

Algorithmes de tri d'une liste

Tri dans l'ordre croissant

1^{ère} méthode : tri direct

Dans chaque tri, on lit entièrement la liste et on inverse au fur et à mesure deux éléments consécutifs si l'élément suivant est inférieur à l'élément précédent.

Et on recommande, jusqu'à ce qu'un indicatif nous prévienne qu'il n'y a plus de modification.

Exemple :

```
def tri_liste(a):
    test = 0
    for i in range(len(a)-1):
        if a[i] > a[i+1]:
            test = 1
            a[i] , a[i+1] = a[i+1] , a[i]
    if test == 1:
        tri_liste(a)
    return a

a = [9,84,25,36,14,5,29,68]
print(a)
b = []
b = tri_liste(a)
print(b)
```

On obtient :

```
[9, 84, 25, 36, 14, 5, 29, 68]
[5, 9, 14, 25, 29, 36, 68, 84]
```



2^{ème} méthode : tri-fusion

On crée deux demi-séries à partir de la série initiale, on les trie puis on prend les éléments successifs de chacune des séries pour constituer la série attendue.

Exemple :

```
def coupe_liste(a):
    b = []
    c = []
    for i in range(len(a)//2):
        b.append(a[i])
    for i in range(len(a)//2,len(a)):
        c.append(a[i])
    return b,c

def tri_liste(a):
    test = 0
    for i in range(len(a)-1):
        if a[i] > a[i+1]:
            test = 1
            a[i] , a[i+1] = a[i+1] , a[i]
    if test == 1:
        tri_liste(a)
    return a
```



```
def coll_listes(a,b):
    c = []
    a.append(10**1000)    # petite ruse
    b.append(10**1000)
    for i in range (len(a)+len(b)-2):
        if a[0] < b[0]:
            c.append(a[0])
            del a[0]
        else:
            c.append(b[0])
            del b[0]
    return c
```

```
a = [9,84,25,36,14,5,29,68]
print(a)
b,c,d,e,f = [],[],[],[],[]
b,c = coupe_liste(a)
print(b)
print(c)
d = tri_liste(b)
e = tri_liste(c)
print(d)
print(e)
f = coll_listes(d,e)
print(f)
```

On obtient :

```
[9, 84, 25, 36, 14, 5, 29, 68]
[9, 84, 25, 36]
[14, 5, 29, 68]
[9, 25, 36, 84]
[5, 14, 29, 68]
[5, 9, 14, 25, 29, 36, 68, 84]
```

3^{ème} méthode : tri par insertion

intéressant sur les données de petite taille

Chaque donnée de la liste est successivement insérée à la bonne place parmi les données déjà triées.

```
def tri_insertion(n,liste):
    i = 0
    test = 0
    while test == 0 and i < len(liste):
        if n < liste[i]:
            liste.insert(i,n)
            test = 1
        i += 1
    if test == 0:
        liste.append(n)
    print(liste)
    return(liste)
```

```
a = [9,84,25,36,14,5,29,68]
liste = []
```

```
liste.append(a[0])
for i in range(1,len(a)):
    liste = tri_insertion(a[i],liste)
print(liste)
```

On obtient à partir de la liste [9,84,25,36,14,5,29,68] :

[9, 84]

[9, 25, 84]

[9, 25, 36, 84]

[9, 14, 25, 36, 84]

[5, 9, 14, 25, 36, 84]

[5, 9, 14, 25, 29, 36, 84]

[5, 9, 14, 25, 29, 36, 68, 84]

[5, 9, 14, 25, 29, 36, 68, 84]