

Exercice 1. Tri par insertion

Le *tri par insertion* est l'un des tri les plus basique. Il consiste à ranger l'élément le plus grand à la fin du tableau et ainsi de suite pour le deuxième plus grand élément.

- a. Ecrire l'algorithme trouvant le maximum d'un tableau.
- b. Ecrire l'algorithme échangeant deux valeurs du tableau.
- c. Ecrire l'algorithme du tri par insertion en entier.
- d. Combien votre algorithme effectue-t-il de comparaisons en fonction de la taille n du tableau ?

Exercice 2. Tri bulle

Le *tri à bulle* ou tri par propagation est un tri très simple à mettre en oeuvre. Il consiste à faire remonter le plus grand élément du tableau (comme une bulle d'air remonte à la surface) en comparant les éléments successifs.

- a. Ecrire l'algorithme permettant de faire remonter le plus grand élément du tableau.
- b. Ecrire l'algorithme permettant de trier le tableau entier.
- c. Combien votre algorithme effectue-t-il de comparaisons en fonction de la taille n du tableau ?
- d. Si ce n'est pas déjà fait, proposer une amélioration (simple) permettant de diminuer le nombre de comparaisons à $n(n-1)/2$.

Exercice 3. Tri fusion

Le *tri fusion* repose sur le fait que pour fusionner deux tableaux triés dont la somme des longueurs des n , $n-1$ comparaisons au maximum sont nécessaires. Il existe deux approches à cet algorithme : une première version est récursive, elle permet à une procédure de s'appeler elle-même pour résoudre le problème, l'autre est itérative, il s'agit de faire un tri fusion "en place". Pour trier un tableau T de taille n , l'algorithme est :

- Couper T en deux sous-tableaux T_1 et T_2 de taille $n/2$;
 - Trier T_1 et T_2 ;
 - Fusionner T_1 et T_2 ;
- a. Ecrire une procédure (important) fusionnant deux tableaux.
 - b. Ecrire la procédure implémentant le tri-fusion.