■ Python Practice Set – 100 Realistic Problem Statements

Designed for students, freshers, and experienced programmers. Each problem includes a realistic goal, input/output format, and a sample case.

Difficulty Tracks: Beginner (1–30) • Intermediate (31–70) • Advanced (71–100)

Beginner Level (1–30): Basics & Logic

Problem 1: Greeting the World

Goal: Print the classic first program message to verify your setup.
Input: No input.

Output: Prints a single line message.

Example:
Output: Hello, World!

Problem 2: Simple Addition

Goal: Read two integers and output their sum in a readable sentence.
Input: Two integers X and Y (space-separated).

Output: A line: Sum of X and Y is Z

Example:
Input: 3 7 Output: Sum of 3 and 7 is 10

Problem 3: Maximum of Three

Goal: Find the largest of three given integers.
Input: Three integers A B C.

Output: Largest value among A, B, C.

Example:
Input: 10 25 7 Output: 25

Problem 4: Even or Odd

Goal: Determine whether a number is even or odd.
Input: Single integer N.

Output: Print Even or Odd.

Example:
Input: 9 Output: Odd

Problem 5: Swap Without Temp

Goal: Swap two variables without using a third variable.
Input: Two integers A B.

Output: Two integers swapped.

Example:
Input: 5 9 Output: 9 5

Problem 6: Factorial (Loop)

Goal: Compute factorial of a non-negative integer using loops.
Input: Integer N ($0 \le N \le 12$).

Output: Value of N!

Example:
Input: 5 Output: 120

Problem 7: Multiplication Table

Goal: Print first 10 multiples of a given number.
Input: Integer N.

Output: 10 lines: i × N = value (i from 1..10).

Example:
Input: 6 Output (first 2 lines): 1 x 6 = 6 2 x 6 = 12 ...

Problem 8: Prime Check

Goal: Check if a number is prime.
Input: Integer N ($N \ge 2$).

Output: Prime or Not Prime.

Example:
Input: 13 Output: Prime

Problem 9: Fibonacci (First N)

Goal: Print first N Fibonacci numbers (starting 0, 1).
Input: Integer N ($N \geq 1$).

Output: Sequence space-separated.

Example:
Input: 6 Output: 0 1 1 2 3 5

Problem 10: Reverse a String (No Slicing)

Goal: Reverse a string manually without using slicing.
Input: One line string S.

Output: Reversed string.

Example:
Input: python Output: nohtyp

Problem 11: Vowel & Consonant Count

Goal: Count vowels and consonants in a lowercase English string.
Input: String S (letters only).

Output: Two integers: vowels consonants.

Example:
Input: education Output: 5 4

Problem 12: Palindrome String

Goal: Check if a given string is a palindrome (case-insensitive, alphanumeric only).
Input: String S.

Output: True or False.

Example:
Input: Madam Output: True

Problem 13: Largest in List

Goal: Find the largest element from a list of integers.
Input: Integer N followed by N integers.

Output: Single integer: maximum value.

Example:
Input: 5 2 9 1 7 3 Output: 9

Problem 14: Second Largest in List

Goal: Find the second largest distinct number.
Input: N then N integers (may have duplicates).

Output: Second largest distinct value or 'NA' if it doesn't exist.

Example:
Input: 5 4 4 1 3 2 Output: 3

Problem 15: Manual Sort

Goal: Sort a list of integers in non-decreasing order without using sort().
Input: N then N integers.

Output: Sorted sequence space-separated.

Example:
Input: 4 9 2 7 2 Output: 2 2 7 9

Problem 16: Remove Duplicates

Goal: Remove duplicates while preserving first occurrence order.
Input: N then N integers.

Output: De-duplicated list space-separated.

Example:
Input: 7 1 2 2 3 1 4 3 Output: 1 2 3 4

Problem 17: Sum of Digits

Goal: Compute sum of digits of a non-negative integer.
Input: Integer N.

Output: Sum of its digits.

Example:
Input: 5029 Output: 16

Problem 18: GCD of Two Numbers

Goal: Compute greatest common divisor using Euclid's algorithm.
Input: Two integers A B.

