☑ Sliding Window: Complete Pattern Coverage with Questions

🔹 **1. Fixed Size Window (Window size = k)**

🔸 **Pattern: Max/Min/Sum in every window of size k**

Leetcode	Problem Name
239	Sliding Window Maximum
2090	K Radius Subarray Averages
1838	Frequency of Most Frequent Element
1438	Longest Subarray with Limit (max-min ≤ limit)

🔹 **2. Variable Size Window (No fixed size, expand and shrink)**

🔸 **Pattern: Longest/Shortest substring or subarray satisfying condition**

Leetcode	Problem Name
3	Longest Substring Without Repeating Characters
76	Minimum Window Substring
424	Longest Repeating Character Replacement
1004	Max Consecutive Ones III
1658	Minimum Operations to Reduce x to Zero

🔹 **3. Anagram / Permutation Based**

🔸 **Pattern: Sliding frequency map**

Leetcode	Problem Name
567	Permutation in String
438	Find All Anagrams in a String

🔹 **4. Count of Substrings/Subarrays with Some Condition**

🔸 **Pattern: Use hashmap + sliding window**

Leetcode	Problem Name
992	Subarrays with K Different Integers
904	Fruit Into Baskets (max 2 types in window)
340	Longest Substring with At Most K Distinct Chars
159	Longest Substring with At Most Two Distinct Chars

🔹 **5. Sliding Window with Deque (for max/min)**

🔸 **Pattern: Maintain decreasing/increasing deque**

Leetcode	Problem Name
239	Sliding Window Maximum
862	Shortest Subarray with Sum at Least K

🔹 **6. Sliding Window with Binary Search / Two-Pointer + Binary Thinking**

Leetcode	Problem Name
1208	Get Equal Substrings Within Budget
1456	Maximum Number of Vowels in a Substring of Given Length
1052	Grumpy Bookstore Owner

🔹 **7. 2-Pointer Sliding Window on Sorted Array**

Leetcode	Problem Name
167	Two Sum II - Input array is sorted
1004	Max Consecutive Ones III (again)
930	Binary Subarrays With Sum

✅ **Summary: Sliding Window Question Types**

Type	Problem Example (Leetcode)
Fixed Size Window	239, 1456, 1838

Type	Problem Example (Leetcode)
Variable Size Window (expand-shrink)	3, 76, 424, 1004
Character Count / Frequency Map	438, 567, 76
Count Substrings / Subarrays	992, 904, 340
Max/Min Tracking with Deque	239, 862
Binary Search Inside Window	1208, 1456
Two Pointers on Sorted Array	167, 930

☑ Two Pointer Technique: Complete Pattern Coverage

🔹 1. Two Pointers for Sorting Based Problems (Pairing/Matching)

Jahan tumhare paas sorted array ho aur tumhe **pair matching** ya **pairs ki sum** find karni ho.

Problem Type	Examples
Pair with sum equal to target	🔸 Leetcode 1 – Two Sum (if sorted)
Pair with sum closest to target	🔸 Leetcode 16 – 3Sum Closest
Two sum in a sorted array	🔸 Leetcode 167 – Two Sum II (Input array is sorted)

🔹 2. Two Pointers for Searching Subarrays/Substrings

Yeh pattern **substring ya subarray ko find karne** ke liye use hota hai, jisme tumhe **expand** aur **shrink** karna padta hai window ko.

Problem Type	Examples
Longest Substring Without Repeating Characters	🔸 Leetcode 3 – Longest Substring Without Repeating Characters
Subarray with sum equal to target	🔸 Leetcode 560 – Subarray Sum Equals K
Smallest window containing substring	🔸 Leetcode 76 – Minimum Window Substring
Find all anagrams in string	🔸 Leetcode 438 – Find All Anagrams in a String

---

🔹 3. Two Pointers for Linked Lists (Reverse / Merge)

Linked list traversal ke liye two pointers ka use hota hai. Yeh pattern mostly **reverse** ya **merge** problems ke liye hota hai.

Problem Type	Examples
Reverse Linked List	🔸 Leetcode 206 – Reverse Linked List
Middle of the Linked List	🔸 Leetcode 876 – Middle of the Linked List
Merge two sorted linked lists	🔸 Leetcode 21 – Merge Two Sorted Lists
Palindrome Linked List	🔸 Leetcode 234 – Palindrome Linked List

---

🔹 4. Two Pointers for Array Partition (Meet in Middle)

Array ko do halves me divide karke dono sides pe pointers rakhein, yeh technique **binary search** ki tarah kaam karti hai.

Problem Type	Examples
Container with most water (find max area)	🔸 Leetcode 11 – Container with Most Water
Find peak element	🔸 Leetcode 162 – Find Peak Element
Find middle element	🔸 Leetcode 167 – Two Pointers for Sorted Array Search
Maximum number of consecutive ones	🔸 Leetcode 1004 – Max Consecutive Ones III

🔹 5. Two Pointers for Array Sorting and Merging (Merging Sorted Arrays)

Merging ya **sorting** related tasks ke liye two pointers efficiently kaam karte hain.

Problem Type	Examples
Merge sorted arrays	🔸 Leetcode 88 – Merge Sorted Array
Merge intervals	🔸 Leetcode 56 – Merge Intervals
Move zeroes to end	🔸 Leetcode 283 – Move Zeroes
Dutch National Flag Problem (sort colors)	🔸 Leetcode 75 – Sort Colors

---

🔹 6. Two Pointers for Removing Duplicates/Validating

Tumhare paas array ya string hai jisme tumhe **duplicates ko remove** ya **validating condition** check karni hoti hai.

