

## 5.4

### tri par insertion

NSI 1ÈRE - JB DUTHOIT

#### 5.4.1 Principe

On insère un à un les éléments parmi ceux déjà trié.

☞ Le **tri par insertion** est aussi appelé "tri naturel" : c'est souvent ce tri que nous utilisons naturellement pour trier un jeu de cartes par exemple.

**VIDÉO** Une animation pour comprendre le tri par insertion - Cliquez ici !

 **Exercice 5.22**

Après avoir étudié la vidéo, donner les différentes étapes du tri par insertion pour le tableau suivant :  
 $T = [8, 5, 1, 9, 7]$

#### 5.4.2 L'algorithme

 **Exercice 5.23**

1. Construire un algorithme d'une fonction `inserer(t,i)` avec  $i$  un entier non nul , et  $t$  un tableau d'entiers où les valeurs de l'indice 0 à  $i-1$  sont triées  
 La fonction doit insérer la valeur  $t[i]$  "au bon endroit" afin que les valeurs de l'indice 0 à l'indice  $i$  soit triées.
2. Implémenter cet algorithme en python.

 **Exercice 5.24**

SAVOIR CONSTRUIRE L'ALGORITHME CORRESPONDANT AU TRI PAR INSERTION  
 On arrive à l'algorithme suivant :

```

1 FONCTION tri_insertion(T :tableau d'entiers)
2   |  POUR i DE 1 A longueur(T) - 1 FAIRE
3     |    inserer(t,i)
  
```

- Implémenter votre algorithme en Python
- Écrire les pré\_conditions
- Écrire les post\_conditions
- Donner un jeu de tests, sous forme de plusieurs assertions

#### 5.4.3 Complexité temporelle

 **Savoir-Faire 5.19**

SAVOIR DÉTERMINER LA COMPLEXITÉ DU TRI PAR INSERTION

Dans le pire des cas, le tableau est rangé dans l'ordre décroissante.

Soit  $T(n)$  Le nombre de comparaisons.

Calculer  $T(n)$ , et en déduire la complexité.

\*\*\*

#### 5.4.4 Terminaison

La boucle POUR ne pose aucun problème !

En revanche, il faut faire attention à la boucle TANT QUE qui peut ne jamais terminer !

La condition de la boucle TANT QUE "est  $j < 0$  and  $T[j - 1] > x$ ".

La boucle s'arrête quant l'une ou l'autre des conditions n'est plus vraies.

Ici, on décrémente  $j$  à chaque fois de, donc nous sommes certains que la condition  $j > 0$  ne sera plus vraie à partir d'une certaine étape !

☞ Notre algorithme s'arrêtera donc !

#### 5.4.5 Validité de l'algorithme

##### Démonstration 2

☛ Invariant de boucle

Considérons la propriété  $P(i)$  : "Le tableau  $T[0], T[1], \dots, T[i-1]$  est trié".

Étape 1 : La propriété est-elle vraie pour  $i=0$  ? :

Étape 2 : On suppose la propriété vraie pour l'entier  $k$ . Est-elle vraie pour l'entier  $k+1$  ?

Etape 3 : Et pour le dernier passage dans la boucle ?

Étape 4 : Conclusion