Introduction

Kittikun Jitpairod

What is Python?

Python is a high-level, interpreted programming language created by Guido van Rossum and first released in 1991. It's designed to be easy to read and write, with a clean and straightforward syntax that emphasizes code readability.

Why Python is Popular

1. **Readability**: Its clear, readable syntax makes it easy to learn and maintain.
2. **Versatility**: Python is used in web development, data science, AI, machine learning, automation, and more.
3. **Large Standard Library**: Python comes with a comprehensive standard library, reducing the need for external modules.
4. **Strong Community**: A large, active community contributes to Python's ecosystem with numerous third-party packages.
5. **Cross-platform**: Python runs on various platforms (Windows, macOS, Linux).
6. **Interpreted Language**: No compilation step is needed, speeding up the development process.
7. **Integration**: Python can be easily integrated with other languages like C and C++.

Basic Concepts of Python Programming

1. **Syntax**: Python uses indentation to define code blocks, making it visually clear and enforcing clean code structure.
2. **Variables**: Python is dynamically typed, meaning you don't need to declare variable types explicitly.
3. **Data Types**: Python has several built-in data types, including:
    - Numeric types (int, float, complex)

- Sequence types (list, tuple, range)
- Text type (str)
- Mapping type (dict)
- Set types (set, frozenset)
- Boolean type (bool)

4. **Control Structures**: Python supports standard control structures like if-else statements, for and while loops.
5. **Functions**: Defined using the `def` keyword, functions in Python can have default arguments and return multiple values.

Getting Started with Writing Python Programs

Python is a beginner-friendly programming language that is widely used for a variety of applications, from web development to data analysis and artificial intelligence. Here's a guide to help you begin writing Python programs:

1. Install Python

- **Check Installation**: First, check if Python is already installed on your computer. Open a terminal (Command Prompt on Windows, Terminal on macOS/Linux) and type:

  ```
  python --version
  ```

  or

  ```
  python3 --version
  ```

- **Download and Install**: If Python isn't installed, download it from python.org and follow the installation instructions. During installation on Windows, ensure you check the box to "Add Python to PATH."

2. Choose an Editor or Integrated Development Environment (IDE)

- **Text Editors**: You can start writing Python code in a simple text editor like Notepad (Windows), TextEdit (macOS), or Gedit (Linux).
- **Code Editors**: For a better experience, consider using code editors like Visual Studio Code, Sublime Text, or Atom. These provide features like syntax highlighting and code suggestions.
- **IDEs**: If you want more advanced tools, consider using an IDE like PyCharm or Spyder, which provide integrated debugging, project management, and other features.

3. Write Your First Python Program

- Open your chosen text editor or IDE and create a new file.
- Start with the most basic Python program, the "Hello, World!" program. Type the following code:

  ```
  print("Hello, World!")
  ```

- Save the file with a `.py` extension, for example, `hello.py`.

4. Run Your Python Program

- **Using Terminal or Command Prompt**:
  - Open a terminal or command prompt.
  - Navigate to the directory where you saved your Python file using the `cd` command:

    ```
    cd path/to/your/file
    ```

  - Run your Python program by typing:

    ```
    python hello.py
    ```

    or, if `python3` is required:

    ```
    python3 hello.py
    ```

- You should see the output `Hello, World!` printed in the terminal.

Python Integrated Development Environment (IDE)

IDE is a software application that provides comprehensive facilities to computer programmers for software development with Python. IDEs typically consist of a source code editor, build automation tools, and a debugger. They offer various features that make writing, testing, and debugging Python code more efficient and convenient. Here's an overview of some common aspects of Python IDEs:

Key Features of Python IDEs:

1. **Code Editor**:
   - A powerful code editor with syntax highlighting, code completion, and formatting helps you write Python code more easily and accurately.
   - Many IDEs also support linting, which checks your code for errors and potential issues as you type.

2. **Integrated Debugger**:
   - IDEs come with built-in debugging tools that allow you to set breakpoints, step through code, inspect variables, and understand the flow of your program.
   - This helps in identifying and fixing bugs more efficiently than using print statements.
3. **Code Navigation**:
   - Features like go-to definition, find references, and code folding make navigating large codebases easier.
   - These tools help you quickly locate functions, classes, and variables, improving your productivity.
4. **Project Management**:
   - IDEs usually include tools for managing projects, such as organizing files, dependencies, and virtual environments.
   - They often have built-in support for version control systems like Git, allowing you to manage your code's history and collaborate with others.
5. **Integration with Tools**:
   - IDEs can integrate with other tools commonly used in Python development, such as testing frameworks (e.g., PyTest, Unittest), package managers (e.g., pip, Conda), and databases.
   - They often support plugins or extensions, allowing you to add more functionality to your IDE.
6. **Interactive Console**:
   - Many Python IDEs include an interactive Python console, which allows you to run Python commands and see results immediately, making it useful for testing small snippets of code or experimenting with new ideas.
7. **Cross-Platform Support**:
   - Most Python IDEs are cross-platform, meaning they can run on various operating systems like Windows, macOS, and Linux.

Popular Python IDEs:

Offline

- Visual Studio Code
- PyCharm
- Jupyter

Online

- Colab
- replit
- Zed
- Keggle

IPython Notebook (`.ipynb`)

IPython Notebook, now more commonly known as **Jupyter Notebook**, is an open-source web application that allows you to create and share documents containing live code, equations, visualizations, and narrative text. It's particularly popular among data scientists, engineers, researchers, and educators for a variety of reasons:

Key Features:

1. **Interactive Coding**: You can write and execute code in multiple programming languages (most commonly Python) within the same document.
2. **Rich Text Support**: In addition to code, you can include Markdown (a lightweight markup language) to add headings, bullet points, hyperlinks, images, etc.
3. **Visualization**: It supports the display of charts and graphs generated by libraries like Matplotlib, Seaborn, Plotly, etc., directly within the notebook.
4. **Data Integration**: Jupyter Notebooks can handle data processing, cleaning, and exploration tasks, making them ideal for data analysis and machine learning.
5. **Notebooks as Documents**: They can be saved and shared as `.ipynb` files, making it easy to share your work with others, including code, results, and explanations.
6. **Extensions and Plugins**: The platform can be extended with various plugins and widgets, allowing for a customized and powerful environment.

Common Uses:

- **Data Science and Machine Learning**: For experimenting with data, training models, and visualizing results.
- **Education**: As a teaching tool to demonstrate concepts and walk students through programming exercises.
- **Research**: For creating and sharing computational experiments, including detailed documentation.

How It Works:

Jupyter Notebook runs in a browser, but the code is executed on a server, which can be your local machine or a remote server. This allows you to keep all your work in a single document, making it easier to reproduce and share your work.