Devoir Surveillé 2 : Listes et Dictionnaires en Python

L'évaluation porte sur 3 exercices indépendants.
Les exercices sont notés sur 18 et la rigueur, rédaction et justifications sont notés sur 2 points.
La présence des en-têtes de fonctions ou documentations compte dans les deux points de rigueur.

Exercice 1 : Relevés de températures (6 points)

On dispose d'une liste de relevés de températures (en °C) sur un mois (31 jours). Par exemple :

```
temperatures = [5, 6, 10, 10, 12, 15, 15, 16, 18, 19, 20, 20, 22, 25, 25, 25, 24, 23, 21, 20, 19, 17, 15, 13, 10, 8, 7
```

1. **Écrire une fonction `moyenne` qui prend en paramètre une liste de températures et renvoie la valeur moyenne de la liste.**
    - Il faudra gérer le cas où la liste serait vide.

    Exemple :

```
>>> moyenne(temperatures)
15.0
```

2. **Écrire une fonction `max_temperature` qui prend en paramètre une liste de températures et renvoie un tuple `(temperature_max, jour)` où :**
    - `temperature_max` est la température la plus élevée de la liste ;
    - `jour` le jour auquel cette température apparaît pour la première fois.

    Exemple :

```
>>> max_temperature(temperatures)
(25, 13)
```

3. **Écrire une fonction `jours_au_dessus_de` qui prend en paramètres une liste de températures et un seuil, et renvoie une liste des indices pour lesquels la température dépasse le seuil donné.**

    Exemple :

```
>>> jours_au_dessus_de(temperatures, 22)
[13, 14, 15]
```

4. **Écrire une fonction `supprimer_valeurs` qui prend en paramètres une liste de températures et une valeur, et supprime toutes les occurrences de cette valeur dans la liste.**

    Exemple :

```
>>> supprimer_valeurs(temperatures, 20)
>>> print(temperatures)
[5, 6, 10, 10, 12, 15, 15, 16, 18, 19, 22, 25, 25, 25, 24, 23, 21, 19, 17, 15, 13, 10, 8, 7, 6, 6, 5, 5]
```

Exercice 2 : Gestion d'un parc automobile (6 points)

On dispose d'une liste de dictionnaires **représentant un concessionnaire**, chacun représentant un modèle de voiture :

```
    concessionnaire = [
        {"modele": "BMW Série 1", "prix": 27000, "energie": "Essence"},
        {"modele": "BMW X1", "prix": 50000, "energie": "Essence"},
        {"modele": "Mini Cooper", "prix": 22000, "energie": "Essence"},
        {"modele": "Mini Clubman", "prix": 25000, "energie": "Diesel"},
        {"modele": "Mini Cooper", "prix": 21000, "energie": "Diesel"}
    ]
```

1. **Écrire une fonction `recherche_voiture` qui prend en paramètres un concessionnaire et un modèle et qui renvoie une liste comportant tous les véhicules du modèle passé en paramètre.**
   Exemple :

```
>>> recherche_voiture(concessionnaire, "Mini Cooper")
[{ "modele": "Mini Cooper", "prix": 22000, "energie": "Essence"},
 {"modele": "Mini Cooper", "prix": 21000, "energie": "Diesel"}]
```

2. **Écrire une fonction `soldes_prix_energie` qui prend en paramètres un concessionnaire et une energie et modifie le prix de ces modèles en le réduisant de 2500€.**
   Exemple :

```
>>> soldes_prix_energie(concessionnaire,"Essence")
>>> print(concessionnaire)
[
    {"modele": "BMW Série 1", "prix": 24500, "energie": "Essence"},
    {"modele": "BMW X1", "prix": 47500, "energie": "Essence"},
    {"modele": "Mini Cooper", "prix": 19500, "energie": "Essence"},
    {"modele": "Mini Clubman", "prix": 25000, "energie": "Diesel"},
    {"modele": "Mini Cooper", "prix": 21000, "energie": "Diesel"}
]
```

3. **Écrire une fonction `prix_moyen` qui prend en paramètre un concessionnaire et une energie et qui**

renvoie le prix moyen des véhicules de cette energie.
Exemple :

```
>>> prix_moyen(concessionnaire, "Diesel")
23000
```

4. **Écrire une fonction `modele_le_plus_cher` qui prend en paramètre un concessionnaire et qui renvoie le modèle le plus cher.**
   Exemple :

```
>>> modele_le_plus_cher(concessionnaire)
"BMW X1"
```

Exercice 3 : Gestion d'un Refuge de Renards (6 points)

On dispose d'une liste de tuples pour stocker les informations des renards dans un refuge. Chaque tuple contient les informations suivantes : (identifiant, nom, espèce, race, sexe).

