

# 1. Python Mastery:

## 0. Why Learn Python? Real-World Use Cases

- 0.1. Python in AI, Web, Automation & DSA
- 0.2. Installing Python, VS Code, Jupyter, PyCharm
- 0.3. Creating and Running .py Files
- 0.4. Python REPL, IDLE & Virtual Environments (venv, virtualenv)

## 1. Variables & Data Types

- 1.1. Variables, Memory Allocation, id()
- 1.2. Constants & Naming Conventions
- 1.3. Data Types: int, float, bool, str, NoneType, complex
- 1.4. Type Checking and Conversion
- 1.5. Number Systems: int, float, complex, decimal, fractions
- 1.6. Floating-Point Precision Issues
- 1.7. Deep Dive: math, decimal, fractions

## 2. Strings in Python

- 2.1. Creating & Accessing Strings
- 2.2. Indexing, Slicing & Extended Slicing
- 2.3. Immutability & Interning
- 2.4. String Methods (Part 1 & 2)
- 2.5. Escape Sequences & Raw Strings
- 2.6. Unicode & Multiline Strings
- 2.7. String Formatting: f-strings, .format(), % formatting
- 2.8. Template Strings & Use Cases

## 3. Booleans & Operators

- 3.1. Boolean Values & Comparisons
- 3.2. Logical Operators: and, or, not
- 3.3. Operator Precedence

3.4. Truthy vs Falsy

3.5. Identity Operators: is, is not

## **4. Control Flow**

4.1. if, elif, else - Basics

4.2. Nested Conditions

4.3. Ternary Operator

4.4. match-case (Python 3.10+ Pattern Matching)

## **5. Loops**

5.1. for Loop on Iterables

5.2. while Loop

5.3. break, continue, pass

5.4. Loop with else

5.5. Loop Idioms (sum, filter)

5.6. enumerate()

5.7. zip()

5.8. reversed()

5.9. sorted()

5.10. Looping Over Dictionaries

5.11. Nested Loops & Complexity

## **6. Python Built-in Data Structures**

6.1. Lists: Creation, Methods, Slicing, Nesting

6.2. Deep vs Shallow Copy

6.3. Tuples: Packing, Unpacking, Single-element, namedtuple

6.4. Sets: Properties, Methods, Set Operations, frozenset

6.5. Dictionaries: Methods, Nested, View vs Copy

6.6. List, Set, Dict Comprehensions

## **7. Functions**

- 7.1. Defining, Calling, Returning
- 7.2. Positional, Keyword, Default Args
- 7.3. \*args, \*\*kwargs, /, \* syntax
- 7.4. Lambda Functions & Use Cases
- 7.5. Closures
- 7.6. Decorators (Basic to Stacked)
- 7.7. Built-in Decorators: @staticmethod, @classmethod

## **8. Modules & Packages**

- 8.1. Import Styles
- 8.2. \_\_name\_\_ == "\_\_main\_\_"
- 8.3. Absolute vs Relative Imports
- 8.4. venv, pip, requirements.txt

## **9. OOP in Python**

- 9.1. Classes & Objects
- 9.2. Special Methods: \_\_init\_\_, \_\_str\_\_, \_\_eq\_\_, etc.
- 9.3. Instance, Class, Static Methods
- 9.4. Inheritance, super(), MRO
- 9.5. Multiple Inheritance
- 9.6. Encapsulation, Name Mangling
- 9.7. Abstraction with ABC
- 9.8. Duck Typing

## **10. File Handling**

- 10.1. Open, Read, Write, Append
- 10.2. File Modes & Pointers
- 10.3. File Context Manager
- 10.4. CSV: csv.reader, DictWriter
- 10.5. JSON: json.load, json.dumps
- 10.6. Pickle: Usage & Security

## **11. Exception Handling & Debugging**

- 11.1. try-except, else, finally
- 11.2. Custom Exceptions
- 11.3. raise, assert
- 11.4. Debugging: pdb, breakpoint(), traceback
- 11.5. Logging

