Les dictionnaires et recherche en table - TP

▷ **Exercice 1**

Comme pour les tableaux de données, le type dictionnaire dispose d'un grande nombre de méthodes et fonctions. Plus particulièrement :

- Le constructeur ***dict()*** fabrique un dictionnaire directement à partir d'une ***liste de paires clé-valeur*** stockées sous la forme de tuples.

```
>>> dict([('sape', 4139), ('guido', 4127), ('jack', 4098)])
{'sape': 4139, 'guido': 4127, 'jack': 4098}
```

- ***dic.clear()*** supprime tous les éléments du dictionnaire ***dic*** et renvoie le dictionnaire actualisé
- ***dic.copy()*** renvoie une copie indépendante du dictionnaire ***dic***.
- ***dic.get(key[, default])*** renvoie la valeur de ***key*** si ***key*** est dans le dictionnaire ***dic***, sinon default. Si default n'est pas donné, il vaut None par défaut, de manière à ce que cette méthode ne lève jamais KeyError.

Le but de cet exercice est de recoder ces procédés sous la forme de quatre fonctions nommées :

dict_, clear, copy et ***get***

On veillera à spécifier chaque fonction écrite et à choisir des tests pertinents.

▷ **Exercice 2 - Recherche en table**

On peut pour un même ensemble de données, utiliser des types de données différents : les tableaux (listes) et les dictionnaires, par exemple. On va dans cet exercice, comparer l'efficacité de recherche dans ces deux types de données. Voici l'exemple d'un tableau annuaire qui stocke des prénoms et le numéro de télephone associé :

```
annuairetableau=[('sophie', '0622602799'), ('cecile', '058178235'),...]
```

1. Programmer une fonction ***recherche1*** de paramètres un prenom ***p*** et une liste de couple prenom, numero ***t*** et qui renvoie le numero de téléphone s'il est présent et None sinon. (on utilisera une boucle)
2. En utilisant la fonction ***dict_*** vue dans l'exercice 1, générer un dictionnaire contenant les associations prenom, numero.
3. Ecrire une fonction ***recherche2*** de paramètres un prenom ***p*** et un dictionnaire ***d*** contenant les associations prenom, numero et qui renvoie le numero de téléphone s'il est présent et None sinon.
4. A l'aide du module ***time*** et des instructions suivantes, comparer et commenter les durées d'éxécution des deux fonctions.

```
import time
ti = time.perf_counter()
recherche1('Pauline',annuairetableau)
ts = time.perf_counter()
print("Recherche1 du prénom :", 'Pauline', ", temps de calcul:", ts - ti)
```

▷ **Exercice 3 - Gestion de bibliothèque** ⋆⋆

Dans cet exercice, nous illustrons les compréhensions et les recherches en tables en travaillant sur une base de données de livres empruntables dans une bibliothèque. La base de données que nous allons utiliser en exemple est la suivante :

```
stock={
9788806222093:{'auteur':'Victor Hugo','titre':"Les misérables",'nombre':5},
9780671201821:{'auteur':'Robert Merle','titre':'Un animal doué de raison','nombre':6},
9783518399989:{'auteur':'Alain Fournier','titre':'Le grand Meaulnes','nombre':1},
9782072864537:{'auteur':'Victor Hugo','titre':'Notre dame de Paris','nombre':4},
9781976166471:{'auteur':'Victor Hugo','titre':'Notre dame de Paris','nombre':3},
9782070518425: {'auteur':'J.K. Rowling','titre':"Harry Potter à l'école des sorciers",'nombre':3},
9782070643059: {'auteur':'J.K. Rowling','titre':"Harry Potter et la coupe de feu",'nombre':0}
}
```

La base va nous permettre de rechercher rapidement un livre à partir de son numéro ISBN. On obtient alors l'auteur du livre, le titre de l'ouvrage ainsi que le nombre d'exemplaire(s) empruntable(s) en stock.

