

Exercice 1 :

- 1) Soit d un dictionnaire. Si la clé k n'est pas dans le dictionnaire, l'instruction $d[k] = 5$ provoque-t-elle une erreur ?
- 2) Soit d un dictionnaire. L'instruction $d[k] = [10,5]$ provoque-t-elle une erreur ?
- 3) Soit d un dictionnaire. L'instruction $d[[10,5]] = 5$ provoque-t-elle une erreur ?
- 3) Créer un programme qui affiche la liste des clés
- 4) Créer un programme qui affiche la liste des valeurs
- 5) Créer un programme qui affiche la liste des paires (clé,valeur)

Exercice 3 :

On considère les deux dictionnaires suivants :

```
dico1={"paris":75,"La Rochelle":17}
dico2={"Rennes":35,"Orléans":45}
```

Ecrire un programme permettant de concaténer en seul dictionnaire les clés et valeurs des 2 dictionnaires précédents

Exercice 4 :

On considère un dictionnaire donnant la moyenne de quelques étudiants.

```
dico={"etudiant1":15,"etudiant2":9,"etudiant3":12,"etudiant4":5}
```

Partitionner ce dictionnaire en deux sous dictionnaires : un dictionnaire des élèves ayant la moyenne et un dictionnaire des élèves n'ayant pas la moyenne

Exercice 5 :

Écrire une fonction en Python qui prend en paramètre une liste de nombres entiers et qui renvoie un dictionnaire dont les clés sont les entiers de la liste et dont les valeurs sont 'pair' ou 'impair' selon la parité du nombre.

Exercice 6 :

Écrire une fonction *comptage* qui prend en paramètre un mot et renvoie un dictionnaire qui pour chaque caractère du mot associe le nombre d'occurrence de chaque lettre. On suppose le texte écrit en lettres capitales non accentuées.

Exercice 7 :

Écrire une fonction *comptage2* qui prend en paramètre un mot et renvoie un dictionnaire qui pour chaque caractère du mot associe le nombre d'occurrence de chaque lettre. On suppose le texte écrit en lettres capitales non accentuées. On précisera que les autres caractères ., ;! ? sont exclus.

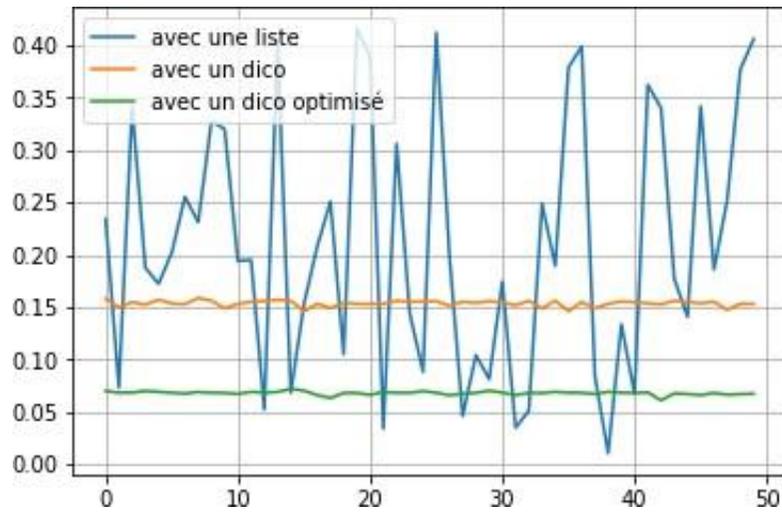
Exercice 8 :

Ecrire une fonction qui prend en argument un mot (type `str`) et qui renvoie un dictionnaire ayant pour clés les caractères du mot et pour valeurs les distances (les plus courtes si des caractères apparaissent plusieurs fois) entre chaque caractère du mot et le dernier caractère de ce mot.

Exercice 9 :

La question est de mesurer l'intérêt d'une représentation par un dictionnaire plutôt que par une liste de listes. Nous allons vérifier que le temps de recherche d'une valeur dans un dictionnaire est le même quel que soit le nombre d'éléments de cette structure (ce qui n'est pas le cas avec une liste).

- 1) Construire une liste, appelée *liste*, dont les éléments sont de la forme $[i, i]$ pour i allant de 0 à $10^6 - 1$. Mélanger cette liste avec la fonction *shuffle* du module *random*. Il suffit d'importer la fonction et d'écrire *shuffle(liste)*. Créer ensuite le dictionnaire correspondant qui contient les couples *cle: valeur* de la forme $i: i$ à l'aide de la fonction *dict*.
- 2) Ecrire une fonction *recherche1* qui prend en paramètres une liste de listes et une variable k et renvoie le 2^e élément de la sous liste dont le premier élément a la valeur de k .
- 3) Ecrire une fonction *recherche2* qui prend en paramètres un dictionnaire et une variable k et renvoie la valeur correspondant à la clé k .
- 4) Tester les fonctions de recherche sur la liste et le dictionnaire en utilisant pour le paramètre k des valeurs allant de 0 à 49. Pour les mesures, utiliser la fonction *time* du module *time* ainsi que le module *matplotlib*



On remarque bien un temps de recherche constant. Le mot clé *in* utilisé pour un dictionnaire fait référence un algorithme appelé table hachage qui utilise une fonction f de hachage $clé \Rightarrow nouvelle\ valeur$. Avec cette fonction f de hachage, la recherche $f(clé)$ se fait à coût constant.

Ce qui n'est pas le cas avec une liste qui présente un temps de calcul qui est fonction de la position de la clé dans la liste.