Output: Single integer: gcd(A,B).

Example:
Input: 36 60 Output: 12

Problem 19: LCM of Two Numbers

Goal: Compute least common multiple using gcd.
Input: Two integers A B.

Output: Single integer: lcm(A,B).

Example:
Input: 12 18 Output: 36

Problem 20: Celsius → Fahrenheit

Goal: Convert temperature from Celsius to Fahrenheit.
Input: Real number C.

Output: Fahrenheit value (C×9/5 + 32) rounded to 2 decimals.

Example:
Input: 37 Output: 98.60

Problem 21: Decimal to Binary (Manual)

Goal: Convert a non-negative integer to binary without using bin().
Input: Integer N.

Output: Binary string.

Example:
Input: 10 Output: 1010

Problem 22: Word Count

Goal: Count the number of words in a sentence (split on spaces).
Input: A sentence line.

Output: Integer word count.

Example:
Input: I love Python programming Output: 4

Problem 23: Pattern – Left Pyramid

Goal: Print a left-aligned pyramid of * of height H.
Input: Integer H (1..20).

Output: H lines forming a pyramid.

Example:
Input: 3 Output: * ** ***

Problem 24: ASCII of Character

Goal: Print ASCII value of a given character.
Input: Single character ch.

Output: Integer ASCII code.

Example:
Input: A Output: 65

Problem 25: Primes in Range

Goal: List all prime numbers in [L, R].
Input: Two integers L R (L ≤ R).

Output: Primes space-separated or 'NA' if none.

Example:
Input: 10 20 Output: 11 13 17 19

Problem 26: Armstrong Number

Goal: Check if a number equals sum of its digits raised to power of digit count.
Input: Integer N.

Output: True or False.

Example:
Input: 153 Output: True

Problem 27: Length Without len()

Goal: Find length of a list/sequence without using len().
Input: A string S.

Output: Integer length of S.

Example:
Input: hello Output: 5

Problem 28: Merge Two Sorted Lists

Goal: Merge two individually sorted lists into one sorted list.
Input: N, list1; M, list2 (both sorted).

Output: Merged sorted list.

Example:
Input: 3 1 4 7 4 2 3 5 6 Output: 1 2 3 4 5 6 7

Problem 29: Reverse Words in Sentence

Goal: Reverse the order of words in a sentence, preserving words themselves.
Input: Sentence S.

Output: Words in reverse order.

Example:
Input: I love deep learning Output: learning deep love I

Problem 30: Simple Calculator

Goal: Perform +, –, ×, ÷ on two numbers based on an operator.
Input: A operator B (e.g., 5 * 7).

Output: Result (float for division).

Example:
Input: 8 / 2 Output: 4.0

Intermediate Level (31–70): DS & Algorithms

Problem 31: Linear Search

Goal: Find index of target in list using linear search; return -1 if not found.
Input: N then list of N integers, then target T.

Output: Index (0-based) or -1.

Example:
Input: 5 4 1 9 2 7 2 Output: 3

Problem 32: Binary Search (Iterative)

Goal: Search target in a sorted array using iterative binary search.
Input: N then sorted list; then target T.

Output: Index (0-based) or -1.

Example:
Input: 5 1 3 5 7 9 7 Output: 3

Problem 33: Bubble Sort

Goal: Sort list using bubble sort and count swaps.
Input: N then list of N integers.

Output: Sorted list and number of swaps.

Example:
Input: 4 4 3 2 1 Output: 1 2 3 4 6

Problem 34: Insertion Sort

Goal: Implement insertion sort and print array after each insertion step.
Input: N then list.

Output: Intermediate arrays per line, then final sorted array.

Example:
Input: 5 5 2 4 6 1 Output: 2 5 4 6 1 2 4 5 6 1 ...

Problem 35: Selection Sort

Goal: Implement selection sort and print selected minimum index each pass.
Input: N then list.

Output: Final sorted list and indices chosen per pass.

Example:
Input: 4 3 1 4 2 Output: 1 2 3 4 1 3 3

Problem 36: Merge Sort

Goal: Implement merge sort; print final sorted list.
Input: N then list.

