DIVISER POUR RÉGNER

Nous avons illustré la technique de programmation consistant à diviser pour régner avec deux exemples : **la recherche dichotomique** et **le tri fusion.**

– L**a recherche dichotomique** s´effectue dans une liste triée. On commence par regarder la valeur médiane de la liste (au milieu de la liste), puis la valeur médiane de la sous-liste et ainsi de suite, jusqu´à trouver l´élément recherché. A chaque itération, on divise par deux la taille de la liste. Dans le pire des cas, nous avons divisé par deux la taille de la liste jusqu´á obtenir une liste de taille inférieure à un. Nous avons vu que cela quantifie le logarithme de base 2: Nous avons donc une complexité logarithmique en $O(\log_2(n))$

– Les méthodes de tris vues en première (sélection, insertion) utilisaient deux boucles imbriquées, leur complexité était toujours quadratiques, c´est à dire en $O(n_2)$. Nous avons appliqué la méthode *diviser pour régner* à l´étude d´une nouvelle méthode de tri, le **tri fusion** (merge sort).

> – On commence par **partitionner** la liste jusqu´à obtenir des listes d´um seul élément. Comme on divise la liste en deux parties égales, $\log_2(n)$ étapes sont effectuées. On remarque ici le lien avec la dichotomie et l´utilisation de la méthode diviser pour règner.
> La complexité de cette approche est *exponentielle* et donc bien trop coûteuse en termes de temps de calcul.
>
> – On procède ensuite à la **fusion** : Cette phase consiste à faire des comparaisons entre les premiers éléments de chaque liste à fusionner. on peut donc supposer que pour un tableau de n éléments, on aura n comparaisons.

En tenant en compte les deux étapes de cette méthode tri, on peut dire que la complexité du tri-fusion est en $O(n\times \log_2(n))$.