Lycée Vaugelas MP 2022 - 2023
TP d'informatique n° 2 du 12 septembre 2023 **Dictionnaires**

On rappelle les éléments du cours de première année :

On accède à un élément d'un tableau par son indice (entier). Un dictionnaire (de type `dict`) est une généralisation de ce concept qui permet d'accéder à une valeur à partir, non pas d'un entier, mais d'une **clé** qui peut être un entier, une chaîne de caractère, ...
Un dictionnaire est donc un ensemble de couple **clé/valeur** et il est implémenté de manière à ce que l'accès à n'importe quelle valeur se fasse en temps constant, peu importe la taille du dictionnaire.

`d={cle1:valeur1, cle2:valeur2}`

Les clés doivent être uniques. On ajoute un nouveau couple `cle3/valeur3` de la manière suivante : `d[cle3]=valeur3` ; on affiche la valeur associée à `cle2` avec `d[cle2]`.
Un dictionnaire implémente un certain nombre de méthodes :

- `d.keys()` renvoie un itérateur sur l'ensemble des clés du dictionnaire `d` (attention, l'ordre des clés n'est pas forcément préservé suivant la version de python utilisée)
- `d.values()` renvoie un itérateur sur l'ensemble des valeurs du dictionnaire `d`
- `d.items()` renvoie un itérateur sur l'ensemble des couples clé/valeur du dictionnaire `d`

2 manières d'afficher l'ensemble des couples clé/valeur du dictionnaire `d`	`for cle in d :`
	`  print(cle, d[cle])`
	
	`for cle, valeur in d.items() :`
	`  print(cle, valeur)`

Exercice 1 (Comptage des caractères d'une chaîne)

1. Écrire une fonction `comptage` qui prend en argument une chaîne (type `str`) et qui renvoie un dictionnaire contenant le nombre d'occurrences (valeurs) de chaque caractère (clés) de la chaîne.
2. Testez la fonction avec une courte phrase.
3. La fonction est-elle capable de compter le nombre d'occurrences des éléments d'une liste ou d'un tableau `numpy` ? La modifier si ce n'est pas le cas.

Exercice 2 (Trinômes de colle)

On donne un dictionnaire d (type `list`) contenant les trinômes de la classe dans un ordre quelconque : `d = { 4:('Durand', 'Martin', 'Dupont'), 2:('Petit', 'Dubois', 'Dumoulin'), 10:('Richard', 'Thomas', 'Roux'), ...}`.

1. Écrire une fonction `eleves` qui reçoit le dictionnaire d en argument et qui renvoie un dictionnaire dont les clés sont les noms et les valeurs les numéros de trinômes.
2. Écrire une fonction `classe` qui reçoit le dictionnaire d en argument et qui renvoie la liste des couples (nom, trinôme) triée par ordre alphabétique des noms.

Exercice 3 (Calcul de score au jeu de Scrabble)

Le jeu de Scrabble attribue des points à chacune des lettres de l'alphabet : 1 point pour A, E, I, L, N, O, R, S, T, U ; 2 points pour D, G, M ; 3 points pour B, C, P ; 4 points pour F, H, V ; 8 points pour J, Q ; 10 points pour K, W, X, Y, Z.

1. Définir un dictionnaire `points` dont les clés sont les lettres et le valeurs les points.
2. Écrire une fonction `score` qui prend en argument une chaîne contenant un mot (on accepte minuscule et majuscules mais pas les lettres accentuées) et renvoie le score associé au mot. La méthode `upper()` du type `str` peut être utilisée.

Exercice 4 (Polynôme dérivé)

On représente un polynôme à une indéterminée par un dictionnaire dont les clés sont les degrés des mônômes non nuls et les valeurs les coefficients correspondants. Par exemple $P = 8 + 3X - 7X^4$ est représenté par `{0:8,1:3,4:-7}`. Le polynôme nul est conventionnellement représenté par $\{0 : 0\}$.

1. Écrire une fonction `derive` qui reçoit en paramètre un polynôme et retourne le polynôme dérivé.
2. Déterminer les complexités spatiale et temporelle de la fonction `derive`.

Exercice 5 (Comparaison de deux tableaux)

Afin de déterminer si deux tableaux contiennent exactement les mêmes éléments (mais en ordre éventuellement différents), on crée pour l'un des tableaux un dictionnaire des occurrences : les clés sont les éléments du tableau et les valeurs les nombres d'occurrences de ces éléments. Il suffit alors de parcourir le second tableau et d'utiliser judicieusement le dictionnaire pour conclure.

1. Écrire une fonction `occurrence` qui reçoit en paramètre un tableau (type `list`) et renvoie le dictionnaire des occurrences de ce tableau.
2. Écrire une fonction `compare` qui reçoit deux tableaux et renvoie un booléen indiquant l'égalité des tableaux (mêmes éléments, pas forcément dans le même ordre, mais avec les mêmes nombres d'occurrences).
3. Évaluer la complexité temporelle de la fonction `compare`.
4. Une autre méthode de comparaison consiste à trier les deux tableaux puis à réaliser un seul parcours linéaire. Laquelle des deux méthodes fournit la meilleure complexité temporelle ?