Output: Sorted list.

Example:
Input: 6 10 9 8 7 6 5 Output: 5 6 7 8 9 10

Problem 37: Quick Sort (Lomuto)

Goal: Implement quick sort with Lomuto partition; print array after each partition.
Input: N then list.

Output: Final sorted list.

Example:
Input: 5 4 5 3 7 2 Output: 2 3 4 5 7

Problem 38: Frequency Count

Goal: Count frequency of each distinct element and print as value:count by ascending value.
Input: N then list.

Output: Lines in 'value: count' format.

Example:
Input: 6 1 2 2 3 1 2 Output: 1: 2 2: 3 3: 1

Problem 39: First Non-Repeating Character

Goal: Return first character with frequency 1 in a string; else 'NA'.
Input: String S.

Output: Character or 'NA'.

Example:
Input: swiss Output: w

Problem 40: Anagram Check

Goal: Check if two strings are anagrams (ignore case and spaces).
Input: Two lines: S1 and S2.

Output: True or False.

Example:
Input: Dormitory Dirty room Output: True

Problem 41: All Substrings

Goal: Generate all substrings of a string S.
Input: String S.

Output: Substrings, each on new line.

Example:
Input: abc Output: a ab abc b bc c

Problem 42: Word Frequencies

Goal: Count occurrences of each word (case-insensitive, strip punctuation).
Input: Sentence line.

Output: word: count lines sorted lexicographically.

Example:
Input: To be, or not to be Output: be: 2 not: 1 to: 2 or: 1

Problem 43: Balanced Parentheses

Goal: Use a stack to validate brackets ()[]{} in a string.
Input: String of brackets.

Output: True or False.

Example:
Input: {[()]} Output: True

Problem 44: Infix to Postfix

Goal: Convert infix expression to postfix (operators +,-,*,/,^).
Input: Infix string with spaces.

Output: Postfix string with spaces.

Example:
Input: A + B * C Output: A B C * +

Problem 45: Stack Using List

Goal: Implement push, pop, top operations; process Q queries.
Input: Q then Q operations (PUSH x / POP / TOP).

Output: Outputs of POP/TOP or 'EMPTY'.

Example:
Input: 4 PUSH 5 PUSH 2 TOP POP Output: 2

Problem 46: Queue Using List

Goal: Implement enqueue, dequeue, front; process Q operations.
Input: Q then operations (ENQ x / DEQ / FRONT).

Output: Outputs of DEQ/FRONT or 'EMPTY'.

Example:
Input: 3 ENQ 10 DEQ FRONT Output: EMPTY

Problem 47: Circular Queue

Goal: Implement circular queue of capacity K.
Input: K then Q operations.

Output: Outputs for DEQ/FRONT or 'FULL'/'EMPTY'.

Example:
Input: 3 ENQ 1 ENQ 2 ENQ 3 ENQ 4 Output: FULL

Problem 48: Singly Linked List

Goal: Create a singly linked list; support insert at head, tail, and print.
Input: Q operations (IH x / IT x / P).

Output: Print list when P.

Example:
Input: 4 IH 2 IT 3 IH 1 P Output: 1 2 3

Problem 49: Doubly Linked List

Goal: Implement doubly linked list with delete key operation.
Input: Q operations (IH/IT/D x/P).

Output: Print list when P.

Example:
Input: 5 IH 2 IH 1 IT 3 D 2 P Output: 1 3

Problem 50: Stack via Linked List

Goal: Implement a stack backed by a linked list.
Input: Q operations (PUSH/POP/TOP).

Output: Outputs for POP/TOP or 'EMPTY'.

Example:
Input: 3 POP PUSH 7 TOP Output: EMPTY 7

Problem 51: Queue via Linked List

Goal: Implement a queue backed by a linked list.
Input: Q operations (ENQ/DEQ/FRONT).

Output: Outputs for DEQ/FRONT or 'EMPTY'.

Example:
Input: 3 ENQ 1 ENQ 2 DEQ Output: 1

Problem 52: Binary Search Tree (Insert, Search, Inorder)

Goal: Implement BST with insert and search; print inorder traversal at end.
Input: N then N inserts; then Q searches.