Problem Type	Examples
Remove duplicates from sorted array	🔸 Leetcode 26 – Remove Duplicates from Sorted Array
Remove element from array	🔸 Leetcode 27 – Remove Element
Valid Palindrome	🔸 Leetcode 125 – Valid Palindrome

🔹 7. Two Pointers for Finding Closest Pair or Triplets

Jab tumhe **pairs ya triplets** find karne ho, jisme kisi condition ko satisfy karte ho (jaise sum).

Problem Type	Examples
3Sum	🔸 Leetcode 15 – 3Sum

Problem Type Examples

2Sum II	🔸 Leetcode 167 – Two Sum II (input array sorted)
4Sum	🔸 Leetcode 18 – 4Sum

🧠 **Summary Table: Two Pointer Patterns**

Pattern Type	Problem Examples
Sorting & Pairing	1, 16, 167
Subarrays/Substrings Search	3, 560, 76, 438
Linked List Operations	206, 876, 21, 234
Array Partitioning	11, 162, 1004
Merging & Sorting Arrays	88, 56, 283, 75
Removing Duplicates/Validating	26, 27, 125
Finding Closest Pair/Triplets	15, 167, 18

☑ Prefix Sum: Complete Pattern Coverage (with Examples)

🔹 **Prefix Sum Ka Basic Idea:**

Agar tumhare paas ek array hai:

makefile

CopyEdit

nums = [2, 4, 1, 3]

prefix[i] = nums[0] + nums[1] + ... + nums[i]

To kisi range (i to j) ka sum hoga:

swift

CopyEdit

rangeSum(i, j) = prefix[j] - prefix[i-1]

🔸 **1. 1D Prefix Sum – Subarray Sum Related**

Problem Type	Leetcode Examples
Sum of all subarrays	🔸 N/A (concept based)
Subarray sum equals K	🔸 560 – Subarray Sum Equals K
Number of subarrays with sum divisible by K	🔸 974 – Subarray Sums Divisible by K
Min size subarray sum	🔸 209 – Minimum Size Subarray Sum
Maximum sum subarray of fixed size	🔸 Sliding Window version also

---

🔸 **2. Prefix Sum with HashMap – For Count Tracking**

Jab **negative numbers** ho aur tumhe count ya frequency track karna ho

Problem Type	Leetcode Examples
Count subarrays with sum K	🔸 560 – Subarray Sum Equals K
Binary Subarrays with Sum	🔸 930 – Binary Subarrays With Sum
Count nice subarrays (odd count)	🔸 1248 – Count Number of Nice Subarrays

---

🔸 **3. 2D Prefix Sum (Matrix)**

Jab tumhe kisi **2D matrix me rectangular area** ka sum nikalna ho

Problem Type	Leetcode Examples
Matrix sum region	304 – Range Sum Query 2D - Immutable
Num of submatrices that sum to target	1074 – Number of Submatrices That Sum to Target

---

4. Difference Array (Inverse of Prefix Sum)

Jab tumhe **range updates in O(1)** karne ho

Problem Type	Leetcode Examples
Flight bookings with range updates	1109 – Corporate Flight Bookings
Range addition	370 – Range Addition

---

5. Prefix Sum on Characters (Strings)

Jab tumhe frequency of characters ya palindromic substrings count karni ho

Problem Type	Leetcode Examples
Palindromic substrings	1177 – Can Make Palindrome from Substring
Count vowels/consonants in range	1456 – Max Vowels in Substring of Given Length

---

6. Prefix XOR / Prefix Product

Jab sum ke jagah tumhe **XOR** ya **Product** nikalna ho

Problem Type	Leetcode Examples
XOR of all subarrays	1310 – XOR Queries of a Subarray
Product of subarray except self	238 – Product of Array Except Self (variant)

---

7. Prefix Sum with Sliding Window

Jab tumhe **fixed window size** ke saath optimized sum calculation chahiye

Problem Type	Leetcode Examples
K Radius Subarray Averages	2090 – K Radius Subarray Averages

Problem Type	Leetcode Examples
Sliding window sum problems	🔸 643 – Maximum Average Subarray I

🔸 8. Prefix Sum with Binary Search

Jab tum **cumulative sum** ka use karke **binary search** se kuch threshold find karte ho

Problem Type	Leetcode Examples
Minimum speed to eat bananas	🔸 875 – Koko Eating Bananas (with prefix-like logic)
Min days to make bouquet	🔸 1482 – Minimum Number of Days to Make m Bouquets

---

🧠 Summary Table: Prefix Sum Variations

Pattern Type	Example Problems
1D prefix + subarray sum	560, 974, 209
Hashmap + prefix for count	560, 930, 1248
2D prefix sum (matrix)	304, 1074
Difference array (range updates)	1109, 370
Prefix on strings	1177, 1456
Prefix XOR/product	1310, 238
Prefix + sliding window	2090, 643
Prefix + binary search	875, 1482

☑ Kadane's Algorithm: Core Idea

Simple Kadane ka kaam hota hai:
"Max sum of any contiguous subarray in O(n) time"

cpp

CopyEdit

```
int maxSubArray(vector<int>& nums) {
    int currSum = nums[0], maxSum = nums[0];
    for (int i = 1; i < nums.size(); i++) {
        currSum = max(nums[i], currSum + nums[i]);
        maxSum = max(maxSum, currSum);
    }
    return maxSum;
}
```

🔰 Kadane's Algorithm Variations (with Problem Examples)

