■ Python Fundamentals Roadmap (Basic → Advanced)

1. Basics of Python

- Introduction to Python & Installation
- Python Syntax & Indentation
- Variables & Constants
- Data Types (int, float, str, bool, complex)
- Type Conversion & Casting
- Input / Output Functions (input(), print())
- Comments & Docstrings

2. Operators

- Arithmetic Operators (+, -, *, /, %, **, //)
- Comparison / Relational Operators (==, !=, <, >, <=, >=)
- Logical Operators (and, or, not)
- Assignment Operators (=, +=, -=, *=, /= etc.)
- Identity Operators (is, is not)
- Membership Operators (in, not in)
- Bitwise Operators (&, |, ^, ~, <<, >>)

3. Control Flow

- Conditional Statements (if, if-else, if-elif-else, nested conditions)
- Looping (for loop, while loop, nested loops)
- Loop Control (break, continue, pass)

4. Data Structures - Strings, Lists, Tuples, Sets, Dictionaries

- Strings: creation, indexing, slicing, methods, formatting
- Lists: creation, methods, comprehensions
- Tuples: immutability, methods
- Sets: operations, methods
- Dictionaries: key-value pairs, methods, nested dictionaries

5. Functions

- Defining functions with def
- Arguments: positional, keyword, default, *args, **kwargs
- Return values
- Scope of variables (local vs global)
- Lambda functions
- Built-in functions (map, filter, reduce, zip, enumerate)
- Recursion basics

6. Modules & Packages

- Importing modules (math, random, os, sys, datetime)
- Creating your own module
- Working with pip & packages
- Virtual environments

7. File Handling

- Reading & writing text files
- File modes (r, w, a, rb, wb)
- CSV file handling (csv module)
- JSON file handling (json module)

8. Exception Handling

- Errors vs Exceptions
- try, except blocks
- finally, else
- Raising exceptions (raise)
- Custom exceptions

9. Object-Oriented Programming (OOP)

- Classes & Objects
- __init__ method (constructor)
- Instance vs Class variables
- Methods (instance, class, static)
- Encapsulation (private, protected, public)
- Inheritance (single, multiple, multilevel, hierarchical)
- Method overriding & super()
- Polymorphism (duck typing, operator overloading)
- Abstraction (abstract classes using abc module)

10. Advanced Python (Fundamentals Extension)

- Iterators & Generators (__iter__, __next__, yield)
- Decorators (function & class decorators)
- Context Managers (with, __enter__, __exit__)
- Regular Expressions (re module)
- Collections module (Counter, defaultdict, namedtuple, deque)
- Date & Time handling (datetime, time, calendar)
- Logging module (basics)