Module 1: Using Python to Interact with the Operating System

🐍 Introduction to Python

Installing Python

- Learn how to install Python on your operating system.
- Verify the installation using `python --version`.

Features of Python

- Easy to read and write.
- Cross-platform compatibility (Windows, Linux, macOS).
- Vast standard library.
- Supports both procedural and object-oriented programming.

💻 Understanding the Operating System

What is an Operating System?

An **Operating System (OS)** is system software that manages computer hardware and software resources. It acts as a bridge between the user and hardware, providing a platform for applications to run.

Functions of an OS:

- Reads, writes, and deletes files from storage.

- Manages process creation and communication.
- Allocates memory to running programs.
- Handles network data transmission and reception.

Parts of an Operating System

1. **Kernel:** The core component that interacts directly with the hardware and manages system resources.
2. **User Space:** The environment where users and applications run. It interacts with the kernel through **system calls**.

How the OS Works

Example: Opening **Google Chrome**

1. User clicks on Chrome → action happens in **User Space**.
2. Chrome requests system resources (internet, memory, display) via **System Calls**.
3. The **Kernel** communicates with hardware components:
    - Network card → send/receive data.
    - CPU/RAM → process and store data.
    - GPU/Display → show webpage.
4. Kernel sends results back to Chrome in **User Space**, displaying the webpage.

🧩 Open Source Software

Open Source Software is software that can be freely shared, modified, and distributed.

Examples of Linux distributions (flavors):

- Ubuntu
- Debian

🌍 Cross-Platform and Portability

Cross-Platform Software: Works on multiple operating systems without major changes.

- Example: Python runs on Windows, Linux, and macOS.

Portable Code: Can run on different systems without modification as long as Python is installed.

🧠 IDE (Integrated Development Environment)

An **IDE** helps write and manage code efficiently.

Common Features:

- Syntax highlighting.
- Code completion.
- Debugging tools.
- Virtual environment management.

🧱 Virtual Environments in Python

A **Virtual Environment** is an isolated Python setup that allows different projects to use different dependencies.

Commands:

```
python -m venv myenv      # Create a virtual environment
```

```
myenv\Scripts\activate     # Activate on Windows
pip freeze                 # Show installed packages
```

⚙️ How Python Executes Code

1. **Create a File:** Example: `hello.py`
2. **Run the File:** Execute using `python hello.py`
3. **Interpreter Starts:** Python reads and translates the file.
4. **Compilation to Bytecode:** Creates `.pyc` files in the `__pycache__` folder. It's optimized instructions for the Python virtual machine.
5. **Execution:** The **Python Virtual Machine (PVM)** runs the bytecode line by line. It tells your computer's CPU what to do (Print, calculate, etc).

🧱 Compiler, Interpreter, and Transpiler

Type	Description	Example
Compiler	Translates the entire code into machine code before execution.	C, C++
Interpreter	Executes code line by line at runtime.	Python, JavaScript
Transpiler	Converts code from one language to another.	TypeScript → JavaScript

---

🚀 Scalability

Scalability means a system can handle increased workload efficiently. As work increases, the system can expand resources or processes to manage it smoothly.

🤖 Automation and Worthwhileness

Automation saves time and reduces human error.

Formula:

Time to Automate < Time to Perform Manually × Number of Times Done

Automation is worthwhile if building the automation takes less total time than performing the task manually over time.

✅ Summary

- Python is a cross-platform, open-source language.
- Operating Systems manage hardware and software via Kernel and User Space.
- Python uses virtual environments to isolate dependencies.
- Understanding compilers, interpreters, and transpilers helps grasp code execution.
- Automation improves productivity and reduces repetitive workload.

📘 *End of Module 1: Using Python to Interact with the Operating System*