

Introduction rapide à Python

Objectif :

- Découvrir les bases de Python nécessaires pour réaliser le TP1.

Plan

- 1) Introduction
- 2) Premier programme
- 3) Variable
- 4) Interaction utilisateur
- 5) Affichage
- 6) Structure conditionnelle
- 7) Indentation

Qu'est-ce que Python ?

- Un langage de programmation **généraliste**, **interprété**, et de **haut niveau**
- Conçu pour être **simple** à apprendre et facile à lire grâce à **une syntaxe claire**
- Utilisé dans de nombreux domaines :
 - Intelligence artificielle (très utilisé)
 - Analyse de données et Data Science
 - Développement d'applications desktop
 - Développement web (ex. Django, Flask)
 - Automatisation des tâches
- Créé en 1989 par Guido van Rossum, aux Pays-Bas
- Première version : publiée en 1991
- Dernière version: Version 3 (Version stable : Python 3.14 (oct. 2025))
- <https://www.python.org/psf/>

Caractéristiques intéressantes

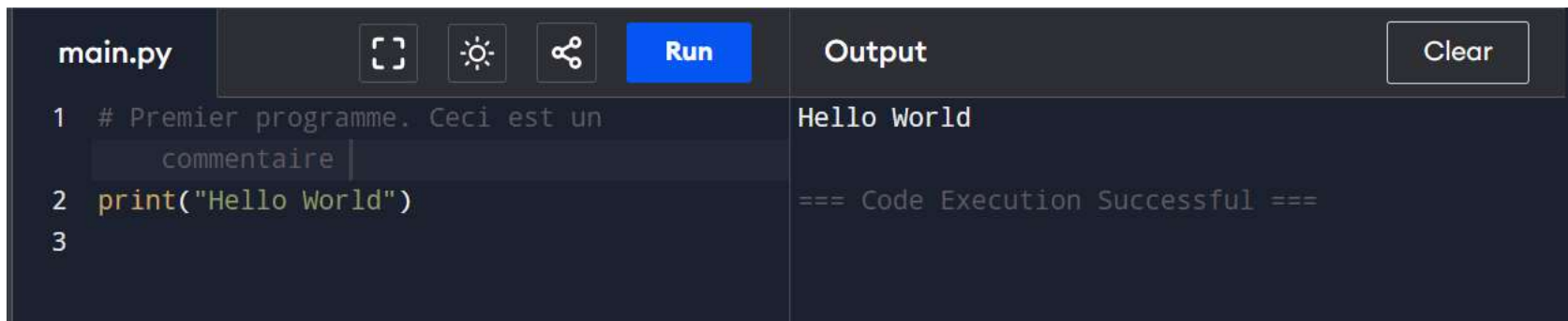
- Multiplateforme: fonctionne sur de nombreux systèmes d'exploitation : Windows, Mac OS, Linux, Android, iOS, etc
- Gratuit: Vous pouvez l'installer sur autant d'ordinateurs que vous voulez (même sur votre téléphone !)
- De haut niveau: Il demande relativement peu de connaissance sur le fonctionnement d'un ordinateur pour être utilisé
- Un langage interprété: Un **script** Python n'a pas besoin d'être compilé pour être exécuté, contrairement à des langages comme le C ou le C++
- Orienté objet
- Relativement simple à prendre en main
- Langage de programmation très utilisé (voir les classements TIOBE ¹ et IEEE Spectrum ²)
- Grande communauté : Support et ressources abondantes pour résoudre des problèmes.

1. <https://www.tiobe.com/tiobe-index/>

2. <https://spectrum.ieee.org/the-top-programming-languages-2023>

Premier programme

```
# Premier programme. Ceci est un commentaire  
print("Hello World")
```



main.py	Run	Output
<pre>1 # Premier programme. Ceci est un commentaire 2 print("Hello World") 3</pre>		<pre>Hello World === Code Execution Successful ===</pre>

- Pas de main()
- Pas de compilation
- Exécution directe
- Pont virgule « ; » n'est pas nécessaire
- Interpréteur en ligne: <https://www.programiz.com/python-programming/online-compiler/>

Variable

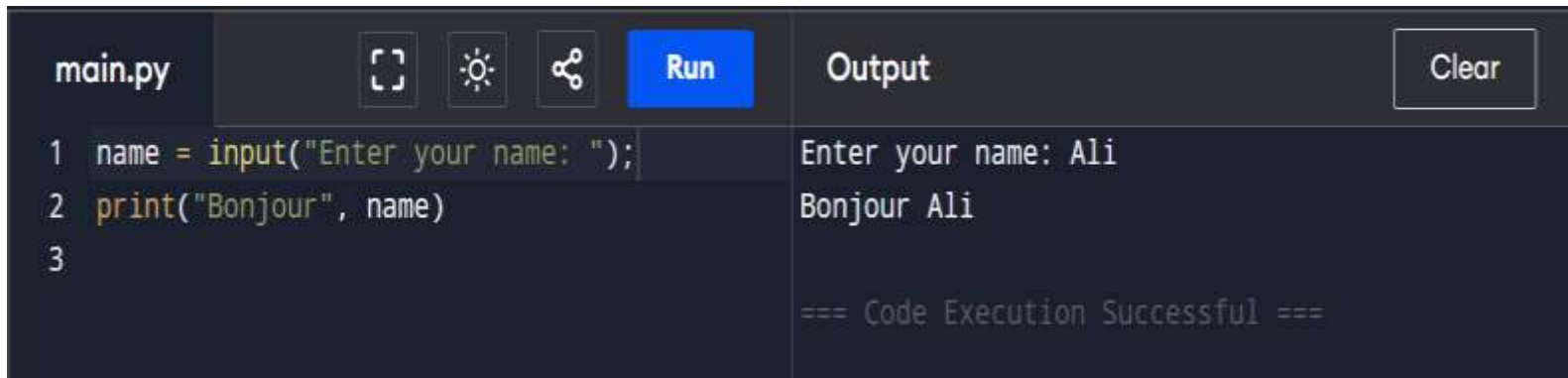
- La **déclaration** et l'**initialisation** se font en même temps
- Pas de déclaration de type
- Types **dynamiques** :
 - Le type d'une variable est déterminé automatiquement au moment de l'exécution
 - En C : typage statique

```
x = 10          # Variable entière
y = 3.14        # Variable flottante
z = "Python"    # Chaîne de caractères
```

