

Travaux Pratiques - Partie 1

Structures de Contrôle Avancées en Python

`if...else`, `match...case`, `for`, `while`

M. DIALLO
L2 Informatique
`muustafa.dllo@gmail.com`

2025



Objectifs du TP

Ce TP vise à pratiquer les structures de contrôle fondamentales en Python :

- Conditions avec `if...elif...else`
- Boucles `for` pour parcourir des séquences
- Boucles `while` pour les répétitions conditionnelles
- Manipulation de listes
- Contrôles de saisie utilisateur

Exercice 1 : Vérificateur de nombre

Écrivez un programme qui demande un nombre à l'utilisateur et affiche un message indiquant si :

- Le nombre est positif, négatif ou nul
- Le nombre est pair ou impair
- Le nombre est premier ou pas
- Le nombre est parfait ou pas
- Le nombre est compris entre 10 et 50 (inclus)

Exemple d'exécution :

```
1 Entrez un nombre : 25
2 25 est positif, impair, pas premier, pas parfait et compris entre 10 et
  50.
```

Exercice 2 : Calcul de moyenne avec contrôle de saisie

Créez un programme qui demande 5 notes (entre 0 et 20) et calcule la moyenne. Le programme doit :

- Vérifier que chaque note est entre 0 et 20
- Redemander la note si elle est invalide
- Afficher la moyenne et la mention (Passable ≥ 10 , Bien ≥ 14 , Tres Bien ≥ 16)

Exemple d'exécution :

```
1 Note 1 : 15
2 Note 2 : 25 (Erreur, entrez une note entre 0 et 20)
3 Note 2 : 12
4 Note 3 : 18
5 Note 4 : 8
6 Note 5 : 14
7
8 Moyenne : 13.4 - Mention : Bien
```

Exercice 3 : Suppression des doublons dans une liste

Écrivez un programme qui :

1. Demande à l'utilisateur d'entrer 10 nombres
2. Stocke ces nombres dans une liste
3. Crée une nouvelle liste sans doublons
4. Affiche les deux listes

Contrainte : N'utilisez pas `set()` pour cet exercice.

Exemple d'exécution :

```
1 Liste originale : [5, 2, 5, 8, 2, 1, 8, 3, 5, 2]
2 Liste sans doublons : [5, 2, 8, 1, 3]
```

Exercice 4 : Gestionnaire de tâches simple

Créez un gestionnaire de tâches avec menu interactif :

```

1 --- GESTIONNAIRE DE TACHES ---
2 1. Ajouter une tache
3 2. Afficher toutes les taches
4 3. Marquer une tache comme terminee
5 4. Modifier une tache
6 5. Supprimer une tache
7 6. Quitter

```

Le programme doit :

- Utiliser une boucle `while` pour le menu
- Stocker les tâches dans une liste
- Utiliser `match...case` ou `if...elif` pour les choix
- Gérer les erreurs (choix invalide, indices hors limite)

Exercice 5 : Table de multiplication complète

Écrivez un programme qui affiche toutes les tables de multiplication de 1 à 10 sous forme de tableau.

Exemple d’affichage :

```

1 | 1  2  3  4  5  6  7  8  9  10
2 ---+---
3 1 | 1  2  3  4  5  6  7  8  9  10
4 2 | 2  4  6  8 10 12 14 16 18  20
5 3 | 3  6  9 12 15 18 21 24 27  30
6 ...
7 10 | 10 20 30 40 50 60 70 80 90 100

```

Exercice 6 : Transfert d’éléments entre listes

Créez un programme qui permet de remplir une liste d’entiers. L’arrêt de la saisie est conditionné par une réponse (o/n).

Le programme doit transférer tous les nombres parfaits, premiers et carrés de `liste_source` vers des listes de destination, puis afficher :

1. La liste source après transfert
2. Le listes destinations après transfert
3. Le nombre total d’éléments transférés pour chaque liste de destination

Attention : Manipulez les listes avec soin pour éviter les erreurs d’index.

Exercice 7 : Analyse de texte simple

Écrivez un programme qui :

1. Demande une phrase à l’utilisateur

2. Compte et affiche :

- Le nombre total de caractères
- Le nombre de mots
- Le nombre de voyelles (a, e, i, o, u, y)
- Si la phrase contient le mot "Python"

Exemple d'exécution :

```
1 Entrez une phrase : J'apprends le langage Python
2 Longueur : 25 caracteres
3 Mots : 4 mots
4 Voyelles : 8 voyelles
5 Contient 'Python' : Oui
```

Exercice 8 : Jeu du nombre mystère

Créez un jeu où l'ordinateur choisit un nombre entre 1 et 30 et l'utilisateur doit le deviner. Le programme doit :

- Donner des indices ("Trop grand", "Trop petit")
- Limiter à 5 tentatives maximum
- Afficher un message de félicitations ou d'échec
- Proposer de rejouer à la fin

Exemple d'exécution :

```
1 Devinez le nombre entre 1 et 20 :
2 Essai 1/5 : 10
3 Trop petit !
4 Essai 2/5 : 15
5 Trop grand !
6 Essai 3/5 : 13
7 Bravo ! Vous avez trouve en 3 essais.
8 Voulez-vous rejouer ? (o/n) : n
```

Bon courage et bonne programmation !