Démo 9 : Tri

Tri

1. Triez la liste suivante en utilisant les tries vues en classe : sélection, insertion, bulles.

   12, 4, 7, 3, 6, 8, 2, 10, 9

   Montrez la liste après chaque échange de valeurs.

2. Le code pour les quatre tries vues en classe est disponible sur le site. Téléchargez-le. Exécutez le code et consultez-le.
3. Ce code contient une deuxième version du `triRapide` (`triPlusRapide`, du moins, nous espérons qu'il est plus rapide). Ce tri utilise un deuxième algorithme de tri quand le tableau devient petit. Du code affiche le temps d'exécution pour cet algorithme sur un tableau de valeurs aléatoires (code de la méthode `testCharge`). Essayez le code pour une `TAILLE` de 10 000, 100 000, 1 000 000 et 10 000 000. Prenez note des temps d'exécutions.
4. Une constante ***BARRIERE*** indique à quelle taille l'algorithme de tri utilise `triInsertion` plutôt que `triPlusRapide`. Modifier le programme pour faire des tests avec les valeurs 0, 5, 10, 15, 20 et 25 pour la constante ***BARRIERE*** avec un tableau de 10 000, 100 000, 1 000 000 et 10 000 000 éléments (astuce : placez le tout dans une boucle qui modifie la constante, donc modifiez la classe pour que les constantes ne soient plus des constantes). Déterminez la valeur de ***BARRIERE*** qui est optimale.
5. Refaites les mêmes testes en remplaçant le `triInsertion` par un `triSelection` à l'intérieur de la méthode `triPlusRapide`.
6. La librairie `Arrays` de Java contient des méthodes de recherche et de tri. Testez la méthode `Arrays.`*`sort`* avec des tableaux de taille 10 000, 100 000, 1 000 000 et 10 000 000. Comparez les résultats avec ceux de nos méthodes.