🔹 1. Basic Kadane – Maximum Subarray Sum

Type	Examples
Basic version	🔸 Leetcode 53 – Maximum Subarray
Max subarray with indices	🔸 Track start and end index manually

---

🔹 2. Kadane with All Negative Elements

Basic Kadane works here too, but initialize carefully (currSum = maxSum = nums[0])

Type	Examples
All elements negative	🔸 Edge case for Leetcode 53
Return empty array if maxSum < 0	🔸 Custom handling required

🔹 3. Kadane on Circular Array

Circular array mein max sum ya to normal hoga ya fir totalSum - minSubarraySum

Type	Examples
Max sum in circular array	🔸 Leetcode 918 – Maximum Sum Circular Subarray

---

🔹 4. Kadane with Size Constraint (Fixed or Min/Max Length)

Kadane doesn't directly apply — combine it with sliding window or prefix sum

Type	Examples
Fixed size k	🔸 Leetcode 643 – Max Average Subarray I
Minimum size subarray with sum ≥ target	🔸 Leetcode 209 – Minimum Size Subarray Sum

---

🔹 5. Kadane with 2D Matrix (Max Submatrix Sum)

Treat each pair of rows as 1D array and apply Kadane

Type	Examples
Max submatrix sum	🔸 Leetcode 363 – Max Sum of Rectangle No Larger Than K
Max rectangle sum (brute + kadane)	🔸 InterviewBit / GFG variant problems

---

🔹 6. Kadane with Deletion / Modification Allowed

Jab ek element delete karne ya modify karne ka option ho

Type	Examples
Max subarray sum with one deletion	🔸 Leetcode 1186 – Maximum Subarray Sum with One Deletion
Max sum after at most one flip	🔸 GFG – Flip bits to maximize 1s

---

🔹 7. Kadane with XOR / Bitwise Modifications

Jab normal sum ke jagah XOR use karna ho (not direct Kadane, but inspired pattern)

Type	Examples
Max XOR subarray	🔸 GFG – Maximum XOR Subarray
Max product subarray	🔸 Leetcode 152 – Maximum Product Subarray (similar but needs 2 vars: max & min)

🔹 8. Kadane with Non-Contiguous Element Selection

Jab elements contiguous nahi hote but sum maximize karna hai

Type	Examples
Max sum with no adjacent elements	🔸 House Robber – Leetcode 198
Max sum in non-adjacent subarrays	🔸 GFG – Maximum sum such that no two elements are adjacent

---

🧠 Summary Table: Kadane's Variations

Pattern Type	Problem Examples
Basic subarray sum	Leetcode 53
All negatives	Leetcode 53 edge case
Circular subarray	Leetcode 918
Size constraint	Leetcode 209, 643
2D matrix version	Leetcode 363
One deletion / flip allowed	Leetcode 1186, GFG Flip Bits
Kadane with XOR / Product variation	Leetcode 152, Max XOR Subarray (GFG)
Non-contiguous max sum	Leetcode 198 (House Robber), GFG variants

☑ Hashing (HashMap / HashSet): Complete Variations Breakdown

🔹 1. Frequency Counting (Most Common Use of HashMap)

Jab tumhe kisi array ya string me **element ya character ki frequency** nikalni ho

Problem Type	Leetcode Examples
Count character frequency	🔸 387 – First Unique Character in a String
Anagrams check	🔸 242 – Valid Anagram
Top K frequent elements	🔸 347 – Top K Frequent Elements
Majority element (frequency > n/2)	🔸 169 – Majority Element
Group anagrams	🔸 49 – Group Anagrams

✅ Can be used with:

- **unordered_map<T, int>** (in C++)
- **map** if sorting is needed

🔹 2. Subarrays with Given Sum (Prefix Sum + HashMap)

Jab tumhe subarrays dhoondhne ho **jinka sum target ke equal ho**, including **negatives**

Problem Type	Leetcode Examples
Count of subarrays with sum K	🔸 560 – Subarray Sum Equals K
Binary subarrays with sum	🔸 930 – Binary Subarrays With Sum
Subarrays divisible by K	🔸 974 – Subarray Sums Divisible by K
Count nice subarrays (with k odd numbers)	🔸 1248 – Count Number of Nice Subarrays

✅ Use prefix sum + hash map:

cpp

CopyEdit

```
unordered_map<int, int> prefixFreq;
prefixFreq[0] = 1; // base case
int sum = 0, count = 0;
for (int i : nums) {
```

```
    sum += i;
    count += prefixFreq[sum - k];
    prefixFreq[sum]++;
}
```

🔹 3. HashSet for Duplicates / Uniqueness

Jab tumhe kisi element ka **duplicate** ya **presence** check karna ho

Problem Type	Leetcode Examples
Contains duplicate	🔸 217 – Contains Duplicate
Happy number	🔸 202 – Happy Number
Longest consecutive sequence	🔸 128 – Longest Consecutive Sequence
Check if array is permutation	🔸 GFG – Check if array contains all elements from 1 to n

✅ Use unordered_set<T> to store seen elements in O(1) time.

🔹 4. Hashing in Sliding Window (Character Frequency)

Mostly used for **anagrams**, **smallest window**, and **fixed-length substring checks**

Problem Type	Leetcode Examples
Find all anagrams in string	🔸 438 – Find All Anagrams in a String
Minimum window substring	🔸 76 – Minimum Window Substring
Check inclusion (permutation)	🔸 567 – Permutation in String

✅ Use 2 hashmaps: need and window, and two pointers.

