🔗 Linked Lists

Two Pointer

- Famous two pointer is the use of fast and slow pointer (also known as Floyd's Algorithm).
- If the slow pointer moves 1 step, the fast pointer moves 2 steps.

Problems:

- 876. Finding the middle node in linked list [Easy]

Getting the Kth From Last Node

- Have two pointers; one is k nodes ahead.
- When the first reaches the end, the second is k from the last.

Problems:

- Remove nth node from end of list

Detecting Cycles

- Two pointers (fast and slow), if they meet → cycle detected.

Problems:

- Detect cycles

Getting the Middle Node

- Same as above: fast moves 2x, slow moves 1x.

Problems:

- Middle node

🌳 Trees

1. Traversal Patterns

DFS (Depth-First Search)

- Preorder
- Inorder
- Postorder

BFS (Breadth-First Search)

- Level-order Traversal

LeetCode Problems:

- #94 Binary Tree Inorder Traversal (Easy)
- #144 Binary Tree Preorder Traversal (Easy)
- #145 Binary Tree Postorder Traversal (Easy)
- #102 Binary Tree Level Order Traversal (Medium)
- #637 Average of Levels in Binary Tree (Easy)

2. Binary Search Tree (BST) Operations

- Searching, Insertion, Deletion
- Validation of BST properties

LeetCode Problems:

- #98 Validate Binary Search Tree (Medium)
- #700 Search in a Binary Search Tree (Easy)
- #235 Lowest Common Ancestor of a BST (Easy)
- #530 Minimum Absolute Difference in BST (Easy)
- #108 Convert Sorted Array to BST (Easy)

3. Tree Height & Depth Problems

- Calculating height, depth, diameter
- Balanced vs Unbalanced trees

LeetCode Problems:

- #104 Maximum Depth of Binary Tree (Easy)
- #110 Balanced Binary Tree (Easy)
- #543 Diameter of Binary Tree (Easy)
- #111 Minimum Depth of Binary Tree (Easy)

4. Lowest Common Ancestor (LCA)

- Finding common ancestors in binary trees or BSTs

LeetCode Problems:

- #236 Lowest Common Ancestor of a Binary Tree (Medium)
- #235 Lowest Common Ancestor of a BST (Easy)
- #1257 Smallest Common Region (Medium)
- #1602 Find Nearest Right Node in Binary Tree (Medium)

5. Path Sum Problems

- Finding paths with a given sum, counting paths, path existence

LeetCode Problems:

- #112 Path Sum (Easy)
- #113 Path Sum II (Medium)
- #437 Path Sum III (Medium)
- #257 Binary Tree Paths (Easy)
- #129 Sum Root to Leaf Numbers (Medium)

🌐 Graphs

Pattern 1: Graph Traversal (BFS and DFS)

Explanation:

- Visit all nodes
- Use for connectivity, shortest paths (unweighted), connected components

LeetCode Problems:

- #200 Number of Islands (Medium)
- #733 Flood Fill (Easy)
- #994 Rotting Oranges (Medium)
- #547 Number of Provinces (Medium)

Pattern 2: Topological Sorting

Explanation:

- Linear ordering of nodes
- Used in task scheduling, dependency resolution

LeetCode Problems:

- #207 Course Schedule (Medium)
- #210 Course Schedule II (Medium)
- #269 Alien Dictionary (Hard, Premium)
- #802 Find Eventual Safe States (Medium)

Pattern 3: Shortest Path Algorithms

Explanation:

- BFS, Dijkstra → Minimum distance/moves

LeetCode Problems:

- #542 01 Matrix (Medium)
- #1091 Shortest Path in Binary Matrix (Medium)
- #286 Walls and Gates (Medium, Premium)
- #1926 Nearest Exit in Maze (Medium)

Pattern 4: Union-Find (Disjoint Set)

Explanation:

- Check/manage connections
- Useful for detecting cycles, groups

LeetCode Problems:

- #684 Redundant Connection (Medium)
- #547 Number of Provinces (Medium)
- #323 Number of Connected Components (Medium, Premium)
- #1319 Number of Operations to Connect Network (Medium)

 Backtracking

Pattern 1: Combinations & Subsets

Explanation:

- Explore all combinations/subsets via recursion

LeetCode Problems:

- #78 Subsets (Medium)
- #77 Combinations (Medium)
- #90 Subsets II (Medium)
- #39 Combination Sum (Medium)

Pattern 2: Permutations

Explanation:

- Generate all orderings using swaps or used[] flags

LeetCode Problems:

- #46 Permutations (Medium)
- #47 Permutations II (Medium)

- #784 Letter Case Permutation (Medium)

Pattern 3: Constraint Satisfaction (e.g., N-Queens)

LeetCode Problems:

- #79 Word Search (Medium)

Pattern 4: Partitioning Problems

LeetCode Problems:

- #131 Palindrome Partitioning (Medium)
- #93 Restore IP Addresses (Medium)

📐 Dynamic Programming

Pattern 1: 0/1 Knapsack & Subset Sum

Explanation:

- Choose subset based on max/min criteria

LeetCode Problems:

- #416 Partition Equal Subset Sum (Medium)
- #494 Target Sum (Medium)
- #474 Ones and Zeroes (Medium)

- #1049 Last Stone Weight II (Medium)

Pattern 2: Longest Increasing Subsequence (LIS)

LeetCode Problems:

- #300 Longest Increasing Subsequence (Medium)
- #673 Number of Longest Increasing Subsequences (Medium)
- #646 Maximum Length of Pair Chain (Medium)
- #368 Largest Divisible Subset (Medium)

Pattern 3: Grid or Matrix DP (Paths & Minimum Cost)

LeetCode Problems:

- #62 Unique Paths (Medium)
- #64 Minimum Path Sum (Medium)
- #931 Minimum Falling Path Sum (Medium)
- #63 Unique Paths II (Medium)

Pattern 4: DP on Strings (LCS, Edit Distance)

LeetCode Problems:

- #1143 Longest Common Subsequence (Medium)
- #583 Delete Operation for Two Strings (Medium)
- #516 Longest Palindromic Subsequence (Medium)
- #72 Edit Distance (Hard)

🌲 Tries

> ⚠️ While useful, these are rarely asked in interviews. Can be prioritized later.

1. Basic Trie Implementation (Insert/Search)

LeetCode Problems:

- #208 Implement Trie (Prefix Tree) (Medium)
- #211 Design Add and Search Words Data Structure (Medium)
- #677 Map Sum Pairs (Medium)
- #648 Replace Words (Medium)

2. Prefix Matching & Autocomplete

LeetCode Problems:

- #14 Longest Common Prefix (Easy)
- #1804 Implement Trie II (Prefix Tree) (Medium)
- #1268 Search Suggestions System (Medium)
- #745 Prefix and Suffix Search (Hard)

3. Word Search in Grids (DFS + Trie)

LeetCode Problems:

- #212 Word Search II (Hard)
- #79 Word Search (Medium)

4. Counting Unique or Distinct Patterns

LeetCode Problems:

- #1698 Count Distinct Substrings in a String (Medium)
- #421 Maximum XOR of Two Numbers in an Array (Medium)
