Structured DSA Course Plan (90 Classes)

Course Overview

Duration: 180 Days (3 classes per week)
Class Duration: 2-3 Hours
Total Classes: 90

Course Structure

- **Phase 1 (Classes 1-30):** Fundamentals to Intermediate Topics
- **Phase 2 (Classes 31-90):** Advanced Topics & System Design

Class Format

1. **Concept Explanation:** 30-60 minutes
2. **Problem Solving & Live Examples:** 60-90 minutes
3. **Doubt Clearing & Assignment Discussion:** 30 minutes
4. **Home Assignments:** 5-10 problems per topic
5. **Mock Interviews & Revision:** Every 10-12 classes

Phase 1: First 30 Classes (Fundamentals - Intermediate)

Module 1: Introduction & Problem-Solving Fundamentals (4 Classes)

1. Meet & Greet + Course Introduction
2. HLD & LLD Basics (Overview of System Design)
3. Introduction to Problem-Solving + Time Complexity - 1
4. Time Complexity - 2 + Space Complexity + Asymptotic Analysis

Module 2: Arrays & Bit Manipulation (7 Classes)

1. Introduction to Arrays + Prefix Sum Technique
2. Carry Forward + Subarrays (Kadane's Algorithm)
3. 2D Arrays & Matrix Operations
4. Array Interview Problems (Sliding Window, Two Pointers)
5. Bit Manipulation - 1 (Basics, AND, OR, NOT, XOR)
6. Bit Manipulation - 2 (Set/Unset Bits, Power of 2, Count Set Bits)
7. Bit Manipulation - 3 (Advanced Applications & Problem Solving)

Module 3: Mathematics & String Manipulation (6 Classes)

1. Modular Arithmetic & Prime Numbers (Sieve of Eratosthenes)
2. GCD, LCM & Number Theory Fundamentals
3. String Basics + String Operations (Reversal, Palindrome)
4. String Pattern Matching (KMP, Rabin-Karp)
5. Advanced String Problems (Anagrams, Subsequences)

6. Combined Array & String Problem Solving

Module 4: Recursion, Sorting & Hashing (8 Classes)

1. Recursion - 1 (Basics, Call Stack Visualization)
2. Recursion - 2 (Backtracking Introduction, N-Queens Preview)
3. Sorting - 1 (Bubble, Selection, Insertion Sort)
4. Sorting - 2 (Merge Sort, Quick Sort)
5. Sorting - 3 (Counting Sort, Radix Sort, Bucket Sort)
6. Hashing - 1 (Maps, Sets, Hash Functions, Load Factor)
7. Hashing - 2 (Collision Handling, Frequency Counting Problems)
8. Subsequences & Subsets Generation

Module 5: Linked Lists, Stacks & Queues (5 Classes)

1. Linked List - 1 (Singly, Doubly, Circular)
2. Linked List - 2 (Reversal, Middle Element, Cycle Detection)
3. Stack - 1 (Implementation & Classic Problems)
4. Queue - 1 (Implementation, Circular Queue, Deque)
5. Stack & Queue Advanced Problems (Next Greater Element, LRU Cache)

Phase 2: Next 60 Classes (Advanced Topics & System Design)

Module 6: Advanced Arrays & Number Theory (6 Classes)

1. Advanced Array Techniques (Sliding Window, Kadane's Extensions)
2. Negative Number Handling & Inverse Modulo
3. Combinatorics & Advanced Modular Arithmetic (nCr, nPr)
4. Sorting - 4 (Heap Sort, Quick Sort Optimization)
5. Two Pointers - 1 (Basics, Pair Sum Problems)
6. Two Pointers - 2 (3Sum, 4Sum, Container Problems)

Module 7: Binary Search & Linked List Mastery (6 Classes)

1. Binary Search - 1 (Fundamentals, Search Space Reduction)
2. Binary Search - 2 (Binary Search on Answer Space)
3. Binary Search - 3 (Matrix Search, Rotated Arrays)
4. Linked List - 3 (Advanced Problems: Merge K Lists, Clone with Random)
5. Linked List - 4 (Flatten, Intersection, Add Numbers)
6. Problem-Solving Session: Linked Lists

Module 8: Trees & Tries (7 Classes)