🔹 5. Hashing with Index Mapping

Jab tumhe kisi element ka **last seen index** ya **position** store karna ho

Problem Type	Leetcode Examples
Two sum	🔸 1 – Two Sum
Longest substring without repeat	🔸 3 – Longest Substring Without Repeating Characters
Subarray sum equals k (tracking index)	🔸 560 variant

Problem Type	Leetcode Examples
Count substrings with equal 0s and 1s	🔸 GFG variant – Equal 0s and 1s

---

🔹 6. Hashing in 2D Grids or Pairs

Jab tum grid ko row-wise hash karo ya pair ko track karo

Problem Type	Leetcode Examples
Equal row and column pairs	🔸 2352 – Equal Row and Column Pairs
Count number of pairs with sum	🔸 1, 170 – Two Sum III
Custom hashing for pair of coordinates	🔸 Hash<pair<int, int>> in unordered_map

---

🔹 7. Hashing for Mathematical Reductions

Jab tum **ratios**, **gcd**, **remainders**, etc. ko hash karke count karte ho

Problem Type	Leetcode Examples
Boomerang count	🔸 447 – Number of Boomerangs
Fraction simplification	🔸 Custom problems (Hash slope as key)
Subarrays with same remainder	🔸 974 – Subarray Sums Divisible by K

🧠 Summary Table: Hashing Patterns

Variation Type	Leetcode Problems
Frequency count	242, 387, 169, 347, 49
Subarrays with sum using prefix	560, 930, 974, 1248
HashSet for uniqueness	217, 202, 128
Sliding window with hash	76, 438, 567
Index-based hashing	1, 3, GFG – equal 0s/1s
Grid/pair hashing	2352, pair hashing problems
Hash with math	447, 974, slope/hash ratio problems

☑ Binary Search: Complete Pattern Coverage (with Variations)

Yahaan maine har **binary search pattern** ko **examples ke saath** cover kiya hai:

🔹 1. Classic Binary Search (Sorted Array, Find Target)

Problem Type	Examples
Find target index	🔸 Leetcode 704 – Binary Search
Find insert position	🔸 Leetcode 35 – Search Insert Position
First or last occurrence	🔸 Leetcode 34 – Find First and Last Position of Element in Sorted Array

---

🔹 2. Binary Search on Rotated Sorted Array

Problem Type	Examples
Search in rotated array	🔸 Leetcode 33 – Search in Rotated Sorted Array
With duplicates	🔸 Leetcode 81 – Search in Rotated Sorted Array II

🔹 3. Binary Search for Answer (Search Space Optimization)

Jab input sorted nahi hota, lekin tumhare paas ek **range** hoti hai jisme se answer nikalna hota hai — usually min/max value ya capacity

Problem Type	Examples
Min capacity to ship packages in D days	🔸 Leetcode 1011 – Capacity To Ship Packages Within D Days
Koko eating bananas	🔸 Leetcode 875 – Koko Eating Bananas
Min number of days to make bouquets	🔸 Leetcode 1482 – Minimum Number of Days to Make m Bouquets
Allocate books / painters partition	🔸 Leetcode 410 – Split Array Largest Sum

---

🔹 4. Binary Search in 2D Matrix

Problem Type	Examples
Matrix as 1D array	🔸 Leetcode 74 – Search a 2D Matrix
Matrix with row-wise + column-wise sorted	🔸 Leetcode 240 – Search a 2D Matrix II

🔹 5. Binary Search on Answers With Predicate Function (YES/NO questions)

Tum answer ko **check()** function se validate karte ho

Problem Type	Examples
Smallest distance pair	🔸 Leetcode 719 – Find K-th Smallest Pair Distance
Aggressive cows (on SPOJ)	🔸 Find max minimum distance between stalls
Minimum time to paint boards	🔸 Leetcode 1578 Variant – with painters & boards

---

🔹 6. Lower Bound / Upper Bound Based Search

Problem Type	Examples
Lower bound of target	🔸 Leetcode 34 (again)
Count elements < target	🔸 Leetcode 300 – Longest Increasing Subsequence (uses lower_bound)

🔹 7. Binary Search with Bit Manipulation / Custom Condition

Problem Type	Examples
Find single non-duplicate element	🔸 Leetcode 540 – Single Element in a Sorted Array
Peak element	🔸 Leetcode 162 – Find Peak Element
Find mountain peak	🔸 Leetcode 852 – Peak Index in a Mountain Array
Min element in rotated sorted array	🔸 Leetcode 153, 154 (with duplicates)

🧠 Quick Summary: Binary Search Patterns

Pattern Type	Example Problems
Classic (sorted array)	704, 35, 34
Rotated sorted array	33, 81

Pattern Type	Example Problems
Search on answer space	875, 1011, 410, 1482
Binary search in 2D matrix	74, 240
Predicate/decision-based	719, Aggressive Cows, Paint Boards
Lower/upper bound usage	34, 300
With bit tricks/custom condition	540, 162, 153, 154

☑ Linked List: Complete Pattern Coverage

📌 **Basics:**

- **Node structure** – val, next, (sometimes prev)
- **Traversal**, **insertion**, **deletion**

🔹 **1. Reversal Variants (Most Common Pattern)**

Variation Type	Leetcode Problems
Reverse entire list	🔸 206 – Reverse Linked List
Reverse first k nodes	🔸 92 – Reverse Linked List II
Reverse in k-group	🔸 25 – Reverse Nodes in k-Group
Reverse alternate k-group	🔸 GFG/Custom Variant

