TD d'algorithmique avancée n°5

1 ON A TOUS À Y GAGNER

Vous travaillez maintenant à la Poste. Votre rôle : le tri du courrier. Vous devez trier le courrier entrant selon les codes postaux qui y figurent, par ordre croissant. Le but est de pouvoir, ensuite, rechercher un courrier spécifique de façon beaucoup plus rapide (voir TD 4).

Dans la suite de l'exercice, vous possédez un tableau **Courrier** d'entiers, chaque entier représentant un code postal (on imagine que le problème Corse a enfin été réglé).

1. Écrire une première fonction qui trie le tableau selon la méthode suivante : le programme parcourt le tableau, et si deux éléments consécutifs ne sont pas à la bonne place, il les échange et recommence son parcours au début du tableau.

Quelle est la complexité (pire cas) de votre algorithme ?

C'est la fonction tri_super_lent du TD précédent.

2. Pouvez-vous faire une simple optimisation qui améliore votre algorithme ?

C'est la fonction tri_meilleur du TD précédent.

3. Maintenant, nous allons changer de stratégie... Écrivez une fonction qui recherche la position du plus grand élément du tableau. Votre algorithme devra rechercher le plus grand élément du tableau, le placer à la fin du tableau, et recommencer...

Quelle est la complexité (pire cas) de cet algorithme ?

C'est la fonction tri_selection du TD précédent.

4. Compte tenu que l'on travaille avec des codes postaux, combien d'éléments différents pouvez-vous avoir en tout dans votre tableau ?

Code postaux sur 5 chiffres, donc, on aura, dans le pire des cas, 100000 éléments différents.

Quelle remarque pouvez-vous faire si le nombre de lettre à trier est très très grand (de l'ordre du milliard) ?

On aura beaucoup de répétitions dans le tableau.

Proposez alors une meilleure méthode de tri, et calculez sa complexité.

On créé un tableau de 100000 cases qui dénombre en fait chaque élément du premier tableau (combien de fois le code 00000 apparaît, combien de fois le code 00001 apparaît, etc...). Puis à partir du dénombrement, on recréé le tableau original trié. Sa complexité est proportionnelle au nombre de cases du tableau original et au code postal maximal.

```
public static void tri_denombrement(int[] T)
{
    int code;
    int[] tous_les_code_postaux = new int[100000];

    //On denombre, dans tabtemp, les elements de T
    for(int cpt=0; cpt<T.length; cpt=cpt+1)
    {
        code = T[cpt];
        tous_les_code_postaux[code] =  tous_les_code_postaux[code]+1;
    }

    //Puis, on recopie les elements de tabtemp dans T, selon le nombre
    //de fois qu'il appraissaients;
    int compteurT=0;
    for(int code_postal = 0; code_postal<100000; code_postal=code_postal+1)
    {
        int nb_apparition_code = tous_les_code_postaux[code_postal];

        for(int cpt=0; cpt<nb_apparition_code; cpt=cpt+1)
        {
            //on recopie nb_apparition_code fois le code postal code_postal
            //dans le tableau T
            T[compteurT] = code_postal;
            compteurT=compteurT+1;
        }
    }
}
```

Sur quels types de données ce genre de tri ne pourrait-il pas fonctionner ?

Sur des double, on est foutus, car on ne peut pas associer un double à une case spécifique d'un tableau (tandis qu'un int, si)...

5. Pouvez-vous imaginer une méthode de tri plus efficace que celles proposées dans les trois premières questions ?

La tri fusion, voir TD précédent.