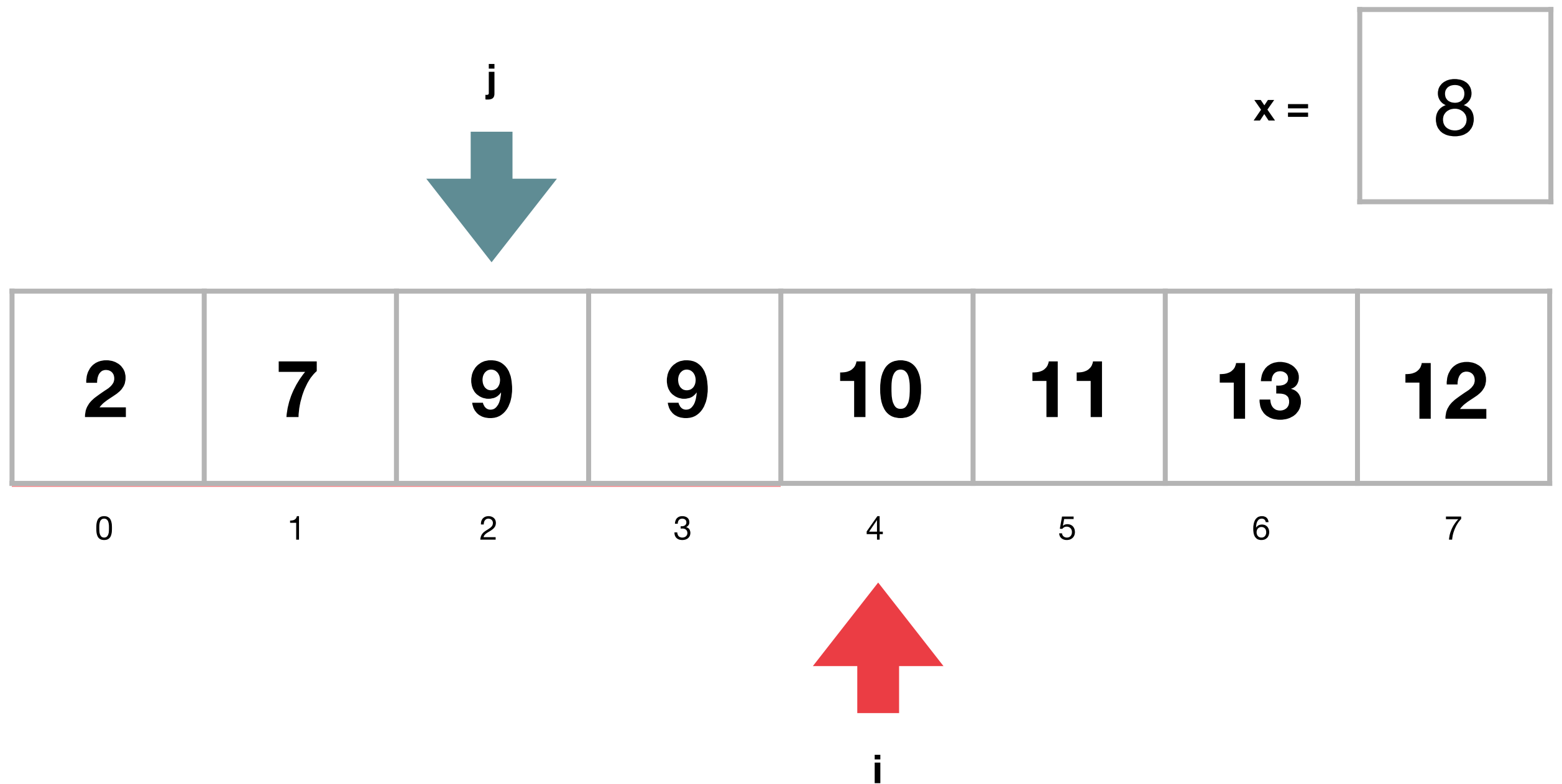
Tri par insertion



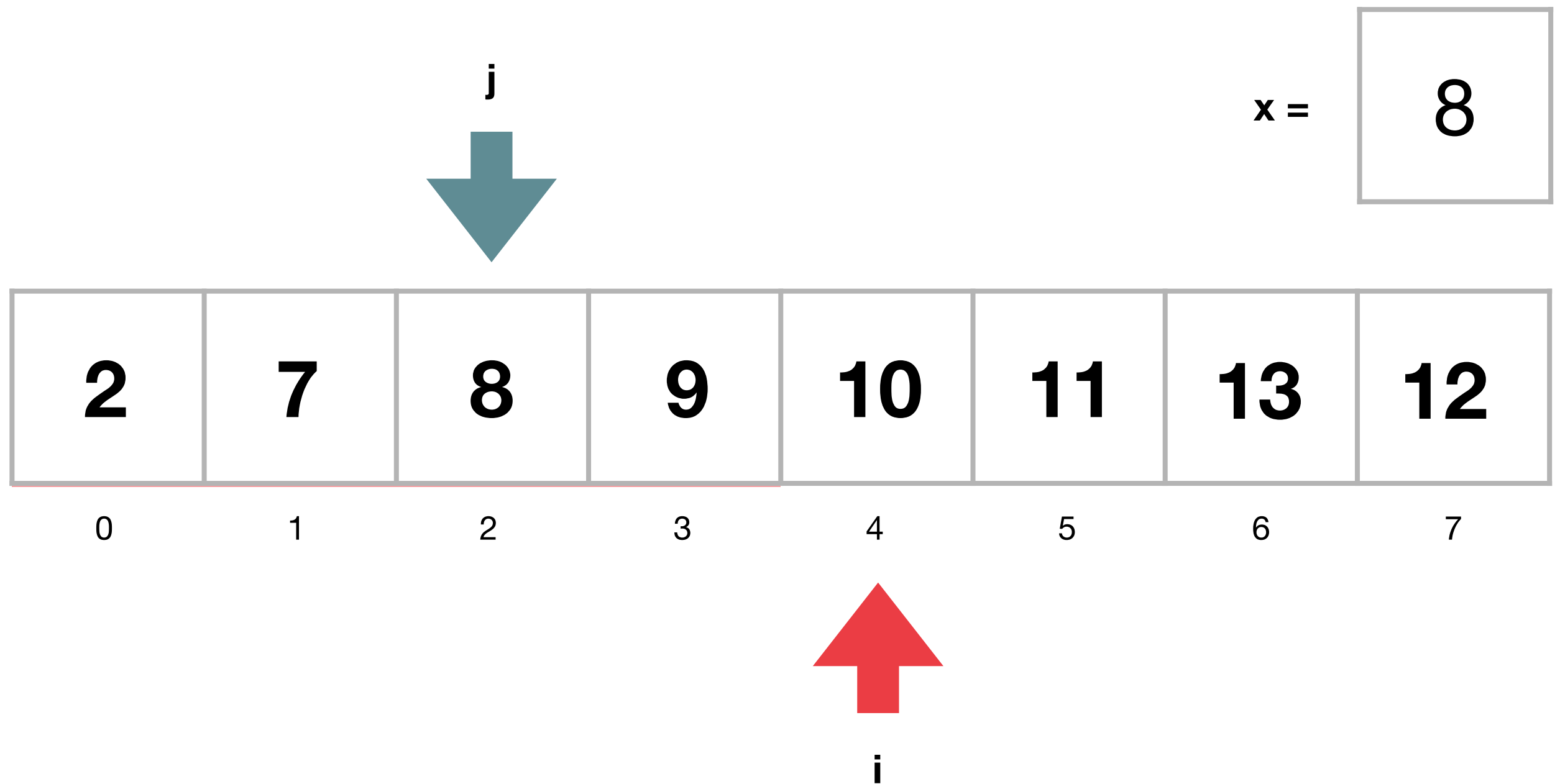
Tri par insertion



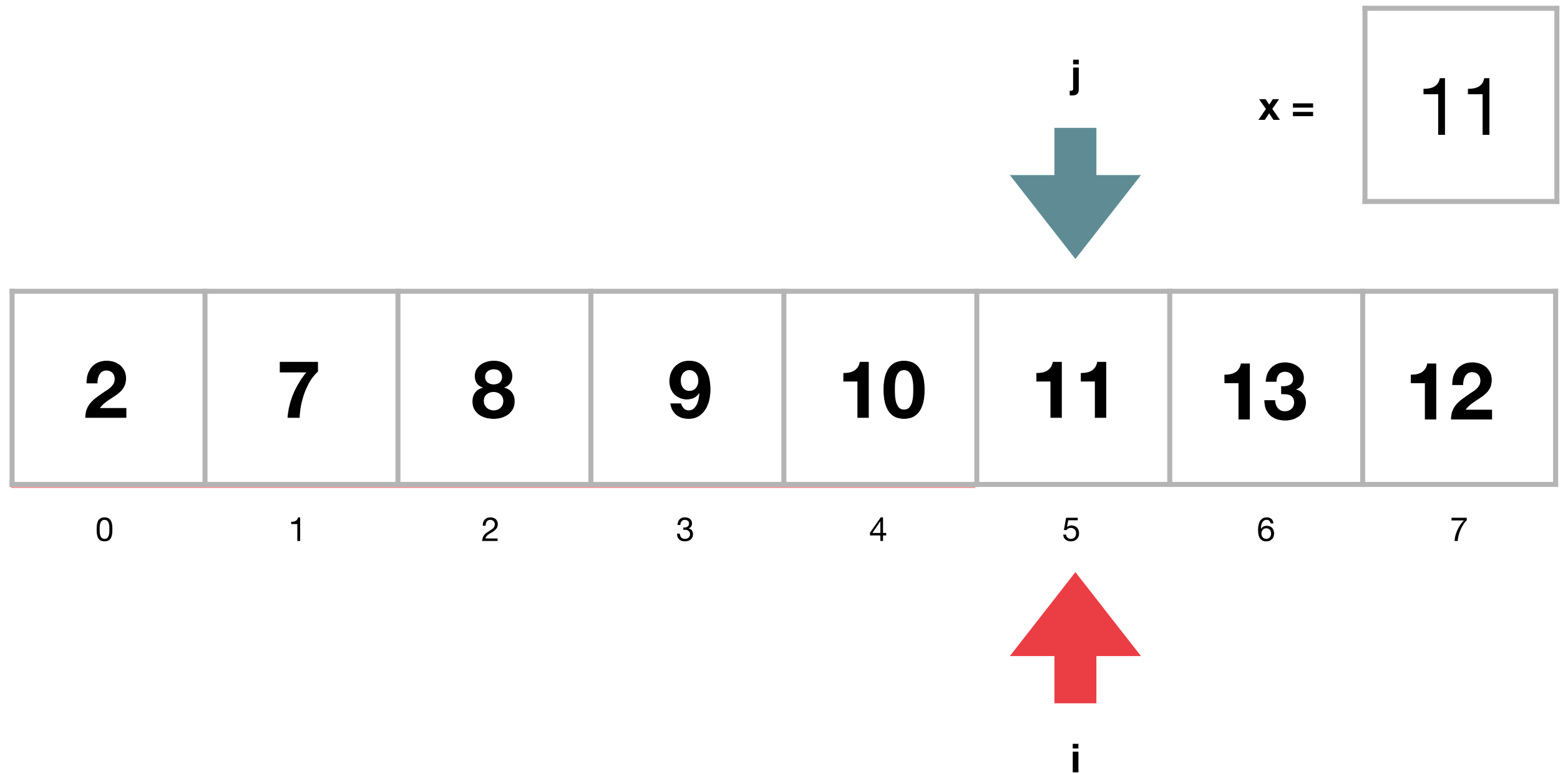
Tri par insertion



Tri par insertion



Tri par insertion



Tri par insertion

x =

13

j



2	7	8	9	10	11	13	12
---	---	---	---	----	----	----	----

0

1

2

3

4

5

6

7



i

Tri par insertion

x =

12

j



2	7	8	9	10	11	13	12
---	---	---	---	----	----	----	----

0

1

2

3

4

5

6

7



i

Tri par insertion

x =

12

j



2	7	8	9	10	11	13	13
---	---	---	---	----	----	----	----

0

1

2

3

4

5

6

7



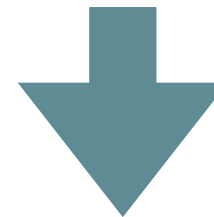
i

Tri par insertion

x =

12

j



2	7	8	9	10	11	12	13
---	---	---	---	----	----	----	----

0

1