Output: Inorder list; then results of searches True/False.

Example:
Input: 5 5 3 7 2 4 2 3 6 Output: 2 3 4 5 7 True False

Problem 53: Binary Tree Height

Goal: Compute height of a binary tree (edges).
Input: Level-order nodes with 'null' for empty.

Output: Single integer height.

Example:
Input: 1 2 3 4 5 null 6 Output: 2

Problem 54: Max Element in Binary Tree

Goal: Find maximum value in a binary tree.
Input: Same input as 53.

Output: Max value.

Example:
Input: 1 7 3 4 5 null 6 Output: 7

Problem 55: Graph (Adjacency List)

Goal: Build an undirected graph; print adjacency list sorted by vertex id.
Input: N M then M edges u v.

Output: Adjacency list (u: neighbors...).

Example:
Input: 3 2 1 2 2 3 Output: 1: 2 2: 1 3 3: 2

Problem 56: BFS Traversal

Goal: Perform BFS from a start node; print visiting order.
Input: N M, edges, then start S.

Output: Order space-separated.

Example:
Input: 4 3 1 2 2 3 2 4 2 Output: 2 1 3 4

Problem 57: DFS Traversal

Goal: Perform DFS from a start node (lexicographic neighbor order).
Input: N M, edges, start S.

Output: Order space-separated.

Example:
Input: 4 3 1 2 2 3 2 4 2 Output: 2 1 3 4

Problem 58: Dijkstra Shortest Paths

Goal: Compute shortest distances from source S in weighted graph.
Input: N M, edges u v w, then S.

Output: Distances for 1..N space-separated (INF if unreachable).

Example:
Input: 3 3 1 2 4 1 3 2 2 3 1 1 Output: 0 3 2

Problem 59: Detect Cycle (Undirected)

Goal: Detect if an undirected graph contains a cycle.
Input: N M, edges u v.

Output: True or False.

Example:
Input: 3 3 1 2 2 3 1 3 Output: True

Problem 60: Connected Components

Goal: Count number of connected components in an undirected graph.
Input: N M, edges u v.

Output: Single integer count.

Example:
Input: 4 2 1 2 3 4 Output: 2

Problem 61: Factorial (Recursion)

Goal: Compute factorial using recursion with base case 0! = 1.
Input: Integer N (0..15).

Output: N!

Example:
Input: 6 Output: 720

Problem 62: Fibonacci (Recursion)

Goal: Print Nth Fibonacci number using recursion (memoization optional).
Input: Integer N (0..30).

Output: Nth Fibonacci number.

Example:
Input: 7 Output: 13

Problem 63: Tower of Hanoi

Goal: Print moves to shift N disks from A to C using B.
Input: Integer N.

Output: Each line: A->C style move; final count = 2^N-1.

Example:
Input: 3 Output (first 2): A->C A->B ...

Problem 64: Fibonacci with Memoization

Goal: Return Nth Fibonacci using memoization to achieve O(N).
Input: Integer N (0..10^5).

Output: Nth Fibonacci modulo 1e9+7.

Example:
Input: 10 Output: 55

Problem 65: Recursive Palindrome

Goal: Check palindrome using recursion (ignore non-alphanumeric).
Input: String S.

Output: True or False.

Example:
Input: No 'x' in Nixon Output: True

Problem 66: N-Queens

Goal: Print one valid arrangement for N queens on NxN board as list of column indices.
Input: Integer N (1..12).

Output: N integers where i-th is column of queen in row i; or 'NA' if none.

Example:
Input: 4 Output: 2 4 1 3

Problem 67: Sudoku Solver

Goal: Solve a 9x9 Sudoku puzzle using backtracking.
Input: 9 lines of 9 digits with 0 for empty.

Output: Solved grid (9 lines).

Example:
Input (first row): 530070000 ... Output (first row): 534678912 ...

Problem 68: All Permutations

Goal: Generate all permutations of a string in lexicographic order.
Input: String S (unique chars).

Output: Each permutation on new line.

