Manipulation des Tuples et des Dictionnaires

Cours Magistral

1. Introduction aux Tuples

Définition

Un **tuple** est une structure de données immuable en Python, c'est-à-dire que ses éléments ne peuvent pas être modifiés après leur création. Il est défini à l'aide de parenthèses ().

Création d'un Tuple

```
coordonnees = (3, 4, 5)  # Tuple contenant trois entiers
```

Accès aux éléments d'un Tuple

On peut accéder aux éléments d'un tuple à l'aide d'indices, qui commencent à 0.

```
print(coordonnees[0])  # Affiche 3
print(coordonnees[1])  # Affiche 4
```

Opérations sur les Tuples

- **Concaténation** : (1, 2) + (3, 4) → (1, 2, 3, 4)
- **Répétition** : (1, 2) * 2 → (1, 2, 1, 2)
- **Appartenance** : 3 in (1, 2, 3) → True

2. Introduction aux Dictionnaires

Définition

Un **dictionnaire** est une structure de données qui stocke des paires clé-valeur, permettant une récupération rapide des valeurs via leurs clés. Il est défini à l'aide d'accolades {}.

Création d'un Dictionnaire

```
etudiant = {"nom": "Alice", "age": 20, "note": 15.5}
```

Accès aux éléments

```
print(etudiant["nom"])  # Affiche "Alice"
print(etudiant["age"])  # Affiche 20
```

Ajout et Suppression d'Éléments

```
etudiant["matiere"] = "Mathématiques"  # Ajout d'une nouvelle clé
print(etudiant)
del etudiant["age"]  # Suppression d'un élément
print(etudiant)
```

Utilisation d'un Dictionnaire : Stockage des Informations des Étudiants

Un dictionnaire peut être utilisé pour stocker des informations sur plusieurs étudiants :

```
etudiants = {
    "Alice": {"age": 20, "note": 15.5},
    "Bob": {"age": 22, "note": 12.0},
}
print(etudiants["Alice"]["note"])  # Affiche 15.5
```

3. Les Opérations sur les Dictionnaires

- **Vérifier la présence d'une clé**

```
if "nom" in etudiant:
    print("La clé 'nom' existe dans le dictionnaire.")
```

- **Obtenir toutes les clés ou valeurs**

```
print(etudiant.keys())  # Renvoie toutes les clés
print(etudiant.values())  # Renvoie toutes les valeurs
```

- **Obtenir tous les éléments sous forme de tuples (clé, valeur)**

```
print(etudiant.items())
```

- **Copier un dictionnaire**

```
    nouveau_dico = etudiant.copy()
```

- **Récupérer une valeur avec une valeur par défaut**

```
note = etudiant.get("note", "Non disponible")
```

- **Supprimer un élément et récupérer sa valeur**

```
age = etudiant.pop("age", "Clé absente")
print(age)
```

4. Parcourir un Dictionnaire avec la Boucle for

- **Parcourir les clés**

```
for cle in etudiant:
    print(cle)
```

- **Parcourir les valeurs**

```
for valeur in etudiant.values():
    print(valeur)
```

- **Parcourir les paires clé-valeur**

```
for cle, valeur in etudiant.items():
    print(f"{cle} : {valeur}")
```

TP1

Exercice 1 : Calcul de la Distance entre Deux Points

Écrire un programme qui utilise un **tuple** pour stocker les coordonnées (x, y, z) d'un point dans l'espace et calcule la distance entre deux points selon la formule :

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Solution

```
import math

def distance_3D(point1, point2):
    return math.sqrt(
        (point2[0] - point1[0])**2 +
        (point2[1] - point1[1])**2 +
        (point2[2] - point1[2])**2
    )

p1 = (1, 2, 3)
p2 = (4, 6, 8)
print("Distance:", distance_3D(p1, p2))
```

Exercice 2 : Gestion des Notes des Étudiants

Écrire un programme qui utilise un **dictionnaire** pour stocker les notes des étudiants et permet d'afficher la note d'un étudiant donné.

Solution

```
# Création du dictionnaire des notes
notes_etudiants = {
    "Alice": 15.5,
    "Bob": 12.0,
    "Charlie": 18.0
}

# Demander le nom de l'étudiant
nom = input("Entrez le nom de l'étudiant : ")

# Vérifier si l'étudiant est dans le dictionnaire
if nom in notes_etudiants:
    print(f"La note de {nom} est {notes_etudiants[nom]}.")
else:
    print("Étudiant non trouvé.")
```

TP2

Exercice 1 : Stockage des Coordonnées GPS dans un Tuple

Écrire un programme qui utilise un tuple pour stocker les coordonnées GPS (latitude, longitude) de plusieurs villes et afficher les coordonnées d'une ville donnée.

```
# Dictionnaire contenant les coordonnées GPS des villes
villes = {
    "Paris": (48.8566, 2.3522),
    "New York": (40.7128, -74.0060),
    "Tokyo": (35.6762, 139.6503)
}

# Demander à l'utilisateur de saisir le nom d'une ville
ville = input("Entrez le nom de la ville : ")

# Vérifier si la ville existe dans le dictionnaire
if ville in villes:
    print(f"Les coordonnées de {ville} sont : {villes[ville]}")
else:
    print("Ville non trouvée.")
```

Exercice 2 : Manipulation des Tuples et des Dictionnaires pour un Agenda

Écrire un programme pour gérer un agenda où chaque rendez-vous est une paire de valeurs (date, description) dans un dictionnaire. Le programme devra afficher tous les rendez-vous d'un jour donné.

Solution

```
# Dictionnaire des rendez-vous
agenda = {
    "2025-03-24": [("10:00", "Réunion avec le client"), ("14:00",
"Cours de Python")],
    "2025-03-25": [("09:00", "Séminaire IA"), ("16:00", "Réunion de
projet")]
}

# Demander la date pour afficher les rendez-vous
date = input("Entrez la date (format YYYY-MM-DD) : ")

if date in agenda:
    print(f"Rendez-vous pour le {date}:")
    for heure, description in agenda[date]:
        print(f"{heure}: {description}")
else:
    print("Aucun rendez-vous pour cette date.")
```

Exercice 3 : Traitement d'une Liste de Dictionnaires

Supposons que vous avez une liste de dictionnaires contenant des informations sur des employés. Écrivez un programme pour calculer le salaire moyen des employés.

Solution

```
# Liste des employés avec leurs informations
employes = [
    {"nom": "Alice", "salaire": 3000},
    {"nom": "Bob", "salaire": 3500},
    {"nom": "Charlie", "salaire": 4000}
]

# Calcul du salaire moyen
salaire_total = sum(employe["salaire"] for employe in employes)
salaire_moyen = salaire_total / len(employes)
print(f"Le salaire moyen des employés est : {salaire_moyen:.2f}")
```

Exercice 4 : Trouver le Plus Grand Élément dans un Tuple

Écrire un programme qui trouve le plus grand nombre dans un tuple de nombres.

Solution

```
# Tuple de nombres
nombres = (2, 5, 9, 3, 6, 8)

# Trouver le plus grand nombre
plus_grand = max(nombres)
print(f"Le plus grand nombre est : {plus_grand}")
```

Exercice 5

Écrire une fonction compterMots ayant un argument (une chaîne de caractères) et qui renvoie un dictionnaire qui contient la fréquence de tous les mots de la chaîne entrée.