DICTIONNAIRES

I) Présentation des dictionnaires

Un dictionnaire est un ensemble de couples de deux objets : le premier est la clé (key en anglais), le second est la valeur (value en anglais).

La clé est **unique**.

Les clés et les valeurs peuvent être de différents types : chaîne de caractères, entiers, flottants, etc.

Exemple :

Voici un tableau donnant pour des amis leur prénom et leur âge :

Prénom	Amin	Samir	Camille
Âge	22	25	17

- La création du dictionnaire sera codée comme suit :

```
dicoAmis={"Amin" : 22 , "Samir" : 25 , "Camille": 17}
```

On peut également créer un dictionnaire autrement :

```
dicoAmis=dict(Amin = 22 , Samir = 25 , Camille = 17)

dicoAmis=dict([("Amin" , 22), ("Samir" , 25), ("Camille", 17)])
```

Mais aussi à partir de deux listes avec la fonction **zip** :

```
prenom=["Amin","Samir","Camille"]
age=[22,25,17]
dicoAmis=dict(zip(prenom,age))
```

- L'accès à l'information se fait par la clé.

Exemple :

```
dicoAmis={"Amin" : 22 , "Samir" : 25 , "Camille": 17}
print("Âge de mon ami Amin :", dicoAmis["Amin"] ) #première méthode
print("Âge de mon ami Camille :", dicoAmis.get("Camille"))#seconde méthode
Affichage : Âge de mon ami Amin :  22
            Âge de mon ami Camille : 17
```

- **dictionnaire[key]** : renvoie la **valeur** associée à la clé key dans le dictionnaire.
- **dictionnaire.get(key)** : retourne la **valeur** associée à la clé key si celle-ci existe .

- L'ajout d'un couple clé/valeur se fait de 2 façons :

Exemple :

```
dicoAmis["Jammy"]=30 #première méthode

nouvelAmi = {"Jammy": 30}#seconde méthode
dicoAmis.update(nouvelAmi)
```

- La suppression d'une valeur se fait de 2 façons :

Exemple :

```
del dicoAmis["Camille"] #première méthode
dicoAmis.pop("Camille")# seconde méthode
```

Exercice n°1

Titre	Auteur
Iliade	Homère
Odyssée	Homère
Enéide	Virgile
Mahabharata	Vyasa

- Créer un fichier « Dico.py ».
- Créer le dictionnaire « epopee » correspondant au tableau ci-contre.
- Peut-on créer le dictionnaire appelé epopeeInversee dont les auteurs des œuvres seront les clés et les titres seront les valeurs ?
- On souhaite compléter la bibliothèque par l'achat du Ramayana. La tradition attribue ce grand poème épique indien au poète nommé Valmiki. Compléter le dictionnaire epopee suite à cet achat.
- On retire ensuite l'Enéide de Virgile de la bibliothèque. Modifier le dictionnaire epopee.
- Tester ce programme en affichant le dictionnaire.

II) Utilisation d'un dictionnaire

a. Test d'appartenance en utilisant l'instruction in :

Exemple de fonction permettant de savoir si un nom est dans le dictionnaire :

```
def appartenanceAmi(dictionnaire:dict,nom:str)->str:
    if nom in dictionnaire:
        return nom + " est un de mes amis"
    else :
        return nom + " n'est pas un de mes amis"

print(appartenanceAmi(dicoAmis,"Camille"))

Affichage : Camille est un de mes amis
```

b. Accéder à l'ensemble des clés grâce à la méthode keys() :

dictionnaire.keys() : retourne l'ensemble des **clés** du dictionnaire.

Exemple de fonction affichant tous les amis :

```
def ensembleAmis(dictionnaire:dict)->None:
      for nom in dictionnaire.keys() :
            print(nom + " est un de mes amis.")
      return

ensembleAmis(dicoAmis)

Affichage :       Samir  est un de mes amis.
                  Camille  est un de mes amis.
                  Amin  est un de mes amis.
```

Remarque : Comme les dictionnaires n'ont pas de structure ordonnée, les clés ne s'affichent pas forcément dans l'ordre dans lequel on les a entrées.

c. Accéder à l'ensemble des valeurs grâce à la méthode values() :

dictionnaire.values() : retourne l'ensemble des **valeurs** du dictionnaire.

Remarque : tout comme dictionnaire.keys(), dictionnaire.values() renvoie une "vue" des valeurs, ce qui n'est pas une liste des valeurs.