Example:
Input: abc Output: abc acb bac bca cab cba

Problem 69: Combinations (nCr) of List

Goal: Print all k-combinations of a list in lexicographic order.
Input: N list, then k.

Output: Each combination on new line space-separated.

Example:
Input: 4 1 2 3 4 2 Output: 1 2 1 3 ...

Problem 70: Binary Search (Recursion)

Goal: Implement recursive binary search returning index or -1.
Input: N sorted list; then T.

Output: Index or -1.

Example:
Input: 5 2 4 6 8 10 8 Output: 3

Advanced Level (71–100): OOP • Files • APIs • Apps

Problem 71: Student Class

Goal: Create a Student class with id, name, marks list; methods: average(), grade().
Input: N then student details; compute average and letter grade.

Output: For each student: id name average grade.

Example:
Input: 1 101 Alice 90 80 100 Output: 101 Alice 90.0 A

Problem 72: Bank Account Simulation

Goal: Class BankAccount with deposit, withdraw (fail on insufficient funds), balance.
Input: Q operations on a single account.

Output: Balance after each operation that changes it; or 'INSUFFICIENT'.

Example:
Input: 4 DEPOSIT 500 WITHDRAW 700 WITHDRAW 300 BALANCE Output: INSUFFICIENT 200 200

Problem 73: Rectangle Class

Goal: Rectangle with width, height; methods area(), perimeter(), is_square().
Input: Multiple rectangles.

Output: For each: area perimeter is_square (True/False).

Example:
Input: 2 3 4 5 5 Output: 12 14 False 25 20 True

Problem 74: Employee Inheritance

Goal: Base Employee(name, base_pay) and Manager(Employee, bonus). Compute annual pay.
Input: N employees then print annual_pay.

Output: For each: name annual_pay.

Example:
Input: 2 E John 50000 M Sara 60000 10000 Output: John 50000 Sara 70000

Problem 75: Method Overriding (Shapes)

Goal: Create Shape base with area(); override in Circle/Rectangle/Triangle.
Input: Q queries with shape type and dims.

Output: Area rounded to 2 decimals.

Example:
Input: 2 C 3 R 3 4 Output: 28.27 12.00

Problem 76: Operator Overloading – Vector

Goal: Implement 2D Vector with +, ==, and str, repr.
Input: Q operations on vectors.

Output: Results of operations.

Example:
Input: 2 ADD 1 2 3 4 EQ 1 1 1 1 Output: (4, 6) True

Problem 77: Singleton Logger

Goal: Implement a singleton Logger that records messages to one in-memory list.
Input: Q operations: LOG msg / COUNT.

Output: Output COUNT result and last message.

Example:
Input: 3 LOG start LOG running COUNT Output: 2 running

Problem 78: Handle ZeroDivisionError

Goal: Divide A by B, handle division by zero gracefully.
Input: Two numbers A B.

Output: Quotient or 'DIVISION BY ZERO'.

Example:
Input: 10 0 Output: DIVISION BY ZERO

Problem 79: Multiple Exceptions

Goal: Read list of integers from strings; handle ValueError; always print final count.
Input: Line of space-separated tokens.

Output: Valid ints space-separated; then count as 'COUNT: k'.

Example:
Input: 10 two 3 Output: 10 3 COUNT: 2

Problem 80: Read Text File

Goal: Read a file path and print its contents line by line with line numbers.
Input: File path string.

Output: LineNumber: content.

Example:
Example: Input: notes.txt Output: 1: First line

Problem 81: File Stats

Goal: Count lines, words, characters in a text file.
Input: File path.

Output: lines words chars separated by spaces.

Example:
Output: 10 120 750

Problem 82: Copy File

Goal: Copy contents of source file to destination file.
Input: src dst paths.

Output: Print 'DONE' on success.

Example:
Output: DONE

Problem 83: Remove Stop Words

Goal: Remove common English stop words from a text and print remaining words.
Input: Line of text.

Output: Filtered text.

Example:
Input: this is a test of the system Output: this test system

Problem 84: Longest Word in File

Goal: Find the longest word in a text file (break ties by earliest).
Input: File path.

Output: The word itself.

