Les tuples, dico, set

Documentation officielle :
https://docs.python.org/fr/3/tutorial/datastructures.html

1. Les tuples

Un tuple est une liste ordonnée d'éléments, de type séquence et immuables.
Le tuple vide se déclare avec des parenthèses vides : **mon_tuple = ()**

Un tuple peut contenir des éléments de type différents.

Nous pourrons lire dans un tuple de la même manière qu'avec les chaines de caractères et nous ne pourrons pas remplacer un élément du tuple ou écrire dans le
tuple une fois celui-ci défini.

L'opérateur ***in*** permet de tester l'appartenance d'un élément à un tuple, le nombre d 'éléments d'un tuple peut être connu grâce à l'instruction **len()**.

Le tuple dispose de méthodes (c'est un objet) qui permettront par exemple de connaître l'index d'un élément (**mon_tuple.index('element')**) ou le nombre d'occurrences d'un élément dans celui-ci (**mon_tuple.count('element')**).

Exercice :

1.facture.py

Le service comptabilité a réalisé une statistique sur les différents produits informatique vendu pendant la semaine qui se déroule du 18 janvier au 22 janvier 2021. Après plusieurs comparaisons, vous vous êtes rendu compte :

- *Qu'il manque le 22 janvier 2021 dans le rapport ou l'entreprise Macumba a acheté des fournitures pour 37.58 HT soit 47.58 TTC.*
- *Qu'il y a une information faussée le 18 janvier au niveau de l'acheteur (IBM).*
- *Que la dernière ligne est un doublon*

Date d'achat	Prix HT	Prix TTC	Acheteur
18 Jan 2021	43.50	92.45	IMB
19 Jan 2021	42.80	51.19	Thales
20 Jan 2021	42.10	34.87	Atos
21 Jan 2021	37.58	37.58	Wordline
21 Jan 2021	37.58	1.58	Wordline

A l'aide du cours et de vos connaissances en programmation aidez le service comptabilité en ajoutant, modifiant et supprimant les informations demandées.

2. Les sets

Un ensemble est une structure de données intégrée à Python qui présente les trois caractéristiques suivantes.

Non ordonné : Les éléments d'un ensemble ne sont pas ordonnés, contrairement aux listes, c'est-à-dire que l'ordre dans lequel les éléments sont insérés n'est pas conservé. Les éléments seront dans un ordre différent à chaque fois que nous accéderons à l'objet Set. Aucune valeur d'index n'est attribuée à chaque élément de l'ensemble.
Immuable : Les éléments de l'ensemble doivent être immuables. Nous ne pouvons pas changer les éléments de l'ensemble, c'est-à-dire que nous ne pouvons pas modifier la valeur des éléments. Mais nous pouvons ajouter ou supprimer des éléments à l'ensemble.
Unique : Il ne peut y avoir deux éléments ayant la même valeur dans l'ensemble.

```
1  votreEnsemble = {1, 2, 3}
2  votreEnsemble = set('ensemble')
3  print(votreEnsemble)
```

Output:

```
1  {'l', 'm', 'n', 's', 'e', 'b'}
2
```

Exercice :

2.add_to_the_set.py

Tâches à réaliser :

Ajouter dans le set suivant le mot **poire** :
{"Pomme", "banane", "cerise"}

Affichage sera du type :

{"Pomme", "banane", "cerise", "poire"}

3.update_list.py

Tâches à réaliser :

Ajouter la liste d'éléments suivant à un ensemble :
sampleSet = {"Yellow", "Orange", "Black"}
sampleList = ["Blue", "Green", "Red"]

Affichage sera du type :

{'Green', 'Yellow', 'Black', 'Orange', 'Red', 'Blue'}

4.identical_items.py

Tâches à réaliser :

Doit comparer et retourner les numéros identiques de chaque liste :
set1 = {10, 20, 30, 40, 50}
set2 = {30, 40, 50, 60, 70}

Affichage sera du type :

{40, 50, 30}

5.remove_items.py

Tâches à réaliser :

Retirer 10, 20, 30 de la liste suivante :
set1 = {10, 20, 30, 40, 50}

Affichage sera du type :

{40, 50}

3. Les dictionnaires

Un dictionnaire est non ordonné, il n'est pas de type séquence.

Chaque élément du dictionnaire contient une association clé/valeur. Un dictionnaire se définit avec des accolades. Un dictionnaire vide s'écrit donc :
mon_dico = {}

Un dictionnaire peut contenir des éléments de types différents.
Exemple :

```
>>> mon_dico = {}
>>> mon_dico
{}
>>>
>>> mon_dico = {'cle':'valeur', 'test': 5}
>>> mon_dico
{'test': 5, 'cle': 'valeur'}
>>>
```

Nous pouvons récupérer indépendamment les clés ou les valeurs du dictionnaire ou l'ensemble.

Les méthodes utilisées seront keys() pour récupérer les clés , values() pour les valeurs et items() pour les clés et valeurs.

```
>>>
>>> mon_dico['cle']
'valeur'
>>>
>>>
>>> mon_dico.values()
[5, 'valeur']
>>> mon_dico.keys()
['test', 'cle']
>>> mon_dico.items()
[('test', 5), ('cle', 'valeur')]
>>>
```