**Uses / Applications of Python**

1. **Data Analysis** – Analyze and interpret complex data sets using libraries like Pandas, NumPy, and SciPy.
2. **Automation Testing** – Automate repetitive tasks and testing using frameworks like Selenium and PyTest.
3. **Data Cleaning** – Clean, preprocess, and transform raw data using tools like Pandas.
4. **REST APIs** – Build RESTful web services using Flask, FastAPI, or Django REST Framework.
5. **Data Visualization** – Represent data graphically using libraries like Matplotlib and Seaborn.
6. **Graph Plotting**
   * **Matplotlib** – Core plotting library, often used in combination with Pandas.
   * **Seaborn** – Based on Matplotlib, used for statistical plots with enhanced visuals.
7. **Data Cleaning and Computation** –
   * Use **Pandas** for handling missing data, filtering rows, transforming columns, etc.
8. **Mathematical Computations** –
   * Use **NumPy** for high-performance multidimensional array operations and complex mathematical calculations.

**Basic Python Example**

python

CopyEdit

x = 14

y = 25

* **Debugger**: A tool used to examine code execution line-by-line.
  + Helps verify input values like x and y.
  + Useful for identifying logical or runtime errors.

**What is Python?**

* **Python** is a **high-level, interpreted scripting language** that is easy to learn and widely used in web development, automation, data science, and more.

**Compiler vs. Interpreter**

| **Feature** | **Compiler** | **Interpreter** |
| --- | --- | --- |
| Definition | Translates entire code before execution | Translates code line-by-line during execution |
| Output | Generates a machine-level or bytecode file | Executes code directly, no separate output file |
| Example | C, C++, Java (uses both compiler & interpreter) | Python, Ruby, Perl |
| Error Handling | Detects all errors before execution | Stops at the first runtime error |

**Additional Notes:**

* **Python File**: .py (source)
* **Compiled Python File**: .pyc
  + Generated when a script is run and cached for faster execution
  + Even incorrect code may generate a .pyc file if the interpreter doesn't reach the faulty line
* **Java**:
  + Uses both compiler and interpreter
  + .java → compiled into .class bytecode → executed by JVM (Java Virtual Machine)

**IDEs for Python Development**

* **VS Code** – Lightweight and customizable IDE with Python extension support
* **PyCharm** – Full-featured Python IDE with advanced features for debugging, testing, and refactoring

**\_\_init\_\_.py File**

* **Purpose**:
  + Marks a directory as a Python package
  + Allows the import of modules from the package
  + Initializes variables or configurations when the package is imported

Kwlist -> keyword list

Identifiers -> variables like “x”, “y”. Names given by programmer for packages, files, variables, class.

Always ensure identifiers are in small case. If there are two words, use underscores eg. “first\_class”.

Class name -> AreaOfCirle.

**You cannot use keywords as an Identifier i.e as a name for class, variable, function etc. Because it confuses the compiler while running the code.**

**Python Formatters and Code Style Tools**

* **Black** – Uncompromising Python code formatter
* **isort** – Automatically sorts Python import statements
* **Flake8** – Tool for checking Python code against style (PEP8), programming errors, and complexity
* How to activate Formatter? Press Ctrl + Y to format the code
* A screenshot of a computer

  AI-generated content may be incorrect.

How to use the formatters using compiler?

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.