Lycée Thiers — Informatique
Année 2016/17 — PC*

TP : Feuille d'Exercices 8

Tri rapide en place - Calcul de Médiane

Exercice 1.

(1) Ecrire en `python` l'algorithme de tri rapide.
(2) Mesurer son temps d'exécution sur un tableau de nombres aléatoires, sur le modèle :

```
In[1]: from random import random
In[2]: T = [100*random() for k in range(1000)]
In[3]: %timeit triRapide(T)
```

Exercice 2. But : écrire une version "en place" du tri rapide.

Pour cela écrire une fonction `partition(T,debut, fin, pivot)` qui :

- prend en paramètre le tableau `T`, les indices du début (`debut`), de la fin (`fin`), et du pivot (`pivot` qui devra être entre `debut` et `fin`),
- réordonne les éléments de `T` entre les indices `debut` et `fin` de sorte que : les éléments inférieurs (respectivement supérieurs) au pivot lui soient placés avant (respectivement après).
- Renvoie l'indice du pivot. Pour celà :
- `e` $\leftarrow$ `T[pivot]`
- échanger les positions de `T[pivot]` et `T[fin]`
- `j` $\leftarrow$ `debut`.

Après chaque passage dans la boucle for suivante, `T[j]` sera le premier élément $\geqslant$ `e` parmi ceux parcourus.

- pour `i` variant de `debut` à `fin - 1` :
  Si `T[i] < e` alors :
  échanger `T[i]` et `T[j]`
  `j` $\leftarrow$ `j+1`
- échanger `T[fin]` et `T[j]`

Schématiquement : après chaque passage dans la boucle `T[j]` est le 1er élément supérieur au pivot.

elts $<$ pivot	`T[j]`$\geqslant$ pivot	elts $\geqslant$ pivot	`T[i]`	$\cdots\cdots$	**pivot**

à la fin de la boucle for :

elts $<$ pivot	`T[j]`$\geqslant$ pivot	elts $\geqslant$ pivot	**pivot**

à la fin :

elts $<$ pivot	**pivot**	elts $\geqslant$ pivot	`T[j]`$\geqslant$ pivot

Exercice 3.

(1) Ecrire l'algorithme de recherche de la médiane d'un tableau de nombres basé sur le tri rapide, vu en cours, sur le modèle :

```
recherche(T,s):
    Soit e=T.pop(len(T)//2)
    Soient T1, T2 obtenu par tri rapide avec pivot e.
    Si len(T1)==s : renvoyer e.
    Sinon, si len(T1)>s : renvoyer recherche(T1,s)
    Sinon (si len(T1)<s) : renvoyer recherche(T2,s-len(T1)-1)
```

(2) Ecrire une version "en place" du calcul de la médiane par tri rapide.

Exercice 4. *Calcul de la médiane en temps linéaire.*

L'algorithme décrit permet de déterminer la médiane, et plus généralement le k-ième élément d'un tableau, sans le trier intégralement. Il est linéaire dans le pire des cas.

Nous allons écrire une fonction `partile(L,i)` qui retourne l'élément d'indice i dans ce que serait le tableau L après avoir été trié.

(1) Ecrire une version du tri par insertion qui s'applique à un sous-tableau ; elle prendra trois arguments, une liste `L` et deux entiers `i` et `j` et triera (en place) son sous-tableau allant de l'indice `i` inclus à l'indice `j` exclu.

(2) Ecrire une fonction qui prend en paramètre un tableau et des indices `i`, `j` délimitant 5 éléments et qui retourne la médiane du sous-tableau de ces 5 éléments.

(3) Ecrire une fonction qui prend en argument un tableau quelconque, le divise en groupes de cinq éléments (ou moins) et retourne la liste des médianes de chacun de ces groupes.

(4) Modifier cette fonction pour qu'elle s'appelle récursivement sur le tableau des médianes construit.

(5) Ecrire une fonction qui effectue une partition en 3 sous-listes (inférieur/médiane des médianes/supérieur) du tableau de départ avec pour pivot la médiane des médianes obtenue à l'aide de la fonction précédente.

(6) Pour trouver le k-ième élément où faut-il le chercher en fonction de la taille des deux sous-tableaux donnés par la partition. Programmer l'appel récursif correspondant dans une fonction `partile(L,k)`.