



Licence 2 GL - IAGE

M. DIALLO

TP 1 : Types et Instructions de Base

Objectifs

- Maîtriser les types de base en Python (int, float, str, list)
- Manipuler les variables et les conversions de types
- Utiliser les chaînes de caractères et leurs méthodes avancées
- Travailler avec les listes et leurs opérations complexes
- Créer et utiliser des fonctions avec manipulation de données
- Développer une réflexion algorithmique sans structures conditionnelles

Consignes importantes

- Tous les exercices doivent être résolus **sans utiliser de structures conditionnelles** (if, elif, else)
- **Pas de boucles** (for, while) - elles seront vues dans les prochains chapitres
- **Pas de dictionnaires, sets ou structures de données avancées**
- Utilisez uniquement : les types de base (int, float, str), les listes, les tuples et les fonctions
- Exploitez les fonctions intégrées et les méthodes de chaînes/listes
- Commentez votre code pour expliquer votre raisonnement
- Testez vos fonctions avec plusieurs jeux de données

1 Exercices

Exercice 1 : Calculs avec formatage

Écrivez un programme qui :

- a) Demande à l'utilisateur de saisir trois nombres décimaux
- b) Calcule leur moyenne arithmétique
- c) Calcule leur moyenne géométrique : $\sqrt[3]{a \times b \times c}$
- d) Affiche les deux moyennes avec exactement 3 chiffres après la virgule
- e) Affiche la différence entre les deux moyennes

Exercice 2 : Manipulation de chaînes basique

Créez un programme qui :

- a) Demande à l'utilisateur de saisir son prénom et son nom
- b) Crée une adresse email au format : `prenom.nom@isi.sn` (en minuscules)
- c) Affiche le nombre total de caractères dans l'email
- d) Affiche l'email inversé (dernier caractère en premier)
- e) Extrait et affiche le domaine de l'email (la partie après @)

Exercice 3 : Opérations sur listes simples

Définissez deux listes : `liste1 = [5, 12, 8, 3, 17]` et `liste2 = [2, 4, 6, 8, 10]`, puis :

- a) Fusionnez les deux listes en une seule
- b) Triez la liste fusionnée
- c) Calculez la somme de tous les éléments avec `sum()`
- d) Calculez la moyenne des éléments
- e) Affichez le premier et le dernier élément de la liste triée

Exercice 4 : Calculs mathématiques sur listes

Écrivez une fonction `statistiques_base(nombres)` qui prend une liste de nombres et affiche :

- a) La moyenne : $\bar{x} = \frac{\sum x_i}{n}$
- b) L'étendue (différence entre max et min)
- c) La médiane (élément du milieu après tri de la liste)
- d) La somme des carrés : $\sum x_i^2$

Astuce : Utilisez `sum()`, `len()`, `sorted()`, `min()`, `max()` et l'opérateur `**`.

Exercice 5 : Manipulation de chaînes avancée

Créez une fonction `transformer_texte(texte)` qui prend une chaîne et retourne :

- a) Le texte avec tous les mots en majuscules
- b) Le texte avec la première et dernière lettre de chaque mot en majuscule
- c) Le nombre de voyelles dans le texte (a, e, i, o, u)
- d) La liste des mots triés par ordre alphabétique

Exemple : `transformer_texte("python est super")`

Exercice 6 : Opérations sur liste de listes

Créez une fonction `somme_lignes(matrice)` qui prend une matrice (liste de listes) et retourne une liste contenant la somme de chaque ligne.

Exemple :

```
1 matrice = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
2 # Resultat attendu: [6, 15, 24]
```

Créez aussi `somme_colonnes(matrice)` qui retourne la somme de chaque colonne.

Exercice 7 : Conversion et formatage

Écrivez une fonction `convertir_secondes(secondes)` qui convertit un nombre de secondes en format lisible :

- a) Calcule le nombre de jours
- b) Calcule le nombre d'heures restantes
- c) Calcule le nombre de minutes restantes
- d) Calcule le nombre de secondes restantes
- e) Retourne une chaîne formatée : "X jours, Y heures, Z minutes, W secondes"

Exemple : `convertir_secondes(90061)` → "1 jours, 1 heures, 1 minutes, 1 secondes"

Astuce : Utilisez les opérateurs `//` (division entière) et `%` (modulo).

