Chapitre 2 : Les bases du langage Python

Par Robert DIASSÉ

Objectifs du chapitre

À l'issue de ce chapitre, vous serez capable de :

- Comprendre la syntaxe de base de Python.
- Manipuler les variables et reconnaître leurs types.
- Effectuer des calculs avec les opérateurs arithmétiques et de comparaison.
- Utiliser les structures de contrôle simples (conditions).
- Écrire et comprendre des programmes simples grâce à des exemples détaillés.

Plan du chapitre

---

1. Variables et types de données

1.1 Qu'est-ce qu'une variable ?

Une variable est un espace mémoire qui stocke une valeur identifiable par un nom.
Exemple :

```
age = 25
nom = "Jean"
taille = 1.75
```

Explications

- `age = 25` : Variable contenant un entier (`int`).
- `nom = "Jean"` : Variable contenant une chaîne de caractères (`str`).

- `taille = 1.75` : Variable contenant un nombre décimal (`float`).

1.2 Types de données principaux

Python propose plusieurs types de données standards :

- `int` : Entiers (ex. : `10`, `-5`)
- `float` : Nombres décimaux (ex. : `3.14`, `-2.71`)
- `str` : Chaînes de caractères (ex. : `"Python"`)
- `bool` : Booléens (`True`, `False`)

Exemple :

```
nombre = 10          # int
prix = 19.99         # float
texte = "Python"     # str
est_vrai = True      # bool

print(type(nombre))  # <class 'int'>
print(type(prix))    # <class 'float'>
```

1.3 Les commentaires en Python

Les commentaires permettent de rendre un code plus lisible.

Commentaires monolignes

- Commencez la ligne avec `#`.

```
# Cette variable contient un prénom
nom = "Alice"
print(nom)  # Affiche le prénom
```

Commentaires multilignes

- Utilisez trois guillemets `'''` ou `"""`.

```
"""
Ce programme calcule la somme
de deux nombres.
"""
def addition(a, b):
    return a + b
```

---

2. Syntaxe de base

2.1 Les règles principales

1. Python est un langage **dynamique** : inutile de déclarer le type des variables.
2. L'**indentation** (tabulation ou 4 espaces) définit les blocs de code.

Exemple :

```
age = 20
if age >= 18:
    print("Vous êtes majeur.")
```

Explications :

- `age = 20` : Création d'une variable.
- `if age >= 18:` : Condition vérifiée.
- `print("Vous êtes majeur.")` : Instruction exécutée si la condition est vraie.

---

3. Les opérateurs

3.1 Opérateurs arithmétiques

- `+` : Addition
- `-` : Soustraction
- `*` : Multiplication
- `/` : Division classique
- `//` : Division entière
- `%` : Modulo (reste de la division)
- `**` : Puissance

Exemple :

```
a = 10
b = 3
print(a + b)  # 13
print(a // b) # 3
```

3.2 Opérateurs de comparaison

- `==` : Égal à
- `!=` : Différent de
- `<`, `<=` : Inférieur, inférieur ou égal
- `>`, `>=` : Supérieur, supérieur ou égal

Exemple :

```
a = 5
b = 8
print(a == b)  # False
print(a < b)   # True
```

3.3 Opérateurs logiques

- `and` : Vrai si **toutes** les conditions sont vraies.
- `or` : Vrai si **au moins une** condition est vraie.
- `not` : Inverse la valeur logique.

Exemple :

```
age = 25
est_étudiant = True

print(age > 18 and est_étudiant)  # True
print(age > 18 or not est_étudiant)  # True
```

4. Entrées utilisateur et conversions de type

4.1. La fonction `input()`

- La fonction `input()` permet de demander à l'utilisateur de saisir des données.
- Par défaut, les données saisies sont de type `str` (chaîne de caractères), même si l'utilisateur entre un nombre.

Exemple simple :

```
nom = input("Entrez votre nom : ")
print("Bonjour,", nom)
```

Explications ligne par ligne :

- `nom = input("Entrez votre nom : ")` : Affiche le message `"Entrez votre nom : "` et attend que l'utilisateur saisisse une donnée, qui sera stockée dans la variable `nom`.
- `print("Bonjour,", nom)` : Affiche un message qui inclut la saisie de l'utilisateur.

4.2. Conversions de type

Si vous avez besoin d'un type spécifique (par exemple, un entier ou un nombre à virgule flottante), il faut **convertir** la saisie grâce aux fonctions suivantes :

- `int()` : Convertit une chaîne de caractères en entier.
- `float()` : Convertit une chaîne de caractères en nombre décimal.
- `str()` : Convertit un entier ou un nombre en chaîne de caractères.

Exemple avec conversion :

```
age = input("Entrez votre âge : ")  # Saisie en chaîne de caractères
age = int(age)  # Conversion en entier
print("L'année prochaine, vous aurez", age + 1, "ans.")
```

Explications:

1. `age = input("Entrez votre âge : ")` : Attend que l'utilisateur entre un âge, qui sera traité comme une chaîne de caractères.
2. `age = int(age)` : Convertit la chaîne de caractères saisie en entier.
3. `print("L'année prochaine, vous aurez", age + 1, "ans.")` : Effectue une addition avec l'âge (maintenant entier) et affiche le résultat.

5. Exercices

Exercice 1 : Calcul simple

- Demandez à l'utilisateur d'entrer deux nombres, puis affichez leur somme, différence, produit, et division.

Exercice 3 : Conversion de type

- Demandez à l'utilisateur de saisir une valeur.
  - Affichez cette valeur en précisant son type initial (chaîne de caractères).
  - Convertissez-la en entier ou flottant (si possible) et affichez à nouveau son type après conversion.

Résumé du chapitre

- Vous savez créer des variables et manipuler différents types de données.
- Vous connaissez les opérateurs principaux (arithmétiques, de comparaison, logiques).
- Vous pouvez écrire des programmes Python simples et structurés.
