

TD n°4

1 Manipulations basiques sur les dictionnaires

Exercice 1. Création d'un dictionnaire

On souhaite créer un dictionnaire (ou tableau associatif) avec les trois paires **clé** / **valeur** suivantes : "nom" : "Dupont", "prénom" : "Antoine" et "âge" : 27.

Donnez deux syntaxes différentes pour y arriver :

- la première crée directement le dictionnaire en donnant les paires clé / valeur au cours de l'affectation initiale,
- la deuxième crée le dictionnaire vide, puis crée les associations clé / valeur une à une.

Exercice 2. Création d'un autre dictionnaire, avec une boucle

Écrivez une fonction *monDico* qui prend en paramètre un entier n et qui retourne un dictionnaire de n entrées, où les clés sont les entiers i entre 1 et n , et les valeurs sont les chaînes de caractères de la forme "nombre 1", "nombre 2", ...

Ainsi, `monDico(3)` retournera `{1 : "nombre 1", 2 : "nombre 2", 3 : "nombre 3"}`.

Exercice 3. Inversion d'un dictionnaire

Écrivez une fonction *inverse* qui prend en paramètre un dictionnaire et qui retourne le dictionnaire inversé : les clés du dictionnaire passé en paramètre (respectivement les valeurs) deviennent les valeurs du dictionnaire retourné (respectivement les clés).

On supposera pour cela que les valeurs du dictionnaire passé en paramètre sont uniques (indispensable pour assurer l'unicité des clés du dictionnaire retourné).

2 Recettes de cuisine

On représente une recette de cuisine par un dictionnaire dont les clés sont des noms d'ingrédients, et les valeurs sont la quantité nécessaire de chaque ingrédient pour réaliser la recette. Les unités peuvent être différentes (litre pour les liquides, kg pour les solides etc) mais ce n'est pas le propos de l'exercice. Voici deux exemples de recettes :

- crepe = {'oeufs' : 3, 'farine' : 0.25, 'lait' : 0.5}
- galette = {'oeufs' : 4, 'sucre' : 0.1, 'amande' : 0.2, 'lait' : 0.8}

Exercice 1. Les ingrédients communs à deux recettes

Écrivez une fonction *ingredients_communs* prenant en paramètres deux recettes et renvoyant les ingrédients communs aux deux recettes dans un dictionnaire ayant comme clé le nom de l'ingrédient et comme valeur le tuple des deux quantités nécessaires de l'ingrédient commun pour réaliser chacune des deux recettes.

Par exemple :

`ingredients_communs(crepe, galette)` retourne `{'lait' : (0.5, 0.8), 'oeufs' : (3, 4)}`.

Exercice 2. Achat possible dans un magasin

On représente un magasin par un dictionnaire dont les clés sont les ingrédients disponibles, et les valeurs un couple (c'est-à-dire un tuple de deux éléments) contenant le prix unitaire et la quantité en stock de l'ingrédient dans le magasin.

Par exemple :

- `mag1 = {'oeufs' : (0.52, 24), 'farine' : (1.15, 30), 'sucre' : (4, 10), 'lait' : (5, 20)}`
- `mag2 = {'oeufs' : (1, 12), 'lait' : (15, 30), 'sucre' : (6, 20)}`
- `mag3 = {'oeufs' : (0.48, 5), 'farine' : (1.30, 30), 'lait' : (5, 21)}`

Écrivez une fonction *achat_possible* prenant en paramètre une recette et un magasin et renvoyant *True* ou *False* selon que l'on peut ou non se procurer dans le magasin chaque ingrédient de la recette en quantité suffisante.

Par exemple :

- `achat_possible(crepe, mag1)` renverra *True* ;
- `achat_possible(crepe, mag2)` renverra *False* ;
- `achat_possible(crepe, mag3)` renverra *True*.

Exercice 3. Le coût d'achat dans un magasin

Écrivez une fonction *calculer_cout* prenant en paramètre une recette et un magasin et renvoyant le prix total pour acheter les ingrédients de la recette dans ce magasin.

Si cela n'est pas possible (quand un ingrédient est absent ou en quantité insuffisante), alors c'est la valeur -1 qui est renvoyée.