1. Ecrire une fonction ***auteurs*** qui prend en paramètre une base de livres ***b*** et qui retourne la liste des noms des auteurs de cette base.
   On donnera dans cette question d'abord une définition sans utiliser de compréhension, puis une définition avec compréhension.
2. Ecrire une fonction ***livresempruntables*** qui prend en paramètre une base de livres ***b*** et qui retourne la liste des titres empruntables
3. Ecrire une fonction ***titres_auteurs*** qui prend en paramètre une base de livres ***b*** et un nom d'auteur ***n*** et qui retourne un dictionnaire comportant les associations cle/valeur avec le titre comme clef et nombre comme valeur.
4. La bibliothèque reçoit une nouveau lot de livres donné par la table suivante :

Numéro ISBN	Auteur	Titre	Nombre
9782070643059	J.K. Rowling	Harry Potter et la coupe de feu	10
9782070556854	J.K. Rowling	Harry Potter et l'Ordre du phénix	7
9782070615360	J.K. Rowling	Harry Potter et les Reliques de la Mort	15

   Créer un dictionnaire nommé ***nouveau]*** correspondant à la table précédente. Ecrire une fonction ***fusion*** qui prend en paramètre deux bases de livres ***b1*** et ***b2*** et qui retourne un dictionnaire correspondant au nouveau stock de la bibliothèque.

Pour aller plus loin ...

▷ **Exercice 4** - Traduction ⋆⋆

Comme son nom l'évoque, une des fonctionnalités d'un dictionnaire est de s'en servir comme outil de traduction. Nous allons voir ici quelques manipulations simples de dictionnaires de langues.
Dans la suite, on prendra, en exemple, les dictionnaires anglais-français et français-allemand suivants :

```
#dictionnaire anglais vers français
danfr={
'one':'un','two':'deux','three':'trois','four':'quatre','five':'cinq','six':'six','seven':'sept'
}
#dictionnaire français vers allemand
dfral={
'un':'eins','deux':'zwei','trois':'drei','quatre':'vier','cinq':'fünf','six':'sechs','sept':'sieben'
}
```

1. Ecrire une fonction ***traduction*** qui prend en paramètres une liste de mots ***l*** et un dictionnaire ***d*** et qui retourne la liste des mots de ***l*** traduits à partir du dictionnaire ***d*** . On supposera que tous les mots de ***l*** sont une cle du dictionnaire.

2. Ecrire une fonction ***inverse*** qui prend en paramètre un dictionnaire ***d*** et qui retourne le dictionnaire inverse.
   On suppose ici qu'une même valeur n'apparaît pas plusieurs fois dans le dictionnaire ***d***.
3. Ecrire une fonction ***composition*** qui prend en paramètres deux dictionnaires ***d1*** et ***d2*** et qui retourne le dictionnaire correspondant à la composition des traductions.
   On suppose ici que toutes les valeurs de ***d1*** sont des clés de ***d2*** . Par exemple, `composition(danfr,dfral)` doit retourner un dictionnaire comportant toutes les associations cle/valeur suivantes :
   {'one' : 'eins', 'two' : 'zwei', 'three' : 'three', 'four' : 'vier', 'five' : 'fünf', 'six' : 'sechs', 'seven' : 'sieben'}

▷ **Exercice 5** - Des Anagrammes ⋆ ⋆ ⋆

Deux mots sont des anagrammes s'ils ont les mêmes lettres, mais dans des ordres différents. Par exemple, CRIME et MERCI sont des anagrammes.
L'indice d'un mot est la suite ordonnée de ses lettres. Par exemple l'indice du mot KAYAK est AAKKY.
Deux mots forment donc des anagrammes lorsqu'ils ont des indices identiques. Par exemple, CRIME et MERCI ont bien le même indice CEIMR.

1. Ecrire une fonction ***calculer_indice*** qui prend en paramètre un mot ***m*** et qui retourne l'indice du mot ***m***.
2. Ecrire une fonction ***anagrammes*** qui prend en paramètre deux mots ***m1*** et ***m2*** et qui teste si les deux mots donnés sont des anagrammes
3. Ecrire une fonction ***dicoindice*** qui prend en paramètre une liste de mots ***l*** et qui retourne un dictionnaire. Dans ce dictionnaire, les clés sont les indices, et à chaque indice est associé la liste des mots correspondants.