Par exemple :

```
renards = [
    (1, 'Foxy', 'Renard', 'Rouge', 'Femelle'),
    (2, 'Rex', 'Renard', 'Arctique', 'Mâle'),
    (3, 'Coco', 'Renard', 'Rouge', 'Mâle')
]
```

1. **Écrire une fonction `ajouter_renard_tuples` qui prend en paramètres la liste `renards` et un nouveau tuple `renard`, et qui ajoute ce tuple à la liste.**
   Exemple :

```
>>> ajouter_renard_tuples(renards, (4, 'Luna', 'Renard', 'Rouge', 'Femelle'))
>>> print(renards)
[(1, 'Foxy', 'Renard', 'Rouge', 'Femelle'), (2, 'Rex', 'Renard', 'Arctique', 'Mâle'), (3, 'Coco', 'Renard', 'Rouge', 'Mâle
```

2. **Écrire une fonction `modifier_renard_tuples` qui prend en paramètres la liste `renards`, un identifiant, un nouveau nom et une nouvelle race, et qui modifie le nom et la race du renard correspondant à l'identifiant donné.**
   Exemple :

```
>>> modifier_renard_tuples(renards, 1, 'Fiona', 'Argenté')
>>> print(renards)
[(1, 'Fiona', 'Renard', 'Argenté', 'Femelle'), (2, 'Rex', 'Renard', 'Arctique', 'Mâle'), (3, 'Coco', 'Renard', 'Rouge', 'M
```

3. **Écrire une fonction `afficher_renards_rouges` qui prend en paramètre la liste `renards` et affiche uniquement les renards de la race "Rouge".**
   Exemple :

```
>>> afficher_renards_rouges(renards)
(1, 'Fiona', 'Renard', 'Rouge', 'Femelle')
(3, 'Coco', 'Renard', 'Rouge', 'Mâle')
```

4. **Expliquer en quelques lignes pourquoi, dès que l'on souhaite effectuer de multiples modifications (changer seulement la race, ou seulement le nom, etc.) sur des champs précis, une liste de tuples devient peu pratique. Pourquoi un modèle plus « flexible » est-il préférable pour des données évolutives ?**
5. **Transformer cette liste de tuples en une liste de dictionnaires.**
   Par exemple, chaque renard deviendra un dictionnaire du type :

```
{
   'id': 1,
   'nom': 'Fiona',
   'espece': 'Renard',
   'race': 'Rouge',
   'sexe': 'Femelle'
}
```

Écrire une fonction `listes_tuples_vers_listes_dict` qui renvoie une nouvelle liste de dictionnaires construite à partir de la liste `renards`.
Exemple :

```
>>> listes_tuples_vers_listes_dict(renards)
[
    {'id': 1, 'nom': 'Fiona', 'espece': 'Renard', 'race': 'Rouge', 'sexe': 'Femelle'},
    {'id': 2, 'nom': 'Rex', 'espece': 'Renard', 'race': 'Arctique', 'sexe': 'Mâle'},
    {'id': 3, 'nom': 'Coco', 'espece': 'Renard', 'race': 'Rouge', 'sexe': 'Mâle'}
]
```

6. **Réécrire la fonction de modification pour qu'elle fonctionne sur la liste de dictionnaires :**
   La fonction `modifier_renard_dict` modifie directement les champs 'nom' et 'race' dans le dictionnaire

correspondant.

Exemple :

```
>>> modifier_renard_dict(renards_dict, 1, 'Fiona', 'Argenté')
>>> print(renards_dict)
[
    {'id': 1, 'nom': 'Fiona', 'espece': 'Renard', 'race': 'Argenté', 'sexe': 'Femelle'},
    {'id': 2, 'nom': 'Rex', 'espece': 'Renard', 'race': 'Arctique', 'sexe': 'Mâle'},
    {'id': 3, 'nom': 'Coco', 'espece': 'Renard', 'race': 'Rouge', 'sexe': 'Mâle'}
]
```