1. Trees - 1 (Structure, Traversals: Inorder, Preorder, Postorder)
2. Trees - 2 (Binary Search Tree, AVL Trees, Balance)
3. Trees - 3 (Lowest Common Ancestor, Path Problems)
4. Trees - 4 (Diameter, Height, Views, Serialize/Deserialize)
5. Tries - 1 (Basics, Implementation, Insert/Search)
6. Tries - 2 (Word Search, Dictionary Problems)
7. Tries - 3 (Prefix Matching, Autocomplete, XOR Problems)

Module 9: Heap & Greedy Algorithms (6 Classes)

1. Heap - 1 (Min Heap, Max Heap, Heapify Operations)
2. Heap - 2 (Heap Sort, Priority Queue Implementation)

3. Heap - 3 (Top K Elements, Median Stream, Merge K Lists)
4. Greedy Algorithms - 1 (Introduction, Activity Selection)
5. Greedy Algorithms - 2 (Fractional Knapsack, Job Sequencing)
6. Greedy Algorithms - 3 (Huffman Coding, Gas Station, Jump Game)

Module 10: Backtracking & Dynamic Programming (12 Classes)

1. Backtracking - 1 (N-Queens, Sudoku Solver)
2. Backtracking - 2 (Permutations, Combinations, Rat in Maze)
3. Dynamic Programming - 1 (Introduction, Memoization vs Tabulation)
4. Dynamic Programming - 2 (Fibonacci, Climbing Stairs, Coin Change)
5. Dynamic Programming - 3 (0/1 Knapsack, Unbounded Knapsack)
6. Dynamic Programming - 4 (LCS, LIS, Edit Distance)
7. Dynamic Programming - 5 (Matrix Chain Multiplication, Optimal BST)
8. Dynamic Programming - 6 (Subset Sum, Partition Problems)
9. Dynamic Programming - 7 (Palindrome Partitioning, Egg Drop)
10. Dynamic Programming - 8 (DP on Trees, Digit DP)
11. Dynamic Programming - 9 (Problem-Solving Marathon)
12. Mock Interview: DP Mastery

Module 11: Graph Algorithms (9 Classes)

1. Graph - 1 (Representation: Adjacency Matrix/List, Edge List)
2. Graph - 2 (DFS, BFS, Connected Components)
3. Graph - 3 (Cycle Detection in Directed/Undirected Graphs)
4. Graph - 4 (Topological Sorting, Kahn's Algorithm)
5. Graph - 5 (Minimum Spanning Tree: Prim's Algorithm)
6. Graph - 6 (Dijkstra's Algorithm, Shortest Path)
7. Graph - 7 (Bellman-Ford, Floyd-Warshall, Negative Cycles)
8. Graph - 8 (Disjoint Set Union, Kruskal's Algorithm)
9. Graph - 9 (Advanced Problems: Bridges, Articulation Points)

Module 12: System Design & Final Preparation (8 Classes)

Low-Level Design (LLD) - 3 Classes

1. **LLD - 1:** SOLID Principles, Design Patterns (Singleton, Factory, Observer)
2. **LLD - 2:** Object-Oriented Design (Parking Lot, Library Management System)
3. **LLD - 3:** Design Chess Game, Elevator System, ATM Machine

High-Level Design (HLD) - 3 Classes

4. **HLD - 1:** System Design Fundamentals (Scalability, Load Balancing, Caching)
5. **HLD - 2:** Database Design (SQL vs NoSQL, Sharding, Replication)
6. **HLD - 3:** Design URL Shortener, Design Twitter, Design WhatsApp

Mock Interviews & Revision - 2 Classes

7. **Mock Interview - 1:** DSA Problem Solving (Arrays, Strings, Trees, DP)
8. **Mock Interview - 2:** System Design Round (LLD + HLD)

Additional Features

Assessment & Progress Tracking

- Weekly quizzes on covered topics
- Coding assignments on platforms (LeetCode, CodeChef, Codeforces)
- Progress tracker with difficulty levels (Easy → Medium → Hard)

Support & Resources

- Dedicated doubt-clearing sessions
- Curated problem lists for practice
- Interview preparation checklist
- Resume building & mock interview feedback

Final Outcomes

By course completion, students will:

- Solve 500+ DSA problems across all difficulty levels
- Master data structures & algorithms for technical interviews
- Understand system design fundamentals (LLD & HLD)
- Be prepared for FAANG-level coding interviews