👉 Key Concept: 3-pointer approach (prev, curr, next)

🔹 **2. Cycle Detection & Removal**

Variation Type	Leetcode Problems
Detect cycle	🔸 141 – Linked List Cycle
Find starting node of cycle	🔸 142 – Linked List Cycle II
Remove loop	🔸 GFG – Remove loop in Linked List

👉 Use **Floyd's Tortoise and Hare Algorithm**

🔹 **3. Palindrome / Middle / Length Based**

Variation Type	Leetcode Problems
Find middle node	🔸 876 – Middle of the Linked List
Check if palindrome	🔸 234 – Palindrome Linked List
Split into parts	🔸 725 – Split Linked List in Parts

👉 Mostly use **fast and slow pointer** or **stack + two pointer**

🔹 4. Merge / Sort / Rearrangement

Variation Type	Leetcode Problems
Merge two sorted lists	🔸 21 – Merge Two Sorted Lists
Merge k sorted lists	🔸 23 – Merge k Sorted Lists (heap)
Sort linked list	🔸 148 – Sort List (merge sort)
Rearrange list alternately	🔸 143 – Reorder List
Zig-zag rearrange	🔸 GFG variant

🔹 5. Add / Multiply Numbers as Lists

Variation Type	Leetcode Problems
Add two numbers (normal order)	🔸 2 – Add Two Numbers
Add two numbers (reverse order)	🔸 445 – Add Two Numbers II
Multiply two numbers	🔸 GFG variant – Multiply Linked Lists

👉 Convert to integers or use **stack-based reverse addition**

🔹 6. Deletion / Duplication / Remove Nodes

Variation Type	Leetcode Problems
Delete node in O(1)	🔸 237 – Delete Node in a Linked List
Remove nth node from end	🔸 19 – Remove N-th Node From End of List
Remove duplicates	🔸 83, 82 – Remove Duplicates from Sorted List
Delete all occurrences	🔸 GFG variant

🔹 7. Intersection / Merge Point / Loop Joining

Variation Type	Leetcode Problems
Intersection of two lists	🔸 160 – Intersection of Two Linked Lists
Merge point in Y-shaped list	🔸 GFG – Find merge point of two lists

Variation Type	Leetcode Problems
Find overlapping node	🔸 Custom problems

👉 Use **length difference method** or **HashSet**

🔹 8. Copy Linked List with Random Pointers

Variation Type	Leetcode Problems
Clone list with random pointer	🔸 138 – Copy List with Random Pointer

👉 Use **hashmap** or **interleaving nodes trick**

🔹 9. Doubly & Circular Linked List

Variation Type	Platform
Insert/Delete at head/tail	🔸 GFG – Doubly Linked List
Circular traversal / insertion	🔸 GFG – Circular List insertion
LRU Cache (Doubly + Map)	🔸 146 – LRU Cache

🔹 10. Flatten / Deep Structures

Variation Type	Leetcode Problems
Flatten multilevel doubly list	🔸 430 – Flatten a Multilevel Doubly Linked List
Flatten binary tree to list	🔸 114 – Flatten Binary Tree to Linked List

👉 Use **recursion** or **stack**

🧠 Summary Table

Pattern Type	Examples / Leetcode
Reverse variants	206, 92, 25
Cycle detection	141, 142
Palindrome / middle	234, 876
Merge / sort	21, 23, 148

Pattern Type	Examples / Leetcode
Add/multiply numbers	2, 445
Delete / duplicates	19, 237, 82
Intersection / merge point	160, GFG
Copy list with random pointer	138
Doubly/circular list	146 (LRU), GFG
Flatten structure	430, 114

☑ Stacks & Queues: Complete Variations for Interview

🔹 1. Basic Stack / Queue Operations

Problem Type	Leetcode / GFG Examples
Implement stack using queues	🔸 225 – Implement Stack using Queues
Implement queue using stacks	🔸 232 – Implement Queue using Stacks
Design circular queue	🔸 622 – Design Circular Queue
Custom queue using linked list	🔸 GFG – Queue using LL

🔹 2. Next Greater / Smaller Element (Monotonic Stack)

🔥 Most important category — used in arrays, histograms, stock span, etc.

Problem Type	Leetcode / GFG Examples
Next Greater Element	🔸 496 – NGE I, 🔸 503 – NGE II
Next Smaller Element	🔸 GFG – NLE / NSE
Stock span problem	🔸 GFG – Stock Span
Largest rectangle in histogram	🔸 84 – Largest Rectangle in Histogram
Max area in binary matrix	🔸 85 – Maximal Rectangle
Daily Temperatures	🔸 739 – Daily Temperatures
Asteroid Collision	🔸 735 – Asteroid Collision

👉 Use **monotonic stack** (increasing or decreasing)
👉 Push indices instead of values when needed

🔹 3. Infix / Postfix / Prefix Evaluation

Interview favorite from classical DSA

Problem Type	Leetcode / GFG Examples
Evaluate postfix	🔸 GFG – Evaluate Postfix Expression
Evaluate prefix	🔸 GFG – Evaluate Prefix

Problem Type	Leetcode / GFG Examples
Infix to postfix / prefix	🔸 GFG – Infix to Postfix
Evaluate infix expression	🔸 GFG – Infix Evaluation

👉 Use stack for both **operands and operators**

🔹 **4. Min/Max Stack Design**

Problem Type	Leetcode Examples
Design Min Stack	🔸 155 – Min Stack
Design Max Stack	🔸 GFG / Custom
Get minimum in O(1)	🔸 155 – Min Stack

