UNIVERSITÉ Larbi Ben Mhidi

Faculté des Sciences de la Terre, de la Géographie et de l'Aménagement du Territoire

Département Géographie et aménagement du territoire

Matière : Programmation

Niveau : 1ère année Licence— Semestre 2

Crédits : 2 | **Coefficient :** 1

Enseignant : Bezzaz Soumia

Année universitaire : 2025–2026

Variables, Types de Données et Opérateurs

PARTIE I — INTRODUCTION GÉNÉRALE

1.1. Qu'est-ce que la programmation ?

La **programmation** (ou codage) est l'activité qui consiste à rédiger un ensemble d'instructions — appelé programme ou code source — destinées à être exécutées par un ordinateur afin de réaliser une tâche précise.

Définition formelle :
Un programme informatique est une suite finie et ordonnée d'instructions écrites dans un langage de programmation, traduisant un algorithme en un format compréhensible et exécutable par une machine.

Un **algorithme**, quant à lui, est une description logique et structurée des étapes nécessaires à la résolution d'un problème donné, indépendamment de tout langage de programmation.

Relation entre algorithme et programme

Problème → Algorithme (logique) → Programme (code) → Exécution (résultat)

Concept	Nature	Exemple
Algorithme	Description abstraite	« Additionner deux nombres et afficher le résultat »
Programme	Code exécutable	print(3 + 5)

1.2. Pourquoi apprendre à programmer en Géosciences ?

Les géosciences modernes reposent de plus en plus sur le traitement numérique de données massives. La programmation permet aux géoscientifiques de :

- **Automatiser** le traitement de grandes quantités de données (sismiques, météorologiques, géochimiques, etc.) ;
- **Visualiser** des données géographiques et géologiques sous forme de cartes, graphiques et modèles 3D ;
- **Modéliser** des phénomènes naturels (propagation d'ondes sismiques, circulation atmosphérique, écoulement des eaux souterraines) ;
- **Analyser statistiquement** des jeux de données géologiques ;
- **Gérer des fichiers** et des bases de données de terrain.

Exemple concret : *Un géologue peut écrire un programme Python pour lire automatiquement des milliers de mesures de porosité à partir d'un fichier CSV, calculer la moyenne, l'écart-type, et produire un histogramme — en quelques secondes, au lieu de plusieurs heures de travail manuel sur un tableur.*

1.3. Les langages de programmation

Un **langage de programmation** est un ensemble de règles syntaxiques et sémantiques permettant de formuler des instructions compréhensibles par un ordinateur.

1.4. Pourquoi Python ?

Dans le cadre de ce cours, nous utiliserons principalement le langage **Python** pour les raisons suivantes :

1. **Simplicité syntaxique :** la syntaxe de Python est claire, lisible et proche du langage naturel anglais ;
2. **Polyvalence :** Python est utilisé en science des données, intelligence artificielle, géomatique, géophysique, etc. ;
3. **Richesse des bibliothèques :** NumPy (calcul numérique), Pandas (manipulation de données), Matplotlib (visualisation), SciPy (calcul scientifique), ObsPy (sismologie) ;
4. **Gratuité et open source :** Python est libre d'utilisation ;
5. **Communauté importante :** une documentation abondante et un support communautaire très actif.

Premier programme : « Hello, World ! »

Par tradition, le premier programme que l'on écrit dans tout langage de programmation affiche le message « Hello, World ! » (Bonjour, le monde !).

```
# Mon premier programme en Python

print("Hello, World !")

print("Bienvenue en Programmation - Géosciences L1")
```

Sortie :

```
Hello, World !

Bienvenue en Programmation - Géosciences L1
```

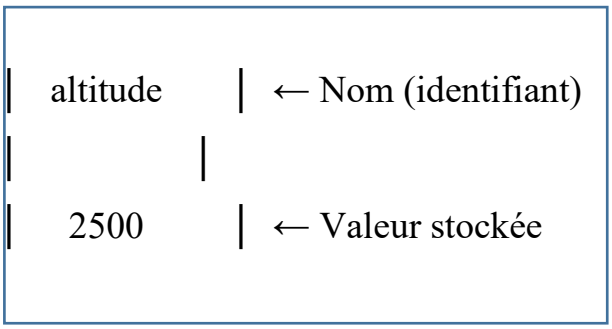
Remarque : *La fonction print() permet d'afficher du texte ou des valeurs à l'écran. Le symbole # introduit un* ***commentaire*** *: texte ignoré par l'interpréteur, destiné au lecteur humain.*

PARTIE II — LES VARIABLES

2.1. Notion de variable

En programmation, une variable est un espace mémoire identifié par un nom (identifiant) dans lequel on peut stocker une valeur qui peut évoluer au cours de l'exécution du programme.

