Aix*Marseille université

Domaine Sciences et Technologies
LICENCE INFORMATIQUE

Algorithmique 1 : TP tris
Code UE : S04IN4A2

Année 2019-2020

TP tris

L'objectif de ce TP est de comparer expérimentalement différents tris. Vous allez implémenter les tris qui nous intéresse, puis vous comparerez les temps d'exécution de chacun sur des suites de longueurs diverses. Ces suites sont constituées d'éléments tous de même type et qui sont comparables. Vous pourrez par exemple trier des suites d'entiers, de chaînes de caractères,... ou tout autres objets dont la classe implémente l'interface `Comparable`. Ce TP s'étends sur 4 semaines. Lors de la dernière séance vous montrerez votre travail au chargé de TP pour évaluation.

Vous implémenterez les tris suivants :
- tri à bulle,
- tri par fusion,
- tri rapide avec la partition dite du *drapeau*,
- tri par tas
- (tri par base.)

Suivez l'ordre indiqué dans la liste pour l'implémentation. Tous ces tris ont été ou seront présentés en cours.

Pour la plupart de ces tris vous représenterez les suites à trier par des `ArrayList`, et vous accèderez aux éléments de la suite avec les méthodes `get` et `set`. Vous pourrez définir l'opération `permuter` qui permute deux éléments dans la liste à partir de leurs positions courantes. Dans certains cas nous avons présenté l'algorithme de tri avec des suites de la forme $e_1, e_2, \ldots, e_n$. L'indexation dans `ArrayList` débute à l'indice 0, vous devrez dans ce cas adapter l'algorithme du cours.

Génération des suites

Afin de comparer les tris sur des suites de grandes tailles vous écrirez des fonctions de génération (dépendant du type des objets de la suite que vous voulez générer, entiers, caractères,...). `java.util.Random` permet de générer des valeurs aléatoires. Vous écrirez au moins une fonction pour générer des suitesde nombres entiers qui prendra en argument les bornes entre lesquelles sont choisies les valeurs et leur nombre.

Tri par fusion

Pour ce tri vous représenterez les suites à trier avec des listes chainées. La classe `LinkedList` propose toutes les méthodes dont vous aurez besoin, notamment `addAll` qui ajoute tous les éléments d'une collection à la liste à partir de laquelle la méthode est appelée. Vous devrez écrire une fonction pour partitionner une liste chainée en deux listes de longueurs égales à un élément près, et une fonction qui fait le fusion de deux listes ordonnées.