👉 Use extra stack for tracking min/max

🔹 **5. Balanced Parentheses / Valid Expressions**

Problem Type	Leetcode Examples
Valid parentheses	🔸 20 – Valid Parentheses
Minimum add to make valid	🔸 921 – Minimum Add to Make Parentheses Valid
Longest valid parentheses	🔸 32 – Longest Valid Parentheses
Remove invalid parentheses	🔸 301 – Remove Invalid Parentheses

👉 Stack + counter logic
👉 Sometimes BFS for multiple valid outputs (301)

🔹 **6. Sliding Window using Deque (Monotonic Queue)**

Very efficient for **max/min in fixed size window**

Problem Type	Leetcode Examples
Sliding window max	🔸 239 – Sliding Window Maximum
First negative in every window	🔸 GFG – First Negative Integer
Sliding window min/max	🔸 GFG Variants

👉 Use **deque** to maintain max/min window

🔹 **7. Queue Variants (Circular, Monotonic, Two Queues)**

Problem Type	Leetcode / GFG Examples
Circular queue	🔸 622 – Design Circular Queue
Monotonic queue	🔸 1425 – Constrained Subsequence Sum
Queue using 2 stacks	🔸 232 – Queue Using Stacks

🔹 **8. LRU Cache / LFU Cache (Deque + Hashmap)**

Most common real-world stack/queue combo question

Problem Type	Leetcode Examples
LRU Cache	🔸 146 – LRU Cache
LFU Cache	🔸 460 – LFU Cache

👉 Use **unordered_map + doubly linked list** for O(1) operations

---

🔹 **9. Stack for Recursion Simulation / Iterative DFS**

Problem Type	Leetcode / GFG Examples
Flatten binary tree to list	🔸 114 – Flatten Binary Tree
Binary tree inorder traversal	🔸 94 – Inorder Iterative
Evaluate expression tree	🔸 Custom
Tower of Hanoi (stack-based)	🔸 GFG variant

🔹 **10. Custom Stack / Queue Design**

Problem Type	Leetcode Examples
Implement Stack With Increment	🔸 1381 – Design a Stack With Increment
Custom queue operations	🔸 641 – Design Circular Deque

🧠 **Summary Table**

Pattern	Key Problems
Monotonic Stack (NGE/NSE)	496, 503, 739, 84
Infix/Postfix Evaluation	GFG classic
Min/Max Stack	155, Custom
Valid Parentheses	20, 32, 301
Sliding Window (Deque)	239, GFG
Stack Using Queues / Vice Versa	225, 232
LRU / LFU Cache	146, 460
Custom Stack Design	1381, 641

☑ Recursion + Backtracking: All Key Variations (with Leetcode Examples)

🔹 1. Subsets / Subsequence Pattern

➡ Binary choice: include/exclude

Problem Type	Leetcode Examples
All subsets	🔸 78 – Subsets
Subsets with duplicates	🔸 90 – Subsets II
All subsequences (string or array)	🔸 GFG – All Subsequences
K-sum subsets	🔸 416 – Partition Equal Subset Sum

👉 Pattern: pick / not pick approach

🔹 2. Permutations Pattern

➡ Har position pe sab options try karo (swap or visited array)

Problem Type	Leetcode Examples
All permutations	🔸 46 – Permutations
Permutations with duplicates	🔸 47 – Permutations II
String permutations	🔸 GFG – All Permutations of a string

👉 Use used[] or swap-based recursion

🔹 3. Combination / Combination Sum Pattern

➡ Recursive + backtrack with for-loop

Problem Type	Leetcode Examples
Combination sum (repetition allowed)	🔸 39 – Combination Sum
Combination sum (no repetition)	🔸 40 – Combination Sum II
K-combinations	🔸 77 – Combinations
Letter combinations from digits	🔸 17 – Letter Combinations of Phone Number

👉 Backtrack inside loop, handle duplicates carefully

🔹 **4. Palindrome / Partitioning Problems**

➡️ Break string in all possible valid ways

Problem Type	Leetcode Examples
Palindrome partitioning	🔸 131 – Palindrome Partitioning
Palindrome partitioning II (min cuts)	🔸 132 – Palindrome Partitioning II
Restore IP addresses	🔸 93 – Restore IP Addresses

👉 Backtrack after checking if prefix is valid

🔹 **5. Sudoku / Grid / Matrix-Based Backtracking**

➡️ Try all numbers/paths recursively

Problem Type	Leetcode Examples
Sudoku solver	🔸 37 – Sudoku Solver
N-Queens	🔸 51 – N-Queens
Rat in a Maze / Knight's Tour	🔸 GFG variants
Word Search	🔸 79 – Word Search

👉 Row/col/diag/visited constraints + recursion

🔹 **6. Expression Building / Evaluate All Possibilities**

➡️ Backtrack with operators and partitions

Problem Type	Leetcode Examples
Add operators to form target	🔸 282 – Expression Add Operators
Diff ways to add parentheses	🔸 241 – Different Ways to Add Parentheses

👉 DFS on all operator positions, return list of values

🔹 **7. Partitioning & Palindromic Problems**

➡️ Divide set/array/string recursively

Problem Type	Leetcode Examples
Partition into k subsets	🔸 698 – Partition to K Equal Sum Subsets
Palindromic partitions	🔸 131 – Palindrome Partitioning
String segmentation	🔸 139 – Word Break

🔹 **8. Path-Finding in Grid / Maze**

➡️ DFS recursion with direction movement

Problem Type	Leetcode Examples
Word search	🔸 79 – Word Search
Unique paths III	🔸 980 – Unique Paths III
Rat in maze	🔸 GFG – Maze Problem