2

3

4

5

6

7



i

Tri par insertion

x =

12

j



2	7	8	9	10	11	12	13
---	---	---	---	----	----	----	----

0

1

2

3

4

5

6

7



i

Terminaison

```
procedure trier-par-insertion(T)
  n := longueur(T)
  pour i := 1 à n - 1 faire (on saute le 0)
    x := T[i]
    j := i
    tant que j > 0 et x < T[j - 1] faire
      (décaler d'un élément)
      T[j] := T[j - 1]
      j := j - 1
    fin tant que
    (ici x ≥ T[j - 1] ou bien j = 0)
    T[j] := x
  fin pour
fin procedure
```

- La boucle **pour** termine toujours
- La boucle **tant que** termine (au pire) quand $j = 0$

Correction

```
procedure trier-par-insertion(T)
  n := longueur(T)
  pour i := 1 à n – 1 faire (on saute le 0)
    x := T[i]
    j := i
    tant que j > 0 et x < T[j – 1] faire
      (décaler d'un élément)
      T[j] := T[j – 1]
      j := j – 1
    fin tant que
    (ici x ≥ T[j – 1] ou bien j = 0)
    T[j] := x
  fin pour
fin procedure
```

- Le sous-tableau $T[0, \dots, i - 1]$ est trié au début de la boucle **pour**

Efficacité

```
procedure trier-par-insertion(T)
  n := longueur(T)
  pour i := 1 à n - 1 faire (on saute le 0)
    x := T[i]
    j := i
    tant que j > 0 et x < T[j - 1] faire
      (décaler d'un élément)
      T[j] := T[j - 1]
      j := j - 1
    fin tant que
    (ici x ≥ T[j - 1] ou bien j = 0)
    T[j] := x
  fin pour
fin procedure
```

- $O(n)$ opérations dans le meilleur des cas
- $O(n^2)$ opérations dans le pire des cas