## **12. Iterators & Generators**

- 12.1. Iterator Protocol
- 12.2. Creating Custom Iterators
- 12.3. Generator Functions, yield
- 12.4. Generator Expressions

## **13. Functional Programming**

- 13.1. map(), filter(), reduce()
- 13.2. zip(), sorted()
- 13.3. any(), all()

## **14. MODULE: WEB & API PROJECTS IN PYTHON | Requests Module**

- 14.1. requests.get, post, headers, params
- 14.2. Status Codes, Timeout, Retry
- 14.3. Downloading Files

## **15. Mini Project**

- 15.1. CLI Weather App using OpenWeatherMap API

## **16. MODULE: DATA STRUCTURES & ALGORITHMS (DSA) | Foundations**

- 16.1. What are DS & Algorithms?
- 16.2. Time & Space Complexity
- 16.3. Big-O, Omega, Theta

## **17. Arrays**

- 17.1. Basics, Indexing, Traversal
- 17.2. Insert, Delete
- 17.3. Prefix Sum
- 17.4. Sliding Window Technique
- 17.5. Two Pointer Technique
- 17.6. Kadane's Algorithm
- 17.7. Trapping Rain Water

## **18. Strings**

- 18.1. Operations & Substrings
- 18.2. Palindromes
- 18.3. Anagram Checking
- 18.4. Hashing in Strings
- 18.5. Rabin-Karp Intro

## **19. Linked Lists**

- 19.1. Singly Linked List: Create, Insert, Delete
- 19.2. Detect Cycle (Floyd's)
- 19.3. Reverse Linked List (Iterative + Recursive)
- 19.4. Doubly Linked List: Head, Tail, Pos
- 19.5. Circular Linked List: Concept & Use
- 19.6. LRU Cache (DLL + HashMap)
- 19.7. Merge K Sorted Lists

## **20. Stacks**

- 20.1. Using Array & Linked List
- 20.2. Balanced Parentheses
- 20.3. Reverse Polish Notation
- 20.4. Stock Span Problem

## 20.5. Min/Max Stack

## 21. Queues

21.1. Normal Queue

21.2. Circular Queue

21.3. Deque (Double Ended Queue)

21.4. Monotonic Queue

21.5. Sliding Window Maximum

21.6. NGE (Next Greater Element)

## 22. Recursion

22.1. Base & Recursive Case

22.2. Tail Recursion

22.3. Recursion Tree

## 23. Backtracking

23.1. N-Queens

23.2. Subsets

23.3. Permutations

23.4. Combinations

23.5. Sudoku Solver

23.6. Word Search

23.7. Rat in Maze

## 24. Linear Search

24.1. Concept & Use

## 25. Binary Search

25.1. Basic Dry Run

25.2. First/Last Occurrence

25.3. Rotated Sorted Array

25.4. Peak Element

25.5. Binary Search on Answers

## **26. Sorting Algorithms**

26.1. Bubble, Selection, Insertion Sort

26.2. Merge Sort

26.3. Quick Sort

26.4. Heap Sort

26.5. Counting Sort

26.6. Radix Sort

26.7. Stability, Time/Space Analysis

## **27. Hashing**

27.1. Concepts: HashMap, HashSet

27.2. Load Factor

27.3. Two Sum

27.4. Longest Subarray with Sum K

27.5. Subarray with 0 Sum

27.6. Group Anagrams

## **28. Greedy Algorithms**

28.1. Strategy, When to Use

28.2. Activity Selection

28.3. Fractional Knapsack

28.4. Huffman Coding

28.5. Job Scheduling

28.6. Gas Station

28.7. Minimum Platforms

## **29. Divide and Conquer**

29.1. Concept, Master Theorem

- 29.2. Merge Sort
- 29.3. Quick Sort
- 29.4. Majority Element
- 29.5. Closest Pair of Points