- list(dictionnaire.values()) permet de transformer cet ensemble en liste.

Exemple de fonction affichant tous mes amis :

```
def ensembleAgeAmis(dictionnaire:dict)->None:
      for valeur in dictionnaire.values() :
            print("J'ai un ami ayant ",valeur,"ans.")
      return

ensembleAgeAmis(dicoAmis)

Affichage :       J'ai un ami ayant  22 ans.
                  J'ai un ami ayant  17 ans.
                  J'ai un ami ayant  25 ans.
```

d. Accéder à l'ensemble des couples (clé,valeur) grâce à la méthode items() :

dictionnaire.items() : retourne l'ensemble du dictionnaire sous forme d'une liste contenant tous les couples **(clé,valeur)** du dictionnaire.

Remarque : chaque couple (clé,valeur) est un **tuple**.

Exemple de fonction affichant l'ensemble de mes amis avec leur âge :

```
def ensembleElementsDictionnaire(dictionnaire:dict)->None:
      for cle,valeur in dictionnaire.items() :
            print("J'ai un ami "+ cle + " ayant "+ str(valeur) + " ans.")
      return

ensembleElementsDictionnaire(dicoAmis)

Affichage :  J'ai un ami Samir ayant 25 ans.
             J'ai un ami Amin ayant 22 ans.
             J'ai un ami Camille ayant 17 ans.
```

Attention !
dictionnaire.items() retourne une **vue** du dictionnaire : toute modification apportée à cette vue est répercutée dans le dictionnaire. Réciproquement, toute modification du dictionnaire est répercutée dans cette liste.

e. Copier un dictionnaire :

Comme pour les listes, on ne peut pas **copier** un dictionnaire en faisant dict1=dict2 :

Exemple :

```
dicoAmis={"Amin" : 22 , "Samir" : 25 , "Camille": 17}
dico2=dicoAmis
dico2["Jammy"]=30
print(dicoAmis)
print(dico2)
```

On obtient comme affichages successifs :

```
{'Camille': 17, 'Jammy': 30, 'Samir': 25, 'Amin': 22}
{'Camille': 17, 'Jammy': 30, 'Samir': 25, 'Amin': 22}
```

On aurait pu penser que seule la variable dico2 aurait été modifiée. Or, l'affichage montre que dicoAmis a aussi été affecté par cette modification. Ainsi, dico2 n'est pas une copie indépendante de dicoAmis. L'instruction dico2=dicoAmis permet seulement de lier ces deux variables à la même référence. Ainsi, comme les deux dictionnaires "pointent" vers la même référence, toute modification de cette donnée affecte chacun des deux dictionnaires.

- **dictionnaire.copy()** : permet de copier une vraie copie.

Exemple :

```
dicoAmis={"Amin" : 22 , "Samir" : 25 , "Camille": 17}
dico2=dicoAmis.copy()
dico2["Jammy"]=30
print(dicoAmis)
print(dico2)
```

On obtient comme affichages successifs :

```
{'Camille': 17, 'Samir': 25, 'Amin': 22}
{'Camille': 17, 'Jammy': 30, 'Samir': 25, 'Amin': 22}
```

dico2 est une copie indépendante de dicoAmis : la modification ne touche qu'un seul dictionnaire.

Dans l'exercice suivant, on s'intéresse à la population des pays membres de l'Union Européenne au 1er janvier 2018. Les données sont issues de l'organisme européen de statistique Eurostat. source :