Par exemple :

- `calculer_cout(crepe, mag1)` renverra 4.3475 ;
- `calculer_cout(crepe, mag2)` renverra -1.

Exercice 4. Achat au plus bas coût dans une liste de magasins

On représente l'ensemble des magasins par un dictionnaire dont les valeurs sont les magasins et les clés des chaînes de caractères identifiant les magasins.

Par exemple :

`les_magasins = {"magasin 1" : mag1, "magasin 2" : mag2, "magasin 3" : mag3}`.

On souhaite, pour une recette donnée et un ensemble donné de magasins, savoir dans quel magasin faire ses achats, au coût le moins élevé.

Écrivez une fonction ***meilleur_magasin*** qui prend en paramètres un dictionnaire de magasins, une recette, et qui retourne un tuple composé du nom du magasin (la chaîne de caractères) et du prix d'achat des ingrédients dans ce magasin.

Si aucun des magasins ne permet les achats, la fonction retourne le tuple `(None, -1)`.

Par exemple :

- `meilleur_magasin(les_magasins, crepe)` retourne `('magasin 3', 4.265)`
- `meilleur_magasin(les_magasins, galette)` retourne `(None, -1)`

3 Base de données musicale

Un album est représenté par un tuple contenant :

- une chaîne de caractères (le titre de l'album) ;
- un dictionnaire avec les titres des morceaux comme clés, et leur durée comme valeur (durée exprimée en secondes).

Par exemple :

`war = ("war", {"sunday bloody sunday" : 279, "like a song" : 287, "new year's day" : 335, "40" : 157})`

Exercice 1. Le titre est-il dans l'album ?

Écrivez une fonction ***in_album*** qui prend en paramètre un album et un nom de morceau et qui teste (qui renvoie un booléen, donc) si le morceau est dans l'album.

Par exemple

- `in_album(war, "like a song")` renverra `True` ;
- `in_album(war, "with or without you")` renverra `False`.

Exercice 2. Durée d'un titre

Écrivez une fonction ***duree_piste*** qui prend en entrée un album et un nom de morceau, et qui retourne sa durée. Si le morceau n'est pas dans l'album, la fonction renvoie -1.

Par exemple, `duree_piste(war, "like a song")` renverra 287 et `duree_piste(war, "with or without you")` renverra -1.

Exercice 3. Durée d'un album

Écrivez une fonction **duree** qui prend en entrée un album, et qui retourne sa durée totale.
Par exemple, `duree(war)` renverra 1058.

Exercice 4. Les titres courts d'un album

Écrivez une fonction **courts** qui prend en entrée un album et qui renvoie la liste des titres durant moins de 3 minutes.

Par exemple, `courts(war)` renverra `["40"]`.

Une discothèque en ligne est représentée par un dictionnaire, où les clés sont les noms des artistes, et les valeurs sont les listes d'albums de ces artistes.

On suppose que les albums sont des tuples (comme dans les questions précédentes) déjà créés dans la base.

Par exemple : `discotheque = {"u2" : [war, the_joshua_tree, achtung_baby], "green day" : [american_idiot, dookie]}`.

On suppose pour l'exemple que les noms des variables correspondent aux titres des albums (par exemple, `achtung_baby = ("achtung baby", {"zoo station" : 276, "one" : 276, "the fly" : 269})`).

Exercice 5. Les albums d'une discothèque

Écrivez une fonction **albums** qui prend une discothèque en paramètre, et qui retourne la liste de tous les albums y figurant, sous forme de couple (`nom_artiste`, `nom_album`).

Sur l'exemple, `albums(discotheque)` renverrait la liste `[("u2", "war"), ("u2", "the joshua tree"), ("u2", "achtung baby"), ("green day", "american idiot"), ("green day", "dookie")]`

Exercice 6. Recherche

On veut faire une recherche dans la base de données. Écrivez une fonction **recherche** qui prend en paramètre une discothèque et un nom de chanson, et qui retourne un message donnant le nom de l'artiste et de l'album correspondant s'ils existent dans la base.

Par exemple :

- `recherche(discotheque, "sunday bloody sunday")` renverra `"sunday bloody sunday : artiste = u2, album = war"` ;
- `recherche(discotheque, "boulevard of broken dreams")` renverra `"boulevard of broken dreams : pas encore dans la base"`.