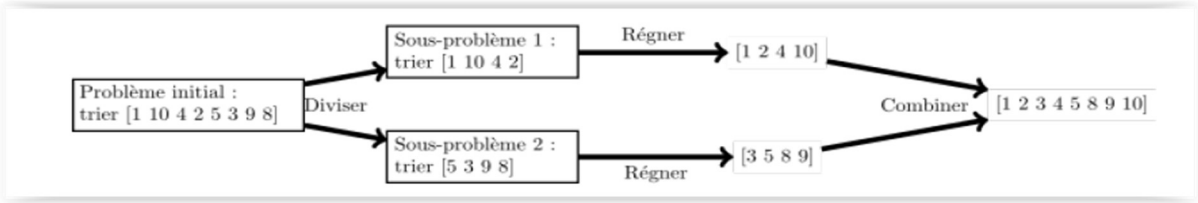
Algorithmique et structures de données
TD2 – recherche et tris performants

Lamia BENAMARA

Consigne :
Pour les séances TD, tous les algorithmes demandés sont à écrire en langage algorithmique, consulter la fiche algo disponible sur le drive pour voir la syntaxe ainsi que des exemples complets.

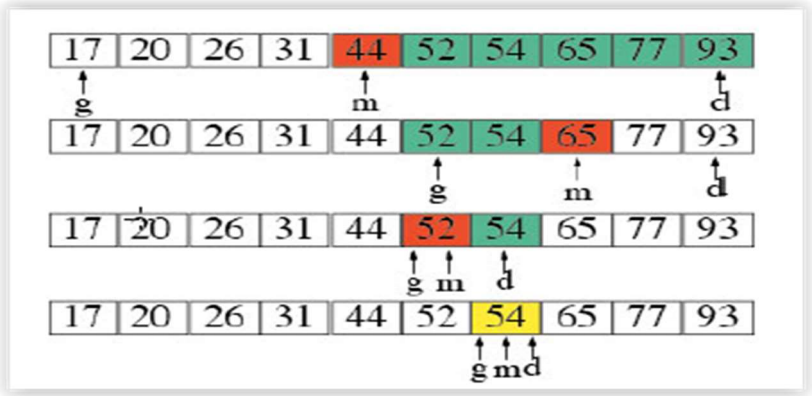
Rappel : Principe «Diviser pour régner »
En informatique « diviser pour régner » est une famille de techniques qui fournit des algorithmes efficaces pour de nombreux problèmes, comme pour trier les éléments d'un tableau. Une telle technique consiste à :

- Diviser : découper un problème initial en sous-problèmes.
- Régner : résoudre les sous-problèmes, plus simples car de plus petites tailles. si sous-problèmes encore trop « gros », alors on les redécoupe... et ainsi de suite de manière récursive.
- Combiner : composer une solution au problème initial à partir des solutions des sous-problèmes.



Exercice 1 : Recherche dichotomique

1) Ecrire un algorithme *itératif* qui implémente la recherche dichotomique d'un élément e dans un tableau 1D d'entiers trié.
2) Ecrire un algorithme *récursif* qui implémente la recherche dichotomique d'un élément e dans un tableau 1D d'entiers trié
3) Quelle est la complexité de chaque algorithme ?
4) Modifiez l'algorithme précédent de façon à trouver la position de la dernière occurrence de l'élément recherché.
5) En utilisant l'algorithme de recherche dichotomique sur un tableau 1D, écrire l'algorithme *récursif* qui implémente la recherche dichotomique sur un tableau 2D ayant chaque ligne triée (le dernier élément de chaque ligne est inférieur au premier élément de la ligne suivante).



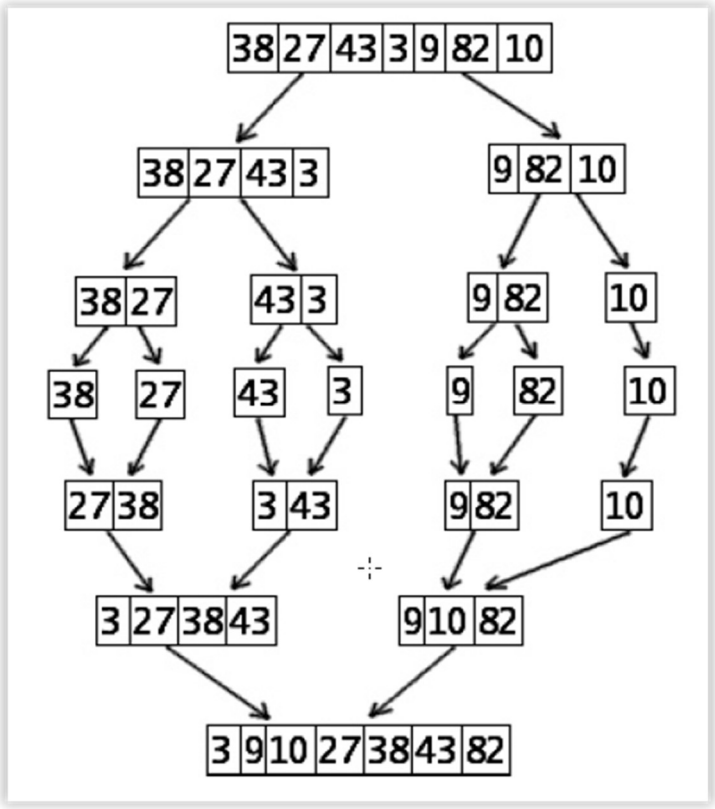
Exercice 2 : Tri par fusion
On étudie maintenant un des plus efficaces méthodes de tri: **tri fusion**, qui est une des variantes de **tri dichotomique**. On utilise la méthode récursive.

Idée: Pout trier un tableau :
1) trier deux sous-tableaux;
2) fusionner les deux sous-tableaux triés pour avoir le bon ordre.

Commençons par la deuxième procédure, la plus simple.
La fusion: On examine le dernier élément de chaque tableau et on garde le plus grand comme résultat, on l'inscrit en dernière place dans le tableau résultat. On passe à l'avant dernier élément du tableau dans lequel on a pris le résultat et on le compare au dernier élément de l'autre tableau et ainsi de suite jusqu'à arriver à la fin d'un des deux tableaux.

- Ecrire un algorithme tri_Fusion qui prend en paramètre un tableau d'entiers non vide et sa taille, et qui en fin de traitement assure que les éléments du tableau sont triés dans l'ordre croissant.
- Quelle est la complexité de ce tri ?

Un exemple détaillé pas à pas est donné ci-dessous :



Exercice 3 : Tri rapide (quick sort)

Ecrire un algorithme récursif qui permet implémente le tri rapide.
Pour rappel, le tri rapide consiste à :

- Choisir (arbitrairement, dans l'exemple, on a choisi le dernier) un élément que l'on appellera pivot.
- On placera ensuite ce pivot à sa place finale :
  > les éléments qui lui sont inférieurs seront placés à sa gauche
  > les éléments qui lui sont supérieurs à sa droite.
- Cette opération s'appelle **le partitionnement**.
- Après le partitionnement, on recommencera la même opération récursivement sur les deux sous-tableaux (celui à la gauche du pivot, et celui à sa droite).
- Lorsque tous les sous-tableaux auront été partitionnés, le tableau sera trié.