Exercices Avancés : Tuples, Dictionnaires et Ensembles en Python

---

Exercice 1 : Analyse de données étudiants

Vous disposez d'une liste de tuples, chacun représentant un étudiant sous la forme suivante :
% Chaque tuple : (identifiant, nom, prénom, cours, note)

```
data = [
    (1, "Khaledi", "Youssef", "Math", 15),
    (2, "Mansouri", "Sami", "Math", 12),
    (3, "Farid", "Lina", "Physique", 18),
    (4, "Nasseri", "Omar", "Math", 8),
    (5, "Jaberi", "Noura", "Physique", 16),
    # ... (vous pouvez imaginer d'autres enregistrements pour tester vos fonctions)
]
```

Questions :

1. **Q1.** Écrire une fonction `stats_cours(data)` qui analyse la liste et retourne un dictionnaire.
   - **Clé** : le nom d'un cours
   - **Valeur** : un tuple contenant trois statistiques sur les notes associées à ce cours :
     - La note minimale
     - La note maximale
     - La moyenne des notes (arrondie à deux décimales)
2. **Q2.** Modifier la fonction précédente pour ne retourner que les cours dont l'étendue (différence entre la note maximale et la note minimale) est supérieure ou égale à 5.
3. **Q3.** Afficher les résultats de la fonction en triant les cours par ordre alphabétique des noms de cours.

Exercice 2 : Gestion de compétences

Vous avez une liste de tuples représentant des employés et leurs compétences. Chaque tuple est de la forme :
% Chaque tuple : (nom, prénom, ensemble_des_compétences)

```
employes = [
    ("Salem", "Amina", {"Python", "SQL", "Machine Learning"}),
    ("Khalifa", "Zayd", {"Python", "C++"}),
    ("Moussa", "Rania", {"Java", "Python", "SQL"}),
    ("Hamzaoui", "Tariq", {"C++", "Java"}),
    ("ElAmri", "Fatima", {"Machine Learning", "Python", "Java"}),
    # ... (d'autres exemples à prévoir)
]
```

Questions :

1. **Q1.** Écrire une fonction `competences_totales(employes)` qui retourne un ensemble contenant **toutes** les compétences disponibles parmi les employés.
2. **Q2.** Écrire une fonction `employes_par_competence(employes)` qui retourne un dictionnaire construit de la manière suivante :
   - **Clé** : le nom d'une compétence
   - **Valeur** : un ensemble (ou un tuple) contenant les noms complets (format « Nom Prénom ») des employés maîtrisant cette compétence.
3. **Q3.** À l'aide de la fonction précédente, afficher (ou retourner) toutes les compétences maîtrisées par au moins 3 employés.

Exercice 3 : Opérations sur ensembles et tuples avancées

Ce problème comporte plusieurs parties, chacune utilisant les opérations sur ensembles et les tuples.

1. **Q1.** Écrire une fonction `ensemble_operations(set1, set2)` qui prend en entrée deux ensembles d'entiers et retourne un tuple contenant :
   - L'intersection des deux ensembles
   - L'union des deux ensembles
   - La différence (`set1 - set2`)
   - La différence symétrique entre `set1` et `set2`

   Vous utiliserez les opérateurs classiques sur les ensembles en Python (ex. : `&`, `|`, `-`, `^`).
2. **Q2.** Écrire une fonction `analyze_numbers(numbers)` qui prend une liste d'entiers et retourne un dictionnaire ayant pour clés :
   - `"pairs"` : un ensemble des nombres pairs présents dans la liste
   - `"impairs"` : un ensemble des nombres impairs présents dans la liste
   - `"min_max"` : un tuple contenant le minimum et le maximum des nombres de la liste

   Pensez à gérer les cas où la liste pourrait être vide pour éviter une erreur.
3. **Q3.** Modifier la fonction `analyze_numbers(numbers)` de façon à ce que les doublons soient exclus dès le départ. C'est-à-dire, convertissez la liste en un ensemble avant d'effectuer les calculs.
4. **Q4.** Rédiger quelques tests unitaires (au moins 2 par fonction créée dans cet exercice) pour vérifier le bon fonctionnement de vos fonctions. Vous pouvez utiliser le module `unittest` ou simplement des assertions dans une section de test (dans le fichier Python).

Bon courage !