

Algorithmes de tri

Introduction

La fonction d'un algorithme de tri est d'organiser un ensemble de données selon un ordre déterminé. Généralement les données à trier sont des **nombre**s ou des **chaînes de caractère**, stockés dans des **listes**, qui peuvent être classés numériquement ou alphabétiquement.

Le tri est un outil essentiel, plus particulièrement dans le domaine de la gestion et de l'analyse des données. Il existe de nombreux types d'algorithmes, plus ou moins performants.

Parmi les principales caractéristiques qui permettent de différencier les algorithmes on peut citer :

- La complexité (temps maximum de traitement et temps moyen de traitement).
- Les ressources nécessaires à son fonctionnement (en particulier l'espace mémoire).

Un tri est précédé par une **acquisition de données** et suivi par une **restitution de données**. L'acquisition et la restitution de données doivent faire l'objet de procédures en relation avec l'interface.



Le tri par insertion

Le tri par insertion est le tri pratiqué intuitivement pour classer un jeu de cartes. Il consiste à ranger toutes les cartes, l'une après l'autre, dans l'ordre de la donne au bon endroit parmi les cartes déjà triées.

Principe de l'algorithme de tri par insertion pour un classement dans l'ordre croissant :

Placer la première carte de la donne dans le tas trié

Pour chaque carte restante de la donne :

Tant que la valeur de la carte à classer est supérieure à la valeur de la carte en cours du tas trié :
Avancer à la carte suivante du tas trié

Exemple 1 : Tri par insertion croissant des valeurs (6, 5, 3, 2, 4) en 4 étapes :

| | | | | | |
|-----------------|---|---|---|---|---|
| valeurs à trier | 6 | 5 | 3 | 2 | 4 |
| en étape 1 | 5 | 6 | 3 | 2 | 4 |
| étape 2 | 3 | 5 | 6 | 2 | 4 |
| étape 3 | 2 | 3 | 5 | 6 | 4 |
| étape 4 | 2 | 3 | 4 | 5 | 6 |

En rouge valeurs non triées

En bleu valeurs triées

Exemple 2 :

Lien vidéo : <https://youtu.be/ROaIU379I3U>

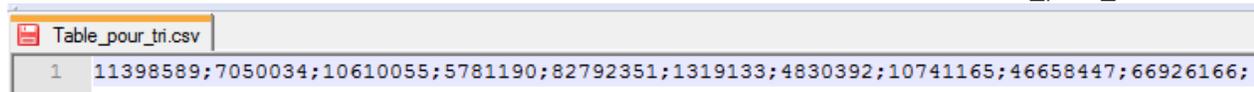
Description en pseudo-code d'un l'algorithmme de tri par insertion par valeurs croissantes.

Les éléments de la liste L (de taille n) sont numérotés de 0 à n-1.

```
fonction trier_liste_asc(L)
  n ← Taille(L)
  POUR i de 1 à n-1:
    # mémoriser L[i] dans x
    x ← L[i]
    # décaler vers la droite les éléments de L[0]..L[i-1]
    # qui sont plus grands que x (en partant de L[i-1])
    j ← i
    TANT QUE L[j-1] > x et j > 0 FAIRE
      L[j] ← L[j - 1]
      j ← j - 1
    FIN TANT QUE
    # placer x dans le "trou" laissé par le décalage
    L[j] ← x
  Retourner L
```

Exemple de programme python d'acquisition / tri par insertion / restitution : « **exemple_tri.py** »

ACQUISITION : Les données à classer sont stockées dans un fichier csv nommé « Table_pour_tri.csv » :



```
Table_pour_tri.csv
1 11398589;7050034;10610055;5781190;82792351;1319133;4830392;10741165;46658447;66926166;
```

TRI : par insertion, par valeurs croissantes

RESTITUTION : création d'un fichier texte avec titre, affichage des données en colonne, lignes numérotées.

```
#!/usr/bin/env python3
```

```
def lire_fichier(nom_fic):
    """
    Lecture d'un fichier csv contenant une ligne de données séparées par des ;
    nom_fic : (type str) nom du fichier à ouvrir (avec l'extension)
    La fonction retourne une liste d'entiers
    """
    fichier_source=open(nom_fic,'r') # Ouverture du fichier csv en lecture seule
    liste_donnees = []
    for ligne in fichier_source:
        donnees = ligne.split(';') #donnees est une liste de chaînes de caractère
    for n in donnees:
        if n.isdigit(): #si la donnée représente un nombre
            liste_donnees.append(int(n)) #liste_donnees est une liste d'entiers
    fichier_source.close() # Fermeture du fichier
    return liste_donnees
```



```

def creer_fichier_texte(nom_fic,tableau):
    """
    Creation d'un fichier texte dans le répertoire courant.
    paramètre nom_fic : (type str) nom du fichier.txt à créer (sans l'extension)
    paramètre tableau : (type list) liste d'entiers
    Le fichier contient :
        - un titre.
        - la liste numérotée croissante des pays avec la population.
    """
    fichier_cible = open(nom_fic+".txt",'w')
    fichier_cible.write("Liste triée\n") #écriture du titre
    i=1 #compteur pour la numérotation
    for valeur in tableau:
        fichier_cible.write(str(i)+" "+str(valeur)+"\n")
        i=i+1
    # Fermeture du fichier
    fichier_cible.close()
    print("Le fichier \"",nom_fic+".txt","\n" a été créé.")

def trier_liste_asc(L):
    """
    Méthode : tri par insertion croissant.
    paramètre : L est une liste d'entiers.
    La fonction retourne la liste triée.
    Exemple :
    >>> trier_liste([10,20,5,15])
    [5, 10, 15, 20]
    """
    n = len(L)
    for i in range(1,n-1):
        # mémoriser table[i] dans x
        x=L[i]
        j=i
        # décaler vers la droite les éléments de table[0]..table[i-1]
        # qui sont plus grands que x (en partant de table[i-1])
        while L[j-1]>x and j>0 :
            L[j]=L[j-1]
            j=j-1
        L[j]=x
    return L

#----- PROGRAMME PRINCIPAL -----#
#ACQUISITION des données et rangement dans la liste tableau_donnees
tableau_donnees = lire_fichier("Table_pour_tri.csv ")
#si le tableau n'est pas vide, alors TRI par insertion
#puis RESTITUTION dans un fichier texte
if len(tableau_donnees)>0:
    trier_liste(tableau_donnees)
    creer_fichier_texte("tableau_trie",tableau_donnees)
#si le tableau est vide on affiche qu'aucune donnée n'est disponible
else:
    print("Il n'y a aucune donnée à trier")

if __name__ == "__main__":
    import doctest
    doctest.testmod()

```