Example:
Output: extraordinary

Problem 85: JSON Read/Write Employees

Goal: Read N employee dicts and write to JSON; then read back and print names.
Input: N then N lines 'name age dept'.

Output: Names one per line after reload.

Example:
Input: 2 Alice 30 IT Bob 25 HR Output: Alice Bob

Problem 86: CSV Read/Write Marks

Goal: Write student marks to CSV and compute subject averages.
Input: N lines: name m1 m2 m3.

Output: Average per subject rounded to 2 decimals.

Example:
Input: 2 A 80 90 100 B 70 60 50 Output: 75.00 75.00 75.00

Problem 87: Persist Student Records

Goal: Store student records (id,name,marks) to a file and retrieve by id.
Input: Q operations: ADD/GET id.

Output: On GET print 'id name avg'.

Example:
Input: 3 ADD 1 John 80 90 ADD 2 Ana 70 70 GET 2 Output: 2 Ana 70.0

Problem 88: Contact Book (File)

Goal: Create a contact book supporting ADD name phone and FIND name (prefix search).
Input: Q operations.

Output: Matching contacts alphabetically or 'NA'.

Example:
Input: 4 ADD Alice 999 ADD Bob 111 FIND Al FIND Z Output: Alice 999 NA

Problem 89: Weather API (Design)

Goal: Design a function that fetches weather for a city using an HTTP API; mock the response.
Input: City name string.

Output: Formatted weather line.

Example:
Input: Delhi Output: Delhi: 32°C, Clear

Problem 90: Send Email (Mock)

Goal: Write a function to send an email via SMTP; mock the actual send in tests.
Input: to, subject, body.

Output: Print 'SENT to '.

Example:
Input: admin@example.com Test Hi Output: SENT to admin@example.com

Problem 91: Web Scrape Headlines (Mock)

Goal: Parse HTML to extract headlines; do not request the web, parse given string.
Input: Raw HTML string.

Output: Each headline on new line.

Example:
Input: AB Output: A B

Problem 92: Mini Chatbot (Rules)

Goal: Implement rule-based chatbot that responds to greetings and farewells.
Input: Multiple user lines until 'exit'.

Output: Bot responses per line.

Example:
Input: hello bye exit Output: Hi there! Goodbye!

Problem 93: Rock–Paper–Scissors

Goal: Play RPS vs computer; ensure unbiased randomness; best of 3.
Input: User choices over rounds.

Output: Final winner statement.

Example:
Output: You win 2–1

Problem 94: Tic-Tac-Toe (CLI)

Goal: Two-player Tic-Tac-Toe on 3x3 board with win/draw detection.
Input: Moves as 'r c' until game over.

Output: Game result and board.

Example:
Output: X wins

Problem 95: Guess the Number

Goal: Computer picks number 1..100; user guesses with hints (higher/lower).
Input: User guesses.

Output: Number of attempts and success message.

Example:
Output: Correct in 6 attempts

Problem 96: Hangman (CLI)

Goal: Implement Hangman with a fixed word list; show masked word and misses.
Input: User letters until win/lose.

Output: Final state and result.

Example:
Output: You lost. Word was: PYTHON

Problem 97: GUI Calculator (Tkinter)

Goal: Build basic Tkinter calculator supporting + - * / with clear and equals.
Input: Button clicks.

Output: Computed results shown in display.

Example:
—

Problem 98: To-Do List (Tkinter)

Goal: Tkinter app with add/remove/complete tasks, persistent storage in JSON.
Input: User interactions.

Output: Final tasks written to file; show current list in UI.

Example:
—

Problem 99: Simple Flask App

Goal: Create Flask app with '/' hello route and '/add?a=&b;=' returning a+b.
Input: HTTP GET queries.

Output: JSON response with result.

Example:
Example: GET /add?a=2&b;=3 -> {"result":5}

Problem 100: Student Management System (Mini Project)

Goal: Menu-driven CLI: add/update/search/delete/display students; persist to file.
Input: Operations chosen by user.

Output: Appropriate confirmations and tabular display.

Example:
Output: Saved 5 records; Found id=3: Ana 78.0

Total problems included: 100 (should be 100).