Exercice n°2

- Mettre en commentaire le code précédent.
- Ajouter dans le même répertoire le fichier « population_pays_ue.csv ».
- Copier les 2 fonctions ci-dessous dans votre programme (ainsi que les appels de test) et exécuter le code. Le but de ces 2 fonctions est de créer 2 listes à partir du fichier, une dans laquelle vont se trouver les pays de l'union européenne, l'autre les populations correspondantes.

```
import csv
def creationPaysFromCSV(nomDeFichier:str)->list:
    liste=[]
    with open(nomDeFichier, newline='',mode='r',encoding='utf-8') as fichierCsv : #ouverture
        lecture = csv.reader(fichierCsv,dialect=csv.excel_tab)
          #création d'un lecteur de fichier
        for ligne in lecture :
          #pour chaque ligne lue, on a le nom et la population du pays
            val=ligne[0]#ligne[0] correspond au nom du pays
            liste.append(val) #ajout de la valeur dans la liste
    fichierCsv.close()#fermeture du fichier même si cela n'est pas nécessaire
    return liste

def creationPopulationFromCSV(nomDeFichier:str) -> list:
    liste=[]
    with open(nomDeFichier, newline='',mode='r') as fichierCsv :
            lecture = csv.reader(fichierCsv,dialect=csv.excel_tab)
            for ligne in lecture :
                val=int(ligne[1])#ligne[1] correspond à la population du pays
                liste.append(val)
    fichierCsv.close()
    return liste

pays=creationPaysFromCSV(nomDeFichier='population_pays_ue.csv') #la liste des noms de pays es
print("pays=",pays)
population=creationPopulationFromCSV(nomDeFichier='population_pays_ue.csv') #la liste de la p
print("population=",population)
```

- Créer à partir des listes pays et population, le dictionnaire, appelé dico, dont les noms de pays seront les clés.
- Suite au Brexit, créer une copie, nommée dico27, du dictionnaire dico puis supprimer le Royaume-Uni de ce dictionnaire dico27.
- Écrire la fonction populationDe qui reçoit en paramètre un nom (de type chaîne de caractères) et un dictionnaire d (de type dict) et qui retourne soit la population du pays de l'UE ayant ce nom, soit le texte "Saisir le nom d'un pays membre de l'UE".
- Tester votre fonction en affichant les populations de la France, puis de l'Italie.

- Écrire la fonction paysPopulationDe qui reçoit en paramètre un nombre n (de type entier) et un dictionnaire d (de type dict) et qui retourne soit le nom du pays de l'UE ayant cette population, soit le texte "aucun pays de l'UE n'est peuplé de tant d'habitants". Penser à utiliser la méthode *ìtems*(), voire les méthodes *keys*() et *values*().
- Tester avec les valeurs 10741165 puis 11125478.
- Écrire la fonction listePaysUEOrdonnee qui reçoit en paramètre un dictionnaire d (de type dict) et qui retourne la liste des noms des pays de l'UE classés par ordre croissant,.
- Tester cette fonction.

Exercice n°3

La liste des données Exif (Exchangeable image file format) intégrées par les appareils photos numériques regroupe de nombreuses informations parmi lesquelles :

- La date et l'heure à laquelle la photo a été prise.
- Les paramètres de prise de vue (vitesse, diaphragme, sensibilité ISO, mémorisation d'exposition, etc.)
- La géolocalisation de l'image (pour les appareils équipés de la fonction GPS).
- L'identification du type de boîtier et d'objectif (y compris les numéros de séries)

On peut accéder à ces données par l'intermédiaire d'un dictionnaire, les clefs étant ici des numéros.

Pour créer le dictionnaire, on utilise la bibliothèque PIL (Python Imaging Library) comme suit :

```
import PIL.Image
img = PIL.Image.open('nom_de_la_photo')
exif_data = img._getexif()  #exif-data représente le dictionnaire
#img.show() permet d'afficher l'image
```

- On demande d'afficher pour les 4 images données dans le fichier zip les données suivantes :
largeur (en pixels)→ clef : 40962
hauteur (en pixels) → clef : 40963
marque de l'appareil → clef : 271
type de l'appareil → clef : 272
type de logiciel → clef : 305
ISO → clef : 34855
date et heure → clef : 36867

Certaines images sont géolocalisées (latitude et longitude). Pour cela, on trouve les données grâce à la clef 34853. Toutefois, on obtient alors un autre dictionnaire :
N ou S (Nord ou Sud)→ clef : 1
Latitude→ clef : 2 → on récupère 3 Tuples (degré, minute, seconde) avec la valeur et un diviseur
W ou E (Ouest ou Est) → clef : 3
Longitude → clef : 4→ on récupère 3 Tuples (degré, minute, seconde) avec la valeur et un diviseur

- Afficher pour « image4.jpg » sa géolocalisation. Afficher ensuite sous la forme:
Latitude : XX° YY' ZZ'' N (ou S)
Longitude : XX° YY' ZZ'' E (ou W)
- En utilisant Google Maps, trouver où la photo a été prise.

- En utilisant le code ci-dessous, lancer une navigateur qui affichera l'emplacement où « image2.jpg » a été prise.

```
import webbrowser
webbrowser.open('http://www.google.com')
```