Lecture Three: Introduction to the Python Language

Python is considered one of the most widely used programming languages in the world today, owing to its simplicity and ease of learning, which makes it an ideal choice for both beginners and researchers alike. In this lecture, we will begin by exploring the nature of the Python language and its areas of application, before moving on to how to set up an appropriate working environment to run it, whether on a computer or a mobile device. We will then examine the concept of a "project" in programming, understood as the framework that brings together program files and their environment. Finally, we will conclude with practical steps for creating and running a Python project using tools such as PyCharm or Pydroid, thereby opening the door to real-world applications and future projects.

First: What is Python?

Among high-level programming languages, Python stands out as a modern, flexible, and easy-to-learn language, making it an ideal choice for students outside the field of computer science, particularly in the social sciences and humanities. Python is distinguished by its simple syntax, readability, and efficiency in handling and analyzing data. It encourages logical thinking and facilitates the translation of abstract concepts into practical instructions, making it a powerful tool for understanding social phenomena through modeling, analysis, and simulation.

The language first appeared in 1991, created by Dutch programmer Guido van Rossum, who sought to develop a language that was readable, flexible, and conducive to logical reasoning. Its name was inspired by the British comedy group *Monty Python*, rather than the animal, reflecting a non-traditional spirit in its design.

Since its initial release, Python has evolved through several major versions. Python 2.x, which lasted for more than a decade, faced limitations in compatibility and updates. The major breakthrough came with Python 3.x, launched in 2008, which introduced significant improvements in text handling, data processing, and overall language structure, though it was not backward compatible. Today, Python is widely used in education, scientific research, application development, and data analysis. It is supported by an active global community and extensive libraries such as NumPy, Pandas, and Matplotlib, making it an ideal tool for students and researchers in the social sciences and humanities.

Second: Preparing Python

Before delving into the technical concepts of programming, it is essential to set up a working environment that allows for practical experimentation. Python, chosen for this course, is among the easiest languages to install and use, and it is freely available across multiple platforms: computers (Windows, macOS, Linux) as well as smartphones through dedicated applications.

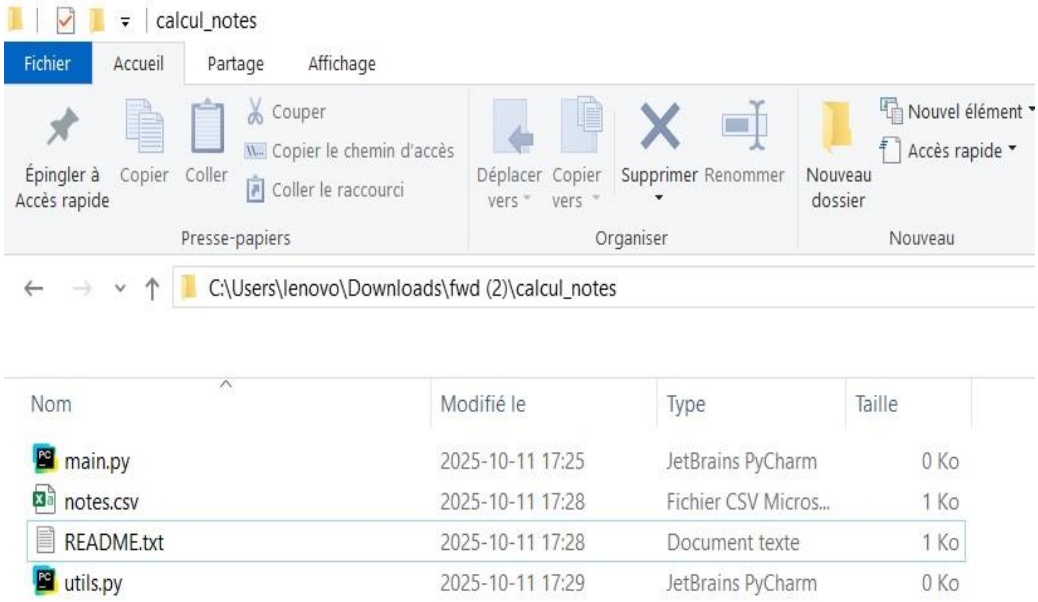
To install Python on a computer, one can visit the official website python.org and download the appropriate version, with the latest release in the Python 3.x series generally recommended. After

installation, code can be written directly using a simple text editor such as Notepad, though it is preferable to use an integrated development environment (IDE) such as PyCharm, VS Code, or Jupyter Notebook. These environments allow for better organization of code, step-by-step execution, and immediate visualization of results.

On smartphones, applications such as Pydroid 3 (for Android) or Pythonista (for iOS) provide integrated environments for writing and running code without the need for a computer. Additionally, Python projects can be developed on cloud-based platforms such as Google Colab (colab.research.google.com in Bing), which enable writing and executing Python code directly from a web browser without requiring any installation on a local device.

Third: What is a Project?

In programming, a project is an organizational unit that encompasses a set of files and folders working together to create a program, application, or complete software system. In Python, for example, a project is represented as a directory containing Python files, images, data, and sometimes configuration instructions—all functioning together to achieve a specific goal, such as



From the illustration, it is clear that the project folder **`calcul_notes`** contains a set of essential files designed by the programmer to ensure the program functions properly. These files are as follows:

```
calcul_notes/
│
├── main.py         ← The main file containing the core code
├── notes.csv       ← A file storing students' data
├── README.txt      ← A brief guide explaining how to use the program
└── utils.py        ← Auxiliary functions (e.g., calculating averages)
```

In programming languages such as Python, the term *project* is used because:

- It contains multiple components that work together.
- It can be further developed and expanded over time.
- It can be easily shared or executed on other devices.

From this perspective, if we only have a single file (e.g., `main.py`), it is merely a small program. However, once we begin adding other files, datasets, images, or libraries, it evolves into a *project*.

Fourth: Steps to Create and Run a Python Project

The process of creating a new Python program varies depending on the integrated development environment (IDE). In this lecture series, we will use **PyCharm** (for Windows) and **Pydroid** (for Android).

A. Creating a Project in PyCharm

1. **Open PyCharm**
   - Launch the PyCharm application on your computer.
2. **Create a New Project**
   - From the main screen, select *New Project*.
   - Specify the location where the project will be saved.
   - Choose Python as the project language and ensure that a suitable interpreter is selected.
   - Click *Create*.
3. **Add a New Python File**
   - In the project window (on the left), right-click on the project folder.
   - Select *New → Python File*.
   - Enter the file name (e.g., `main.py`) and press Enter.
4. **Write the Code**
   - Inside the new file, you can now write Python code.
   - Example:
   - `print("Welcome to Python!")`
5. **Run the Program**
   - Right-click inside the file and select *Run 'filename'*.
   - The output will appear in the *Run* window at the bottom.

B. Creating a Program in Pydroid (Android)

Unlike PyCharm, Pydroid does not use the concept of a *project*. Instead, it works with independent `.py` files. This makes it excellent for experimentation and quick learning on mobile devices, though it is less suitable for complex or multi-file projects.

Steps:

1. **Open the Application** on your phone.
2. From the home screen, tap *Editor*.
3. In the editor, start writing Python code directly.
4. **Save the File**:
    - Tap the ⋮ (menu) at the top.
    - Select *Save As*.
    - Enter the file name (e.g., `main.py`) and choose the save location.
5. **Run the Code**:
    - Tap the ▶ (*Run*) button at the top.