Exercice 8 : Palindromes et inversions

Créez les fonctions suivantes qui travaillent avec des chaînes :

- a) `inverser(texte)` : Inverse une chaîne
- b) `est_palindrome(texte)` : Retourne `True` si le texte est identique à son inverse
- c) `plus_long_mot(phrase)` : Retourne le mot le plus long d'une phrase

Exercice 9 : Manipulation de listes numériques

Écrivez une fonction `operations_liste(liste)` qui prend une liste de nombres et retourne une nouvelle liste où :

- a) Chaque élément est multiplié par 2
- b) Les éléments négatifs sont remplacés par leur valeur absolue
- c) Les éléments sont triés dans l'ordre croissant

Exemple : `operations_liste([3, -5, 2, -1, 8])` → `[2, 4, 6, 10, 16]`

Exercice 10 : Calculs de prix

Créez une fonction `calculer_facture(prix_ht, taux_tva, remise)` qui :

- a) Calcule le montant de la TVA : $TVA = \text{prix_ht} \times \frac{\text{taux_tva}}{100}$
- b) Calcule le prix TTC : $TTC = \text{prix_ht} + TVA$
- c) Applique une remise en pourcentage : $\text{remise_montant} = TTC \times \frac{\text{remise}}{100}$
- d) Calcule le prix final : $\text{final} = TTC - \text{remise_montant}$
- e) Retourne une chaîne formatée avec tous ces montants arrondis à 2 décimales

Exercice 11 : Slicing et extraction

Créez une fonction `extraire_info(liste, debut, fin, pas)` qui :

- a) Extraît une sous-liste de `debut` à `fin` avec un pas de `pas`
- b) Inverse cette sous-liste

c) Calcule la somme des éléments de la sous-liste

d) Retourne la sous-liste inversée et la somme

Exemple : `extraire_info([1,2,3,4,5,6,7,8,9,10], 2, 8, 2)`

Extrait `[3, 5, 7]`, inverse en `[7, 5, 3]`, somme = 15

Exercice 12 : Formatage de données

Créez une fonction `formater_contacts(noms, prenom, ages)` qui prend trois listes de même longueur et crée une liste de chaînes formatées :

Format : "NOM Prenom (XX ans)"

Exemple :

```
1 noms = ["diallo", "sow", "fall"]
2 prenom = ["mamadou", "fatou", "ali"]
3 ages = [20, 22, 21]
4 # Resultat: ["DIALLO Mamadou (20 ans)",
5 #           "SOW Fatou (22 ans)",
6 #           "FALL Ali (21 ans)"]
```

Astuce : Utilisez `upper()`, `title()` et le slicing d'indices.

Exercice 13 : Analyse de texte simple

Créez une fonction `analyser_phrase(phrase)` qui retourne les informations suivantes sous forme de chaîne formatée :

- a) Nombre total de caractères
- b) Nombre de caractères sans espaces
- c) Nombre de mots (utiliser `split()`)
- d) Longueur moyenne des mots (arrondie à 2 décimales)
- e) Le mot le plus long
- f) Le mot le plus court

Exemple : `analyser_phrase("Python est un langage formidable")`

Conseils pour réussir

- Décomposez les problèmes complexes en sous-problèmes plus simples
- Utilisez les fonctions intégrées de Python : `sum()`, `len()`, `min()`, `max()`, `sorted()`, `abs()`
- Exploitez les méthodes de chaînes : `upper()`, `lower()`, `title()`, `split()`, `join()`, `replace()`
- Exploitez les méthodes de listes : `append()`, `extend()`, `sort()`, `reverse()`, `index()`
- Utilisez le slicing de manière créative : `liste[debut:fin:pas]`, `texte[::-1]`
- Les opérations mathématiques sont vos amies : `//`, `%`, `**`, `+`, `-`, `*`, `/`
- Testez avec des cas simples avant de passer aux cas complexes
- Commentez votre code pour expliquer votre raisonnement