

Informatique en CPGE (2017-2018) TD 5 : algorithmes de tri

Exercice 1 : temps d'exécution

Le nombre d'appels récursifs est limité. Nous augmentons cette limite en écrivant les instructions suivantes :

```
import sys
sys.setrecursionlimit(10000)
```

1. Construire trois listes de 3000 entiers : **liste1** contient les entiers de 1 à 3000 rangés dans l'ordre croissant ; **liste2** contient les entiers de 3000 à 1 rangés dans l'ordre décroissant ; **liste3** contient 3000 entiers choisis de manière aléatoire avec la fonction **randint** parmi les entiers de 1 à 3000 bornes comprises.
2. Mesurer le temps d'exécution des trois algorithmes de tri (tri insertion, tri rapide, tri fusion) pour trier chacune des trois listes précédentes.

On utilisera la fonction **time** du module **time**.

Par exemple :

```
from time import time
start=time()
tri_insertion(liste1)
print(time()-start)
```

3. Mesurer sur les mêmes listes le temps d'exécution des algorithmes de tris utilisés par la fonction **sorted** et par la méthode **sort()**.
La syntaxe est `sorted(liste)` et `liste.sort()`.
4. Mesurer maintenant le temps d'exécution de la fonction **sorted** et de la méthode **sort()** sur une liste de 100000 entiers choisis de manière aléatoire avec la fonction **randint** parmi les entiers de 1 à 10000 bornes comprises.

Exercice 2 : ordre lexicographique

L'objectif est d'écrire un programme qui trie une liste de mots et les range dans l'ordre lexicographique (ordre des dictionnaires).

1. Ecrire la définition de la variable `alphabet` :
`alphabet = "AaàBbCcDdEeèèFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuùVvWwXxYyZz"`
2. Ecrire une fonction **ordre_alphabetique(c1, c2)** qui prend en arguments deux caractères alphabétiques et renvoie `-1` si `c1` est avant `c2`, `1` si `c2` est avant `c1` et `0` si `c1 = c2`. On pourra utiliser la méthode **index** qui renvoie l'indice d'un élément dans une chaîne de caractère.
3. Ecrire une fonction **ordre_lexicographique(m1, m2)** qui prend en arguments deux mots et renvoie `-1` si "`m1 < m2`" pour l'ordre lexicographique, `0` si "`m1 = m2`" et `1` si "`m1 > m2`". On utilisera la fonction **ordre_alphabetique**.
4. Ecrire une fonction **tri_lexicographique(liste)** qui prend en argument une liste de mots et renvoie la liste triée. On utilisera la fonction **ordre_lexicographique** avec l'algorithme du **tri par insertion**.