👉 Use dx[], dy[] for 4-way movement

🔹 **9. Generate Parentheses / Valid Expressions**

➡️ Balanced recursion with open/close count

Problem Type	Leetcode Examples
Generate valid parentheses	🔸 22 – Generate Parentheses
Remove invalid parentheses	🔸 301 – Remove Invalid Parentheses
Generate binary strings	🔸 GFG variants

🔹 **10. Backtracking with Bitmask (Advanced)**

➡️ Use bits to track state

Problem Type	Leetcode Examples
Count arrangements	🔸 526 – Beautiful Arrangement
Tiling problem	🔸 790 – Domino and Tromino Tiling
Matchsticks to square	🔸 473 – Matchsticks to Square

🧠 Summary Table

Pattern Type	Key Problems
Subsets / Subsequences	78, 90
Permutations	46, 47
Combination Sum	39, 40, 77
Palindromic Partition	131, 132
Sudoku / N-Queens	37, 51
Expression Add	282, 241
Partition into Subsets	698
Word Search / Grid	79, 980
Generate Parentheses	22, 301
Bitmasking + Recursion	473, 526

☑ Dynamic Programming (DP): All Important Patterns & Variations

---

🔹 1. Fibonacci / Climbing Stairs (1D DP)

➡ Most basic form — dp[i] = dp[i-1] + dp[i-2]

Problem Type	Leetcode Examples
Climbing Stairs	🔸 70 – Climbing Stairs
Frog Jump	🔸 GFG / Custom
Min Cost Climbing Stairs	🔸 746 – Min Cost Climbing Stairs
House Robber (gap DP)	🔸 198 – House Robber

🔹 2. 0/1 Knapsack Pattern

➡ Choice to **pick or not pick** items

Problem Type	Leetcode / GFG
0/1 Knapsack	🔸 GFG – Classic
Subset sum	🔸 GFG – Subset Sum
Partition equal subset sum	🔸 416
Count of subset sum	🔸 GFG

👉 Recursive → Memo → Tabulation → Space Optimized

🔹 3. Unbounded Knapsack Pattern

➡ Infinite supply of items

Problem Type	Leetcode / GFG
Coin change (min coins)	🔸 322
Coin change (total ways)	🔸 518
Rod cutting	🔸 GFG – Rod Cutting
Complete knapsack	🔸 GFG variants

🔹 **4. Subsequence-Based DP (LCS, LIS, etc.)**

➡️ Compare two strings or indices

Problem Type	Leetcode
Longest Common Subsequence (LCS)	🔸 1143
Longest Palindromic Subseq	🔸 516
Longest Increasing Subsequence	🔸 300
Edit Distance	🔸 72
Delete Operation for Two Strings	🔸 583
Min insertions to make palindrome	🔸 1312

👉 Use dp[i][j] → strings/index combos
👉 Can convert to 1D for LIS

🔹 **5. Matrix DP (2D Grids)**

➡️ DP on grids → directions: right/down or 4-way

Problem Type	Leetcode
Unique Paths	🔸 62, 63
Minimum Path Sum	🔸 64
Maximal Square	🔸 221
Cherry Pickup	🔸 741
Grid with obstacles	🔸 63
Gold Mine Problem	🔸 GFG

🔹 **6. Palindromic DP**

➡️ Substring-based DP + palindrome check

Problem Type	Leetcode
Palindrome Partitioning II	🔸 132

Problem Type	Leetcode
Longest Palindromic Subsequence	516
Count palindromic substrings	647
Longest Palindromic Substring	5

🔹 7. Partition / Subset DP

➡️ Divide array/string into valid parts

Problem Type	Leetcode
Word Break	139
Partition Equal Subset Sum	416
Palindrome Partitioning II	132
Array Partition	GFG variants

🔹 8. Digit DP / Bitmask DP (Advanced)

➡️ Use bitmasking for subset states, or digit limits

Problem Type	Leetcode
Matchsticks to Square	473
Count numbers with unique digits	357
Tiling problem	790
Beautiful Arrangement	526
Traveling Salesman	GFG – Bitmasking DP

🔹 9. Game Theory / MinMax DP

➡️ DP[i][j] represents best score from i to j

Problem Type	Leetcode
Predict the Winner	486
Stone Game	877

Problem Type	Leetcode
Burst Balloons	🔸 312
Removal Game	🔸 GFG

🔹 **10. Tree DP (on N-ary or binary trees)**

➡️ Apply DP in postorder fashion

Problem Type	Leetcode
House Robber III	🔸 337
Diameter of Binary Tree	🔸 543
Max path sum in tree	🔸 124
Longest ZigZag Path	🔸 1372

🔹 **11. DP with State Compression (Advanced)**

➡️ Track states in compressed form

Problem Type	Leetcode
Paint House	🔸 256, 265
Job Scheduling	🔸 1235
Domino and Tromino Tiling	🔸 790
Super Egg Drop	🔸 887

🧠 **Summary Table**

Pattern	Key Problems
Fibonacci/1D	70, 746
0/1 Knapsack	416, GFG
Unbounded Knapsack	322, 518
Subsequence (LCS/LIS)	1143, 300
Matrix/Grid DP	62, 64

Pattern	Key Problems
Palindromic	5, 516, 647
Subset/Partition	139, 416
Bitmask/Digit DP	473, 357
Game Theory	486, 877, 312
Tree DP	337, 124
State Compression	790, 887

