Préparation à l'épreuve pratique NSI	13
Tri fusion d'une liste	
Nom :	**Note : / 20**
	Classe :

1 Spécification du programme à réaliser

1.1 Description générale

On souhaite trier un tableau de n éléments ici numéroté de 1 à n. L'algorithme est naturellement décrit de façon récursive.

1. Si le tableau n'a qu'un élément, il est déjà trié.
2. Sinon, séparer le tableau en deux parties à peu près égales.
3. Trier récursivement les deux parties avec l'algorithme du tri fusion.
4. Fusionner les deux tableaux triés en un seul tableau trié.

Entrées	Sorties
Un tableau T [1,…,n] de n éléments	Le tableau trié.

1.2 Pseudo code de l'algorithme[1]

```
fonction triFusion(T[1, …, n])
si n ≤ 1
        renvoyer  T
sinon
        renvoyer
                fusion (  triFusion(T[1, …, n/2]) , triFusion(T[n/2 + 1, …, n]) )
```

La fonction fusion a été définie dans le travail précédent il faudra la réutiliser ici.

1.3 Amélioration

- On traitera le cas où la liste donnée en paramètre ne contient aucun élément. La fonction retournera la valeur **N**one dans ce cas.

[1] Voir https://fr.wikipedia.org/wiki/Tri_fusion

2 Codage et mise en œuvre

2.1 Script de l'exercice

NSI-PROG-013-Tri-Fusion.py

2.2 Fonction à compléter

```
## Votre fonction à réaliser
def tri_par_fusion(liste):
    '''
    Voir fiche n°006
    '''
    return ....

def fusion_de_deux_listes(liste_A, liste_B):


    return ...
```

Point particulier : vous noterez là encore l'utilisation d'une fonction qui s'appelle elle-même. C'est la programmation récursive.

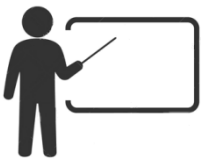
2.3 Résultats attendus

```
>>> (executing lines 1 to 98 of "ALGO_Tri-Par-Fusion-corrige.py")
Liste_A :  [-13, -8, -2, 10, 41, 52, 56]
Liste_B :  [-10, 8, 52, 84, 250]
Listes fusionnées    :  [-13, -10, -8, -2, 8, 10, 41, 52, 52, 56, 84, 250]

Liste de départ :  [10, 70, -77, -78, -64, -74, -47, -34, 8, 92, 91, 37, 22, 25, 20]
Liste triée     :  [-78, -77, -74, -64, -47, -34, 8, 10, 20, 22, 25, 37, 70, 91, 92]

Liste de départ :  []
Liste triée     :  None

Liste de départ :  [24]
Liste triée     :  [24]
```



2.4 Un point sur la complexité

Le **tri fusion** est un algorithme de tri par comparaison *stable*. Sa complexité temporelle pour une entrée de taille *n* est de l'ordre de *n* log *n*, ce qui est asymptotiquement optimal. Ce tri est basé sur la technique algorithmique *diviser pour régner*.

L'opération principale de l'algorithme est la *fusion*, qui consiste à réunir deux listes triées en une seule. L'efficacité de l'algorithme vient du fait que deux listes triées peuvent être fusionnées en temps linéaire.

Un algorithme de tri *stable* est un algorithme de tri conservant l'ordre initial de deux éléments égaux.