exercice 1 : Tri Bulle

On souhaite trier le tableau T=[1,4,8,5,3,2,7,6].
On fournit les fonctions suivantes :

```
def echange(T,i,j):
    T[i],T[j] = T[j],T[i]

def bulle(T):
    N=len(T)
    for i in range (N):
        if T[i]>T[i+1]:
            echange(T,i,i+1)
    return T

T=[1,4,8,5,3,2,7,6]
```

1. Essayez d'appliquer la fonction bulle au tableau T.
   La « condition d'arrêt » dans le pour est fausse. Proposez une correction.
2. Quel effet produit la fonction bulle corrigée ?
   Combien de fois faudrait-t-il l'appliquer pour obtenir le tableau T trié ?
3. Complétez le code de la fonction bulle de manière à ce qu'elle retourne systématiquement le tableau trié. Combien de comparaisons effectue-t-on ?

exercice 2 : Implémentation du tri fusion

Comme le tri rapide, le tri fusion applique le principe du « diviser pour mieux régner ».
Il partage arbitrairement les éléments à trier en deux sous ensembles de même taille (sans les comparer) afin d'éviter le pire cas du tri rapide où les deux sous ensembles sont de taille disproportionnées.
Une fois les deux parties triées récursivement, il les fusionne.

exemple sur T=[4,12,8,5,9,6,13,3].

pour trier T[0 :8] on trie T[0 :4] et T[4 :8]	4	12	8	5	9	6	13	3
pour trier T[0 :4] on trie T[0 :2] et T[2 :4]	4	12	8	5				
pour trier T[0 :2] on trie T[0 :1] et T[1 :2]	4	12						
on fusionne T[0 :1] et T[1 :2] triés								
on fusionne T[0 :2] et T[2 :4] triés								
on fusionne T[0 :4] et T[4 :8] triés								

1. Pour implémenter cette méthode de tri sous Python, vous avez besoin d'écrire deux fonctions :
   • une fonction fusion qui prend en entrée deux tableaux T1 et T2 supposés triés et renvoie un tableau T contenant les mêmes éléments que T1 et T2 mais rangés par ordre croissant.
   • la fonction récursive de tri qui, si le tableau ne contient qu'un élement le renvoie tel quel (puisqu'il est déjà trié) et sinon renvoie le résultat de la fusion de sa *partie gauche* triée et de sa *partie droite* triée.
2. Comparaison des temps d'execution :
   En important la bibliothèque random, vous pourez utiliser la fonction randint(a,b) qui renvoie un entier aléatoire entre a et b.
   Créer un tableau T de 10000 valeurs aléatoires.
   L'objectif est de comparer les temps mis par des algorithmes de tris différents appliqués au même tableau de données T.
   Pour cela vous pourrez utiliser la fonction time de la librairie time.
   d=time() *heure de départ*
   tache a effectuer
   print(time()-d) *heure d'arrivée - heure de départ*
3. *Pour aller plus loin :*
   idée 1 : Optimisez le tri fusion lorsque tous les éléments du tableau de gauche sont inférieurs au tableau de droite.
   idée 2 : Afin entre autre de limiter les appels récursifs lors du tri fusion, on peut choisir d'utiliser le tri par insertion lorsque le tableau à trier est de taille par exemple inférieure à 5...