
MPSI - ITC - TD 6 : ÉTUDE D'ALGORITHMES

Exercice 1 (Tri à bulles)

On se propose d'étudier un algorithme, qu'on appelle le *tri à bulles*, dont la description en français est la suivante

Tant que la liste n'est pas triée :

 Parcourir la liste de gauche à droite :

 Si on trouve deux éléments consécutifs tels que le premier est plus grand que le deuxième, on les échange entre eux.

1. Écrire une fonction `est_triee` qui prend en entrée une liste d'entiers, et renvoie un booléen indiquant si la liste en entrée est triée dans l'ordre croissant ou pas.
2. Écrire une fonction `tri_bulles` qui implémente l'algorithme *tri à bulles* décrit en introduction. La structure de votre fonction sera une boucle `while` à l'intérieur de laquelle il y a une boucle `for`.
3. Proposez un variant permettant de prouver la terminaison de la boucle `while`.
4. Grâce au variant proposé, écrire une fonction `tri_bulles_for` qui implémente l'algorithme du *tri à bulles* sans utiliser de boucle `while` (la structure de votre fonction sera deux boucles `for` l'une dans l'autre).
5. Proposer un invariant pour la boucle extérieure permettant de prouver que votre fonction résout correctement le problème du tri. Si vous manquez d'inspiration, affichez l'état de la liste à la fin de chaque itération. Que constatez-vous sur la fin de la liste ?
6. Grâce à votre invariant, écrire une fonction `tri_bulles_optimal` qui consiste en deux boucles `for` imbriquées, mais où les boucles `for` ne s'exécutent pas plus que nécessaire (en particulier, la boucle intérieure ne parcourt pas toute la liste).
7. Majorer le nombre d'affectations que fait votre fonction par un polynôme en la taille de la liste en entrée.

Cet exercice nous montre qu'étudier les algorithmes non seulement permet de montrer qu'ils terminent et qu'ils sont corrects, mais ça permet au passage de les rendre plus explicites (question 4) voir plus efficaces (question 6).