Une variable est comparable à une boîte étiquetée. L'étiquette est le nom *de la variable, et le contenu de la boîte est sa* valeur.



2.2. Déclaration et affectation

En Python, la création d'une variable se fait par affectation à l'aide de l'opérateur **= :**

```
# Déclaration et affectation de variables
altitude = 2500          # Variable de type entier
temperature = 15.3       # Variable de type décimal (flottant)
nom_station = "Oum ElBouaghi"  # Variable de type chaîne de caractères
est_active = True        # Variable de type booléen
```

Attention : *L'opérateur = en programmation signifie* ***« affecter »*** *(assigner une valeur), et non « égal à » au sens mathématique. Le test d'égalité s'écrit ==.*

Affectation multiple

Python permet d'affecter plusieurs variables simultanément :

```
# Affectation multiple
x, y, z = 10, 20, 30
print(x)  # Affiche : 10
print(y)  # Affiche : 20
print(z)  # Affiche : 30
```

Réaffectation (modification de la valeur)

La valeur d'une variable peut être modifiée à tout moment :

```
temperature = 25.0
print(temperature)   # Affiche : 25.0
```

2.3. Règles de nommage des variables

Le choix du nom d'une variable en Python obéit à des règles strictes et à des conventions :

Règles obligatoires (syntaxe)

Règle	**Exemple valide**	**Exemple invalide**
Commence par une lettre ou _	altitude, _temp	2altitude
Contient uniquement lettres, chiffres et _	station_1	station-1, station 1
Ne doit pas être un mot réservé Python	vitesse	print, for, if
Sensible à la casse (majuscules ≠ minuscules)	Temp ≠ temp ≠ TEMP	—

PARTIE III — LES TYPES DE DONNÉES

3.1. Les types fondamentaux

Chaque variable en Python possède un **type** qui détermine la nature des données qu'elle contient et les opérations applicables.

Type	Mot-clé Python	Description	Exemple
Entier	int	Nombre entier (positif ou négatif)	42, -7, 0
Flottant	float	Nombre à virgule flottante (décimal)	3.14, -0.5, 2.0
Chaîne de caractères	str	Texte entre guillemets	"Bonjour", 'Alger'
Booléen	bool	Valeur logique (vrai/faux)	True, False
Aucun	NoneType	Absence de valeur	None

3.2. Le type entier (int)

Les entiers représentent des nombres sans partie décimale. En Python, leur taille n'est pas limitée.

```
nombre_echantillons = 150

profondeur = -340

annee = 2024

print(type(nombre_echantillons))  # <class 'int'>
```

Remarque : *La fonction type() permet de connaître le type d'une variable.*

3.3. Le type flottant (float)

Les flottants représentent des nombres réels (avec une partie décimale).

```
latitude = 36.7538       # Latitude d'Alger

print(type(latitude))     # <class 'float'>
```

Notation scientifique

```
distance_soleil = 1.496e8    # 1.496 × 10⁸ km
```

3.4. Le type chaîne de caractères (str)

Une chaîne de caractères est une séquence de caractères encadrée par des guillemets simples ('...') ou doubles ("...").

mineral = "Quartz"

description = "C'est un minéral très courant"

3.5. Le type booléen (bool)

Le type booléen ne peut prendre que deux valeurs : True (vrai) ou False (faux). Il est fondamental pour les tests conditionnels.

est_sedimentaire = True

est_volcanique = False

Les booléens résultent souvent de comparaisons

print(5 > 3) # True

print(10 == 20) # False

***La fonction** input() **:** Elle permet de demander à l'utilisateur de saisir une valeur au clavier. La valeur retournée est toujours de type str, même si l'utilisateur entre un nombre. Il faut donc effectuer une conversion explicite.*

PARTIE IV — LES OPÉRATEURS

4.1. Opérateurs arithmétiques

Les opérateurs arithmétiques permettent d'effectuer des calculs mathématiques.

