Exercices Pratiques de Python

Ali ZAINOUL

Exercices Pratiques de Python

1. **Fonctions Intégrées et Types de Base** :
   - Utilisez les fonctions `len()`, `input()`, `type()` et `print()` dans un programme simple qui demande à l'utilisateur son nom, son âge, et affiche une phrase utilisant ces informations.
2. **Booléens et Nombres** :
   - Créez un programme qui vérifie si un nombre donné par l'utilisateur est pair ou impair.
   - Écrivez un programme qui demande deux nombres à l'utilisateur et affiche leur somme, différence, produit et quotient.
3. **Notations et Conversions** :
   - Écrivez un programme qui convertit un nombre donné par l'utilisateur en notations binaire, octale et hexadécimale.
4. **Slicing des Chaînes de Caractères** :
   - Créez un programme qui demande à l'utilisateur d'entrer une chaîne de caractères et utilise le slicing pour extraire et afficher la première moitié de la chaîne.
5. **Slicing des Tuples** :
   - Développez un script qui crée un tuple de nombres et utilise le slicing pour extraire les éléments de l'indice 2 à 5.
6. **Utilisation de `enumerate()`** :
   - Créez un programme qui utilise `enumerate()` pour itérer sur une liste de noms et afficher chaque nom avec son index.

7. **Compréhensions de Listes et Tuples** :
    - Écrivez un programme qui utilise des compréhensions de listes pour créer une liste de carrés des nombres de 1 à 10.
8. **Filtrage de Séquences** :
    - Développez un script qui prend une liste de nombres et utilise une compréhension de liste pour créer une nouvelle liste contenant uniquement les nombres pairs.
9. **Fonctions sur les Chaînes** :
    - Créez un programme qui utilise différentes fonctions de la classe `str` (comme `split()`, `find()`, `replace()`, etc.) pour manipuler des chaînes de caractères fournies par l'utilisateur.
10. **Classes String (`str`) et Tuple** :
    - Écrivez un programme qui demande à l'utilisateur de saisir une chaîne de caractères et un tuple de mots, puis recherche si les mots du tuple sont présents dans la chaîne.
11. **Fonction `range()`** :
    - Créez un programme qui utilise la fonction `range()` pour générer une liste de nombres de 1 à 20 et affichez-les.
12. **Opérations sur les Listes** :
    - Créez des listes en utilisant le constructeur `list()` et la notation `[]`.
    - Utilisez les méthodes `append()`, `insert()`, `sort()`, `reverse()`, `remove()`, `extend()`, `pop()`, `clear()` sur une liste.
13. **Copie et Références** :
    - Écrivez un programme qui montre la différence entre une copie superficielle et une copie en profondeur d'une liste.
14. **Création et Manipulation des Dictionnaires** :
    - Créez des dictionnaires en utilisant différentes méthodes et accédez aux valeurs en utilisant des clés.

- Utilisez les méthodes `keys()`, `values()`, `items()`, `update()`, `get()` sur un dictionnaire.

15. **Création et Manipulation des Ensembles** :
    - Créez des ensembles et utilisez les opérateurs `-`, `|`, `&`, `^` pour effectuer des opérations ensemblistes.
16. **Conditions et Boucles** :
    - Écrivez des programmes utilisant les structures conditionnelles `if`, `elif`, `else`.
    - Utilisez les boucles `while` et `for` pour itérer sur des séquences.
17. **Instructions `break` et `continue`** :
    - Donnez des exemples d'utilisation des instructions `break` et `continue` dans des boucles.
18. **Concaténation de Chaînes** :
    - Écrivez un programme qui demande à l'utilisateur d'entrer plusieurs chaînes de caractères et les concatène en une seule chaîne en utilisant différentes méthodes (opérateur `+`, `join()`).
19. **Projet Final** :
    - Créez un mini-projet où l'utilisateur peut entrer plusieurs phrases. Le programme doit stocker chaque phrase dans un tuple et fournir des statistiques telles que la longueur moyenne des phrases, les mots les plus fréquents, etc.