Python Full Roadmap

🔰 PART 1: PYTHON FUNDAMENTALS

1. Introduction & Setup

- What is Python and why it's popular
- Installing Python
- Choosing an IDE (VS Code, PyCharm, Replit)
- Running your first Python program (print("Hello, World!"))
- Introduction to pip (Python package manager)

2. Basic Syntax

- Variables and naming rules
- Comments (# single-line, ''' multi-line ''')
- print() and input()
- Escape characters (\n, \t)

3. Data Types and Type Conversion

- Numbers: int, float, complex
- Text: str
- Boolean: True, False
- Type conversion: int(), str(), float()

4. Operators

- Arithmetic: +, -, *, /, //, %, **
- Comparison: ==, !=, >, <, >=, <=
- Logical: and, or, not
- Assignment: =, +=, -=, etc.

5. Control Flow

- if, elif, else
- Nested conditionals
- match (Python 3.10+ only)

6. Loops

- for loops with range()
- while loops

- break, continue, pass

🔧 PART 2: WORKING WITH DATA & STRUCTURES

7. Strings

- Indexing, slicing, and formatting
- String methods: .upper(), .lower(), .replace(), .split(), etc.
- f-strings and format()

8. Lists

- Creating, accessing, modifying
- List methods: .append(), .remove(), .sort(), .reverse()
- List comprehension

9. Tuples

- Immutable sequences
- Tuple unpacking

10. Dictionaries

- Key-value pairs
- .get(), .items(), .keys(), .values()
- Nested dictionaries

11. Sets

- Unique, unordered elements
- Set operations: union, intersection, difference

🧠 PART 3: FUNCTIONS AND OOP

12. Functions

- def, return values
- Parameters and arguments
- Default, keyword, *args, **kwargs
- Lambda functions
- Scope: local vs global

13. Modules & Packages

- import, from, as

- Writing your own modules
- Using standard library: math, random, datetime, os

14. Object-Oriented Programming

- Classes and Objects
- __init__ constructor
- self keyword
- Instance and class variables
- Inheritance and method overriding
- Encapsulation and abstraction

💾 PART 4: FILES, ERRORS, & STANDARD LIBRARIES

15. File Handling

- Opening, reading, writing, appending
- with statement
- Working with text, CSV, and JSON

16. Error Handling

- try, except, else, finally
- Custom exceptions
- Raising errors: raise

17. Useful Standard Libraries

- os, sys, platform, shutil
- time, calendar, collections, pathlib

🔌 PART 5: WORKING WITH LIBRARIES

18. Data Handling

- pandas: dataframes, CSV reading/writing
- numpy: arrays, matrix operations

19. Data Visualization

- matplotlib: line, bar, scatter plots
- seaborn: heatmaps, histograms, pair plots

20. Web & Networking

- requests: GET, POST requests
- BeautifulSoup: scraping HTML pages
- socket: basic networking
- http.server: basic web server

21. Automation

- pyautogui: controlling keyboard/mouse
- subprocess: running shell commands
- schedule: task scheduling
- paramiko: SSH automation

22. Cybersecurity Tools

- hashlib: hashing passwords
- secrets: generating secure tokens
- scapy: packet manipulation (advanced)

23. GUI Programming

- tkinter: basic windows, buttons, inputs
- customtkinter: modern GUI toolkit

24. Game Development

- pygame: sprites, sounds, events, collision

🚀 PART 6: ADVANCED PYTHON & PROJECTS

25. Working with APIs

- REST APIs
- JSON responses
- Weather app, news app using public APIs

26. Multithreading & Multiprocessing

- Threads vs Processes
- threading.Thread
- multiprocessing.Process

27. Virtual Environments & Project Structure

- venv, pip
- requirements.txt

- Folder organization

28. Capstone Projects

- **Web App** using Flask
- **Data Dashboard** with Pandas + Matplotlib
- **Todo GUI App** with Tkinter
- **Cybersecurity Scanner** with Sockets
- **Simple Game** with Pygame
- **Scraper + API Tool** using Requests + BeautifulSoup