Algorithmes de tri

Nous allons dans ce TP revoir le tri par insertion, et découvrir deux nouveaux algorithmes : le tri fusion et le tri rapide.

Pour tester l'efficacité de nos algorithmes, on pourra utiliser utiliser le code suivant :

```
import time
l = ... # mettre ici une liste
t = time.time()
l_tri = tri(l) # on trie la liste
print("Temps :",time.time() - t)
```

1 Le tri par insertion (Insertion sort)

Cet algorithme de tri est à connaître.

Rappelez le principe et le code du tri par insertion.

2 Le tri fusion (Merge sort)

On a ici le cas typique d'une fonction récursive. Voilà le principe du tri fusion :

- On divise la liste à trier en deux listes de tailles (environ) égales
- Récursivement, on trie les deux sous-listes
- On fusionne les deux sous-listes triées en une liste triée, en utilisant l'algorithme suivant :
  - on note $a_1, \dots, a_n$ et $b_1, \dots, b_p$ les deux listes à trier
  - si l'une est vide, on renvoie l'autre
  - sinon, on compare a_1 et b_1 ; si $a_1 \leqslant b_1$, alors on renvoie a_1 suivi de la fusion de $a_2, \dots, a_n$ et $b_1, \dots, b_p$, et sinon, on renvoie b_1 suivi de la fusion de $a_1, \dots, a_n$ et $b_2, \dots, b_n$.

Implémenter le tri fusion.

3 Le tri rapide (Quick sort)

Cet algorithme de tri est à connaître.

Cet algorithme est une variation du tri fusion. Voici le principe :

- on choisit un élément de la liste, qu'on appellera le *pivot*, qu'on exclut de la liste
- on divise la liste à trier en deux listes : la liste des éléments inférieurs au pivot, et celle des éléments strictement plus grands

- on trie récursivement les deux sous-listes
- on concatène les deux sous-listes.

Implémenter le tri rapide

4 Sedgesort

C'est une version modifiée du tri rapide : au lieu de descendre jusqu'à des listes de taille 1, on trie les petites sous-listes en utilisant le tri par insertion.

Implémenter le sedgesort.

5 Calcul de médiane

On va utiliser le principe du tri rapide pour trouver le i-ième plus petit élément d'une liste, sans la trier complètement. On utilise le principe suivant :

- on paritionne la liste comme pour le tri rapide, avec un pivot
- on note k la longueur de la liste des petits éléments
- si $i = k$, alors on renvoie le pivot
- si $i < k$, on cherche le i-ième plus petit élément dans la liste des petits éléments
- si $i > k$, on cherche le $(i - k - 1)$-ième plus petit élément dans la liste des grands éléments

Implémenter cet algorithme, puis écrire une fonction calculant la médiane d'une liste.