





Phase 1 – Strong Foundations (2–3 Weeks)

Topics: Arrays, Strings, Recursion, Sorting & Searching **roadmap.sh milestones:** Arrays  Strings 
Recursion  Sorting & Searching 

LeetCode Practice:

Arrays

1. Two Sum – [#1](#)
2. Best Time to Buy & Sell Stock – [#121](#)
3. Maximum Subarray – [#53](#)
4. Move Zeroes – [#283](#)
5. Product of Array Except Self – [#238](#)

Strings

1. Valid Palindrome – [#125](#)
2. Longest Substring Without Repeating Characters – [#3](#)
3. Group Anagrams – [#49](#)
4. Implement strStr() – [#28](#)

Recursion Basics

1. Factorial (custom) – implement in Java
2. Fibonacci Number – [#509](#)
3. Climbing Stairs – [#70](#)

Sorting & Searching

1. Binary Search – [#704](#)
2. Merge Intervals – [#56](#)
3. Search in Rotated Sorted Array – [#33](#)
4. Kth Largest Element in an Array – [#215](#)

Codeforces Examples: - 1462B – [Link](#) - 1520C – [Link](#)

Action Plan: - Watch Kunal Kushwaha's videos → implement in Java - Solve 50–70 problems from above list - Track progress in roadmap.sh checklist

Phase 2 – Core Data Structures (1.5–2 Months)

Topics: Linked List, Stack & Queue, Hashing, Binary Trees & BST, Advanced Recursion **roadmap.sh**

milestones: Linked List  Stack & Queue  Hashing  Trees  Advanced Recursion 

LeetCode Practice:

Linked List

1. Reverse Linked List – [#206](#)
2. Merge Two Sorted Lists – [#21](#)
3. Linked List Cycle – [#141](#)
4. Remove Nth Node From End – [#19](#)

Stack & Queue

1. Min Stack – [#155](#)
2. Valid Parentheses – [#20](#)
3. Sliding Window Maximum – [#239](#)
4. Implement Queue using Stacks – [#232](#)

Hashing

1. Two Sum (already #1) – practice variations
2. Subarray Sum Equals K – [#560](#)
3. Group Anagrams (already #49)
4. Top K Frequent Elements – [#347](#)

Binary Trees & BST

1. Maximum Depth of Binary Tree – [#104](#)
2. Diameter of Binary Tree – [#543](#)
3. Lowest Common Ancestor of a BST – [#235](#)
4. Validate Binary Search Tree – [#98](#)

Advanced Recursion / Backtracking Basics

1. Subsets – [#78](#)
2. Permutations – [#46](#)
3. Combination Sum – [#39](#)

Action Plan: - Watch Kunal's videos → implement each topic in Java - Solve 150 problems in this phase - Use Babbar's playlist if a concept is unclear - Mark each topic as done in roadmap.sh

Phase 3 – Advanced & Interview-Critical (2 Months)

Topics: Graphs, DP, Greedy, Backtracking, Tries, Segment Trees **roadmap.sh milestones:** Graphs  DP 
Greedy  Backtracking  Tries  Segment Trees 

LeetCode Practice:

Graphs

1. Number of Islands – [#200](#)
2. Clone Graph – [#133](#)
3. Course Schedule – [#207](#)
4. Dijkstra Variations – [#743](#)

Dynamic Programming

1. Longest Increasing Subsequence – [#300](#)
2. 0/1 Knapsack – [#416](#)
3. Unique Paths – [#62](#)
4. Coin Change – [#322](#)

Greedy

1. Jump Game – [#55](#)
2. Gas Station – [#134](#)
3. Minimum Number of Arrows to Burst Balloons – [#452](#)

Backtracking

1. N-Queens – [#51](#)
2. Word Search – [#79](#)
3. Sudoku Solver – [#37](#)

Tries

1. Implement Trie (Prefix Tree) – [#208](#)
2. Add and Search Word – [#211](#)

Segment Trees (Basics)

1. Range Sum Query – [#307](#)

Action Plan: - Watch Kunal's advanced videos → implement in Java - Solve ~150–200 problems from Striver's SDE Sheet + LeetCode - Track roadmap.sh milestones

Phase 4 – Mastery & Interview Prep (1–1.5 Months)

Goal: Problem-solving speed, timed practice, mocks

Action Plan: - LeetCode Weekly / Biweekly contests - Codeforces Div 3/4 contests for speed + mixed problems - Revisit weak areas (DP, Graphs, Trees) - Mock interviews on Pramp / Interviewing.io - Final review of roadmap.sh checklist to ensure all topics covered

Problem Target: 500+ problems solved in total across LeetCode + Striver + Codeforces

File Name Recommendation: `DSA_FANG_Master_Roadmap_Java.pdf`

This PDF contains all phases, topic-wise breakdown, exact LeetCode problem numbers, Codeforces examples, and roadmap.sh checklist integration for a complete FAANG-ready DSA plan.