I Tri fusion

Le tri fusion permet de trier une liste l récursivement, de la façon suivante :

- Diviser l en deux listes l_1 et l_2.
- Trier récursivement l_1 et l_2 pour obtenir l_1' et l_2'.
- Fusionner l_1' et l_2' pour obtenir l', qui est une liste triée des éléments de l.

1. Écrire une fonction `divise : 'a list -> 'a list * 'a list` qui sépare une liste en deux (à $\pm$ 1 près).
   Exemple : `divise [1;2;3;4;5]` peut renvoyer `([1;3;5], [2;4])`.
2. Écrire une fonction `fusion : 'a list -> 'a list -> 'a list` qui fusionne deux listes triées en une seule liste triée.
   Exemple : `fusion [1;3;5] [2;4]` doit renvoyer `[1;2;3;4;5]`.
3. Écrire une fonction `tri : 'a list -> 'a list` qui trie une liste par le tri fusion.
   Exemple : `tri [5;4;3;2;1]` doit renvoyer `[1;2;3;4;5]`.
4. Montrer que la complexité de `tri` est $O(n \log n)$ sur une liste de taille n.

II Nombre de Catalan

afficher$_a$*nagrammesaveccalculer*$_o$*ccurrences*

III Zigzag

https://leetcode.com/problems/longest-zigzag-path-in-a-binary-tree