LI 2 - Semestre 4 Octobre 2024

Université d'Artois
Faculté des Sciences Jean Perrin

Algorithmique et Programmation 3

TP3 : Algorithmes de tri

1. Implémenter les algorithmes de tri simples vu en cours : tri par selection, tri par insertion, et tri à bulles.
2. Générer aléatoirement plusieurs tableaux en variant leurs tailles (100, 1000, 10000, etc.) et comparer le temps mis par les trois algorithmes.
3. Quel est le plus rapide ?

4. Écrire un algorithme pour le tri de piles utilisant le principe du tri par sélection. La pile est la seule structure de données autorisée dans cet exercice (vous n'avez le droit ni aux files, ni aux listes)

5. Reprenez le tri fusion vu en cours. Générer aléatoirement des listes en variant leurs tailles comme pour la première question et comparer les temps mis entre ces deux algorithmes mais également en les comparant aux tris de la première question.

6. Le *tri par insertion* est généralement plus intéressant que le tri fusion lorsque la portion de liste à trier est petite. Écrire une version améliorée du *tri fusion* qui utilise le tri par insertion lorsque la traille de la liste ne dépasse pas une taille définie par défaut..

7. Le tri mixte consiste à diviser une liste en k sous listes, à trier chacune des listes séparèment en utilisant par exemple le *tri insertion* et enfin fusionner les listes obtenues. Implémenter ce tri. Trouver la meilleure valeur de k permettant d'obtenir la meilleure complexité.

suite

8. Réalisez avec l'algorithme de votre choix une fonction `trie_chaine` de type $string \rightarrow string$ qui construit la chaîne de caractères obtenues en triant les caractères de la chaîne passée en paramètre.

9. Nous souhaitons améliorer le tri insertion de la façon suivante. Lors de l'insertion d'un élément, la recherche de sa position est faite par une recherche dichotomique. Implémenter cette amélioration.

10. Le tri "Shaker" est une variante du tri à bulles. Son principe est similaire à celui du tri à bulles, excepté qu'il change de direction à chaque passe. Ce tri permet d'envoyer les plus grands éléments vers la fin mais également aux plus petits éléments vers le début. Écrire cet algorithme.

11. Implémenter le tri rapide puis générer aléatoirement des listes et comparer le au tri fusion.

12. Tester différentes versions du choix du pivot.

13. Écrire la fonction `partition` du tri rapide en récursive.

Tri & Classes

1. Écrire un programme qui permet de trier en ordre croissant une liste de points. Un `Point` est défini par sont abscisse et son ordonné. Un point est inférieur à un autre si sa distance à l'origine est plus petite. Vous devez implémentez au moins deux algorithmes de tri parmi ceux vu en cours.