Opérateur	Signification	Exemple	Résultat
+	Addition	7 + 3	10
-	Soustraction	7 - 3	4
*	Multiplication	7 * 3	21
/	Division (réelle)	7 / 3	2.3333...
//	Division entière	7 // 3	2

Opérateur	Signification	Exemple	Résultat
%	Modulo (reste de la division)	7 % 3	1
**	Puissance (exponentiation)	7 ** 3	343

Exemples

```
# Calcul de la vitesse d'une onde sismique
distance = 450.0    # km
temps = 75.0        # secondes
vitesse = distance / temps

# Conversion de température (°F → °C)
temp_fahrenheit = 98.6
temp_celsius = (temp_fahrenheit - 32) * 5 / 9
print(f"Température : {temp_celsius:.2f} °C")  # 37.00 °C

# Calcul de la pression hydrostatique : P = ρ × g × h
rho = 1025      # Masse volumique de l'eau de mer (kg/m³)
g = 9.81        # Accélération gravitationnelle (m/s²)
h = 200         # Profondeur (m)
pression = rho * g * h
print(f"Pression à {h} m : {pression:.0f} Pa")  # 2 011 050 Pa
```

4.2. Opérateurs de comparaison

Les opérateurs de comparaison retournent un **booléen** (True ou False).

Opérateur	Signification	Exemple	Résultat
==	Égal à	5 == 5	True
!=	Différent de	5 != 3	True
>	Strictement supérieur à	5 > 3	True

Opérateur	Signification	Exemple	Résultat
<	Strictement inférieur à	5 < 3	False
>=	Supérieur ou égal à	5 >= 5	True
<=	Inférieur ou égal à	5 <= 3	False

magnitude = 6.5

print(magnitude > 5.0) # True → séisme significatif

print(magnitude == 7.0) # False

print(magnitude != 0) # True

print(magnitude >= 6.0) # True

4.3. Opérateurs logiques

Les opérateurs logiques permettent de combiner plusieurs conditions booléennes.

Opérateur	Signification	Exemple	Résultat
and	ET logique (les deux conditions doivent être vraies)	True and False	False
or	OU logique (au moins une condition vraie)	True or False	True
not	NON logique (inverse la valeur)	not True	False

Tables de vérité

Opérateur and **:**

A	**B**	**A** and **B**
True	True	True
True	False	False
False	True	False
False	False	False

Opérateur or **:**

A	**B**	**A** or **B**
True	True	True
True	False	True
False	True	True
False	False	False

PARTIE V — LA FONCTION input() **ET L'INTERACTION UTILISATEUR**

La fonction input() permet de créer des programmes interactifs.

```
nom = input("Entrez votre nom : ")
age = int(input("Entrez votre âge : "))
ville = input("Entrez votre ville : ")
print(f"Bonjour {nom}, vous avez {age} ans et vous habitez à {ville}.")
```

EXERCICES

Exercice 1 — Variables et types

Déterminez le type de chaque variable sans exécuter le code, puis vérifiez avec type() :

```
a = 42
b = 3.14
c = "Géologie"
d = True
e = 7 / 2
f = 7 // 2
g = str(100)
h = int("50")
```

Exercice 2 —

Écrire un programme qui :

1. Demande à l'utilisateur la distance épicentrale (en km) et le temps d'arrivée de l'onde P (en secondes) ;
2. Calcule la vitesse de propagation de l'onde P ;
3. Affiche le résultat avec un message formaté.

Solution

Rappel de la formule $v=d/t$

- d = distance épicentrale (km)
- t = temps d'arrivée (s)
- v = vitesse de l'onde P (km/s)

Programme Python

```
# Calcul de la vitesse de l'onde P
distance = float(input("Distance épicentrale (km) : "))
temps = float(input("Temps d'arrivée (s) : "))
vitesse = distance / temps
print("La vitesse de l'onde P est :", vitesse, "km/s")
```

Explication simple

On lit la distance. - On lit le temps. - On applique la formule :

vitesse = distance / temps - On affiche le résultat.

Exercice 3 — Expressions booléennes

Sans exécuter le code, déterminez le résultat de chaque expression :

```
print(10 > 5 and 3 < 1)
print(10 > 5 or 3 < 1)
print(not (5 == 5))
print(7 != 7 or 4 <= 4)
print(not False and True)
print((3 ** 2) > 8 and (10 % 3) == 1)
```