Python

Python is a dynamically typed, high-level programming language known for its simplicity and readability. Developed by **Guido van Rossum** in 1991, Python emphasizes code readability with an easy-to-learn syntax, making it popular among beginners and experienced developers alike.

Key Concepts in Python

1. **Interpreted & Dynamically Typed** – Python is an interpreted language, meaning code is executed line by line without prior compilation. It is dynamically typed, so variables do not require explicit type declarations.
2. **Object-Oriented & Functional Programming** – Python supports both OOP (classes, objects, inheritance, and polymorphism) and functional programming paradigms (lambda functions, higher-order functions).
3. **Memory Management** – Python uses automatic memory management and **garbage collection** to handle unused objects, optimizing memory usage.
4. **Exception Handling** – Python includes a robust exception-handling mechanism using **try, except, finally**, and **raise** for error management.
5. **Python Standard Library** – Python comes with a vast collection of built-in modules, covering areas like:
    - File I/O (os, shutil)
    - Data processing (csv, json)
    - Networking (socket, http.server)
    - Mathematics (math, random, decimal)
6. **Python Ecosystem & Libraries** –
    - **Data Science & Machine Learning**: NumPy, Pandas, Matplotlib, Scikit-learn, TensorFlow, PyTorch
    - **Web Development**: Django, Flask, FastAPI
    - **Automation & Scripting**: Selenium, PyAutoGUI, BeautifulSoup
    - **Cloud Computing**: AWS SDK (boto3), Google Cloud SDK
7. **Python for Web Development** –
    - **Django**: A full-stack web framework with built-in authentication, ORM, and security features.
    - **Flask**: A lightweight micro-framework for building web applications quickly.
8. **Python for Data Science & AI** –
    - **Pandas** for data analysis

    - **NumPy** for numerical computing
    - **Matplotlib & Seaborn** for visualization
    - **Scikit-learn** for machine learning algorithms
    - **TensorFlow & PyTorch** for deep learning
9. **Performance & Optimization** – Python can be slow due to its interpreted nature, but performance can be improved using:
    - **Cython** (compiling Python to C for better performance)
    - **Numba** (JIT compilation for numerical computations)
    - **Multiprocessing & Threading** for parallel execution
10. **Applications of Python** – Python is used in various domains, including:

- **Web development** (Django, Flask)
- **Data science & analytics** (Jupyter Notebook, Pandas)
- **AI & machine learning** (TensorFlow, PyTorch)
- **Cybersecurity & penetration testing** (Scapy, PyCrypto)
- **Cloud & DevOps** (AWS Lambda, Kubernetes automation)