Dictionnaires

Créer et modifier un dictionnaire

1. Tester le code suivant et comprendre à quoi correspondent les résultats affichés :

```
Date = {"jour" : 1, "mois" : "décembre", "année" : 2022}
print(Date["jour"])
print(Date["année"])
```

puis écrire à la suite et tester le code suivant :

```
Date["année"] = 2021
Date["heure"] = "13:00"
print(Date["année"])
print(Date)
```

2. Toujours à la suite de ce qui précède, tester le code suivant et comprendre à quoi correspondent les résultats affichés :

```
print(Date.keys())
print(Date.values())
```

On dit que **Date** est un *dictionnaire*. On peut retenir que cet objet enregistre des relations de la forme :

```
"jour"   ⟶  1
"mois"   ⟶  "décembre"
"année"  ⟶  2021
"heure"  ⟶  "13:01"
```

Les éléments de la colonne de gauche sont appelés des *clés* et ceux de la colonne de droite sont les *valeurs* associés à ces clés. Les valeurs peuvent être de n'importe quel type (ici on a

des chaines de caractères et des nombres), les clés peuvent être de (presque) n'importe quel type mais, dans la suite de ce sujet, ce seront des chaines de caractères.

3. En procédant comme dans les codes qui précèdent, créer un dictionnaire **Adresse** qui contient les associations **"numéro"** → 20, **"ville"** → **"Reims"**, **"rue"** → **"Roosevelt"**. Vérifier avec **print(Adresse)** que le dictionnaire contient bien les information demandées.

4. À partir du dictionnaire précédent, et sans le réécrire entièrement, modifier l'information **"numéro"** en lui donnant la valeur 10 et ajouter l'association **"cp"** → 51100. Vérifier avec **print(Adresse)**.

Parcours d'un dictionnaire

5. Tester le code suivant (à écrire à la suite du code des questions précédentes, on réutilise le dictionnaire **Adresse**). Comprendre à quoi correspondent les résultats affichés :

```
for k in Adresse.keys():
    print(k)
    print(Adresse[k])
```

6. On considère un dictionnaire **D** dont toutes les *valeurs* sont des nombres. Écrire une fonction **somme(D)** qui construit et renvoie la somme de toutes les valeurs d'un tel dictionnaire **D**. Vérifier que les instructions :

```
D = {"a":4, "b":1, "c":2}
print(somme(D))
```

affichent bien le résultat attendu.

7. On considère toujours un dictionnaire **D** dont toutes les *valeurs* sont des nombres positifs. Écrire une fonction **maximum(D)** qui détermine et renvoie la plus grande valeur apparaissant dans ce dictionnaire **D**. Vérifier que les instructions :

```
D = {"a":4, "b":1, "c":2}
print(maximum(D))
```

affichent bien le résultat attendu.

Construction d'un dictionnaire

8. On considère une liste **L** contenant des mots (chaines de caractères) et on souhaite construire un dictionnaire **D** qui répertorie, pour chaque mot, le nombre de fois où il apparait dans **L**. Ainsi pour les listes :

```
L1 = ["maison","avion","maison"]
L2 = ["a","a","b","c","b"]
```

on obtiendra respectivement les dictionnaires :

```
D1 = {"maison":2, "avion":1}
D2 = {"a":2, "b":2, "c":1}
```

On donne le code suivant (inutile de recopier les commentaires) :

```
def repertorier(L):
    D = {} # Dictionnaire vide pour démarrer
    for i in range(0,len(L)):
        s = L[i] # s est le mot d'indice i dans L
        # On augmente le nombre de fois où on a vu s
        D[s] = D[s]+1
    return D

L1 = ["maison","avion","maison"]
L2 = ["a","a","b","c","b"]

print(repertorier(L1))
print(repertorier(L2))
```

Tester ce code et constater qu'il ne fonctionne pas.

9. Modifier la fonction **`repertorier`** pour qu'elle fonctionne correctement. On utilisera l'instruction **`if s in D.keys()`** qui teste si **`s`** apparait déjà parmi les clés de **`D`**. Vérifier alors que les commandes :

```
print(repertorier(L1))
print(repertorier(L2))
```

affichent bien les dictionnaires attendus.

Corrections

Q 3.

```
Adresse = {"numéro":20,"ville":"Reims","rue":"Roosevelt"}
print(Adresse)
```

```
'numéro': 20, 'ville': 'Reims', 'rue': 'Roosevelt'
```

Q 4.

```
Adresse["numéro"] = 10
Adresse["cp"] = 51100
print(Adresse)
```

```
'numéro': 10, 'ville': 'Reims', 'rue': 'Roosevelt', 'cp': 51100
```

Q 6.

```
def somme(D):
    s = 0
    for k in D.keys():
        s = s+D[k]
    return s

D = {"a":4, "b":1, "c":2}
print(somme(D))
```

```
7
```

Q 7.

```
def maximum(D):
    m = 0
    for k in D.keys():
        if D[k]>m:
            m = D[k]
    return m

D = {"a":4, "b":1, "c":2}
print(maximum(D))
```

```
4
```

Par hypothèse, tous les éléments du dictionnaire sont positifs donc en partant de $m = 0$ et en changeant sa valeur à chaque fois qu'une valeur plus grande est rencontrée, à la fin de la boucle m contient la plus grande valeur rencontrée dans le dictionnaire.

Q 9.

```
def repertorier(L):
    D = {}
    for i in range(0,len(L)):
        s = L[i]
        if s in D.keys():
            D[s] = D[s]+1
        else:
            D[s] = 1
    return D

L1 = ["maison","avion","maison"]
L2 = ["a","a","b","c","b"]

print(repertorier(L1))
print(repertorier(L2))
```

```
'maison': 2, 'avion': 1 'a': 2, 'b': 2, 'c': 1
```