### **30. Bit Manipulation**

- 30.1. AND, OR, XOR, NOT
- 30.2. Shifts: <<, >>
- 30.3. Count Set Bits
- 30.4. Power of 2
- 30.5. XOR Tricks
- 30.6. Bit Masking

### **31. Dynamic Programming**

- 31.1. Recursion → Memo → Tabulation
- 31.2. Fibonacci Variations
- 31.3. Climbing Stairs
- 31.4. House Robber
- 31.5. Min Cost Climbing Stairs
- 31.6. 0/1 Knapsack
- 31.7. Subset Sum
- 31.8. Target Sum
- 31.9. Longest Common Subsequence
- 31.10. Longest Palindromic Subsequence
- 31.11. Matrix Chain Multiplication
- 31.12. DP on Trees
- 31.13. DP on Grids
- 31.14. DP with Bitmask
- 31.15. DP with Multiple States

### **32. Graph Algorithms**



- 32.1. Graph Types: Directed, Undirected, Weighted
- 32.2. Adjacency List vs Matrix
- 32.3. BFS Traversal
- 32.4. DFS Traversal
- 32.5. Connected Components
- 32.6. Cycle Detection using DFS
- 32.7. Cycle Detection using Union Find
- 32.8. Topological Sort
- 32.9. Bipartite Check
- 32.10. Dijkstra's Algorithm
- 32.11. Bellman-Ford Algorithm
- 32.12. Floyd-Warshall Algorithm
- 32.13. Prim's Algorithm
- 32.14. Kruskal's Algorithm
- 32.15. DSU (Disjoint Set Union)

### **33. Trees**

- 33.1. Binary Tree Basics
- 33.2. Inorder Traversal
- 33.3. Preorder Traversal
- 33.4. Postorder Traversal
- 33.5. Level Order Traversal (BFS)
- 33.6. Tree Height
- 33.7. Tree Diameter
- 33.8. Leaf Node Count
- 33.9. Binary Search Tree: Insert, Search, Delete
- 33.10. Validate BST
- 33.11. Kth Smallest/Largest in BST
- 33.12. AVL Trees (Intro)
- 33.13. Segment Tree Basics
- 33.14. Fenwick Tree (BIT)

33.15. Trie (Prefix Tree)

### **34. Heaps & Priority Queues**

- 34.1. Heap Basics: Max/Min
- 34.2. Heapify Logic
- 34.3. Python PriorityQueue
- 34.4. Kth Largest/Smallest Element
- 34.5. Median in Stream
- 34.6. Merge K Sorted Arrays
- 34.7. Top K Frequent Elements

### **35. Sliding Window Techniques**

- 35.1. Max/Min Sum in Window
- 35.2. Longest Substring without Repeat
- 35.3. Max Sliding Window

### **36. Two Pointer Techniques**

- 36.1. Pair Sum
- 36.2. Container With Most Water
- 36.3. 3Sum, 4Sum
- 36.4. Remove Duplicates

### **37. MODULE: PRACTICE + CHALLENGES**

- 37.1. 3 problems per concept (LeetCode, GfG, Codeforces)
- 37.2. Practice Sheets
- 37.3. Daily Coding Challenge Plan
- 37.4. Mock Tests

### **38. MODULE: CAPSTONE PROJECTS | Project 1: Build LRU Cache**

- 38.1. DLL + HashMap Design
- 38.2. Insert/Delete/Access with  $O(1)$

### 38.3. Unit Testing

## **39. Project 2: Resume Filter System**

39.1. Parse Resumes from CSV/PDF

39.2. JD-Resume Match Score using Keywords

39.3. Hashing + Search + Ranking Logic

39.4. GUI/CLI Options

## **40. Project 3: Visual Pathfinding App**

40.1. BFS, DFS, Dijkstra on Grid

40.2. GUI with Tkinter or Streamlit

40.3. Realtime Node Visualization

40.4. Export Solution Path