- Le symbole = est l'opérateur d'affectation (== est l'opérateur de comparaison)
- Nommage
 - Règles habituelles
 - Python est sensible à la **casse**, ce qui signifie que les variables Test, test et TEST sont différentes

Interaction utilisateur

- La fonction `input()`
 - Permet de lire une entrée saisie par l'utilisateur au clavier
 - Retourne une chaîne de caractères (string)



The screenshot shows a Python IDE interface. On the left, a code editor window titled 'main.py' contains the following code:

```
1 name = input("Enter your name: ");
2 print("Bonjour", name)
3
```

On the right, an 'Output' window displays the execution results:

```
Enter your name: Ali
Bonjour Ali

=== Code Execution Successful ===
```

The IDE also features a toolbar with icons for file operations, a 'Run' button, and a 'Clear' button for the output window.

- Si on veut un entier :

```
age = int(input("Enter your age: "))
```

Affichage

- La fonction print()

```
print("Hello", name)
```

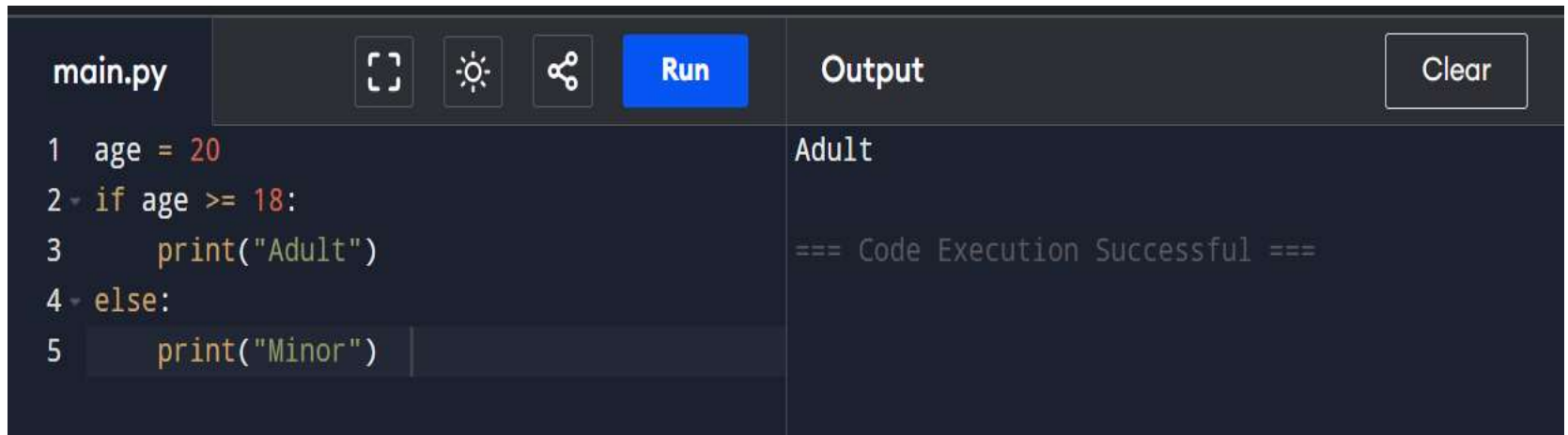
- On peut afficher plusieurs éléments :

main.py    	Output Clear
<pre>1 age = 20 2 print("You are", age, "years old.") 3</pre>	<pre>You are 20 years old. === Code Execution Successful ===</pre>

Structure conditionnelle

- Syntaxe

```
if condition:
    instruction
elif autre_condition:
    instruction
else:
    instruction
```



The screenshot shows a code editor interface with a dark theme. On the left, a file named 'main.py' is open. The code contains a conditional statement that checks if 'age' is greater than or equal to 18. Since the age is 20, the 'if' branch is executed, printing 'Adult'. The 'Run' button is highlighted in blue. On the right, the 'Output' panel shows the result 'Adult' and a success message '=== Code Execution Successful ==='. There are also icons for running, debugging, and sharing code.

```
main.py
```

```
1 age = 20
2 if age >= 18:
3     print("Adult")
4 else:
5     print("Minor")
```

Run

Output

Adult

=== Code Execution Successful ===

Clear

bloc d'instructions et Indentation

- Indentation

- Point **essentiel**
- utilisé **obligatoirement** pour délimiter les blocs de code
- Pas d'accolades

```
# Python
age = 20
if age > 18:
    print("Adult")
    print("You can vote.")
print("End of the program.")
```

```
//C
if (age > 18) {
    printf("Adult\n");
    printf("You can vote.\n");
}
printf("End of the program.\n");
```

bloc d'instructions et Indentation (2)

- Erreur courante
 - Ne pas indenter
 - ou utiliser une mauvaise indentation génère une erreur

```
if True:  
print("Erreur car pas d'indentation")
```

- Pratiquement,
 - l'indentation doit être **homogène** (soit des espaces, soit des tabulations, mais pas un mélange des deux).
 - Une indentation avec 4 espaces est le style d'indentation recommandé
- Une mauvaise indentation change le comportement du programme.