✅ **Practice Strategy**

1. Start with **1D → 2D → LCS/LIS → Grid DP**
2. Master **Knapsack (0/1 + Unbounded)**
3. Learn **DP on Strings, Partitions, and Palindromes**
4. Finally try **Game DP, Bitmasking & Tree DP**

📌 **Bonus Tip**:
Har DP question me pehle ye 4 cheeze identify karo:

1. **State** (dp[i], dp[i][j] etc.)
2. **Choice** (pick/skip, move right/down, etc.)
3. **Base case**
4. **Transition / Recurrence Relation**

☑ Greedy Algorithms – All Important Patterns & Variations

🔹 1. Activity / Interval Scheduling Problems

➡ Sort intervals by end/start time and pick non-overlapping ones

Problem Type	Leetcode
Activity Selection	🔸 GFG – Classic
Non-overlapping Intervals	🔸 435
Minimum number of arrows	🔸 452
Meeting Rooms I/II	🔸 252, 253
Merge Intervals (greedy + sort)	🔸 56

👉 Sort + Choose max activities without clash

🔹 2. Jump Game Type

➡ Maximize reach at each index

Problem Type	Leetcode
Jump Game I (can reach end)	🔸 55
Jump Game II (min jumps)	🔸 45
Minimum number of platforms	🔸 GFG

👉 Use farthest variable to track max reachable index

🔹 3. Minimum Number of Operations / Steps

➡ Minimize steps with best greedy choice

Problem Type	Leetcode
Gas Station Circuit	🔸 134
Candy Distribution	🔸 135
Monotone Increasing Digits	🔸 738

Problem Type	Leetcode
Remove K Digits	🔸 402

👉 Many problems reduce to “do optimal now” logic

🔹 **4. Huffman Encoding / Optimal Merge Pattern**

➡️ Minimize cost using heap/priority queue

Problem Type	Leetcode
Huffman Coding	🔸 GFG
Minimum cost to connect ropes	🔸 GFG
Merge files at min cost	🔸 GFG variants

👉 Use min-heap for lowest cost pairwise merge

🔹 **5. Greedy + Sorting Problems**

➡️ Sort first and apply greedy logic

Problem Type	Leetcode
Assign Cookies	🔸 455
Boats to Save People	🔸 881
Bag of Tokens	🔸 948
Two City Scheduling	🔸 1029

👉 Sorting opens up greedy opportunity

🔹 **6. Greedy + Heap (Priority Queue)**

➡️ Pick best option available at the moment

Problem Type	Leetcode
IPO	🔸 502
Task Scheduler	🔸 621
Reorganize String	🔸 767

Problem Type	Leetcode
Frequency Stack	🔸 895

👉 Max-heaps often used for choosing largest freq/value

🔹 7. Coin Change Type (Classic Greedy)

➡️ Use highest denomination first

Problem Type	Leetcode
Coin Change (greedy fails)	🔸 322 (DP required)
Valid greedy coin systems	🔸 GFG
Minimum coins (valid denominations only)	🔸 GFG

📌 Greedy works only when coin denominations follow **canonical system**

🔹 8. Fractional Knapsack (Not 0/1)

➡️ Take maximum value per weight unit

Problem Type	Leetcode / GFG
Fractional Knapsack	🔸 GFG
Fractional Job Scheduling	🔸 GFG

👉 Sort by value/weight, take partial if needed

🔹 9. Greedy in Strings / Lexicographical

➡️ Remove characters to get smallest or largest string

Problem Type	Leetcode
Remove K Digits	🔸 402
Smallest Subsequence of Distinct Chars	🔸 1081
Lexicographically Smallest Subsequence	🔸 316
Construct K Palindrome Strings	🔸 1400

🔹 10. Scheduling Jobs / Tasks / Intervals

➡️ Assign jobs in such a way that cost/time is minimized

Problem Type	Leetcode
Job Sequencing Problem	🔸 GFG
Course Schedule III	🔸 630
Task Scheduler	🔸 621
Meeting Rooms II	🔸 253

🧠 Summary Table

Pattern	Key Problems
Interval Scheduling	435, 452, 252
Jump Game	45, 55
Min Steps	135, 134, 738
Huffman/Optimal Merge	GFG
Greedy + Sort	455, 1029
Greedy + Heap	621, 502, 767
Coin Change	322, GFG
Fractional Knapsack	GFG
Lexicographical Greedy	316, 402
Job Scheduling	630, 253

✅ Practice Strategy

1. Master **Activity/Interval scheduling** (sort + select)
2. Learn **Jump Game & Min Steps logic**
3. Understand **greedy heap use cases**
4. Try **Huffman / fractional knapsack**
5. Explore **Greedy in strings and task assignment**

📌 **Pro Tip**: Greedy tabhi kaam karta hai jab har local optimal choice se global optimal ban sakta ho. Greedy ≠ always correct → **proof / counterexample** dekhna zaroori hai (specially in coin change).

☑ Trees – Har Important Variation aur Pattern (Interview Focused)

🔹 1. Tree Traversals (Recursive + Iterative)

➡ DFS (Pre, In, Post), BFS (Level Order)

Traversal Type	Leetcode Problems
Preorder	🔸 144 – Binary Tree Preorder Traversal
Inorder	🔸 94 – Binary Tree Inorder Traversal
Postorder	🔸 145 – Postorder Traversal
Level Order	🔸 102 – Level Order Traversal
Zigzag Level Order	🔸 103
Iterative Inorder/Postorder	🔸 94, 145

🔹 2. Binary Search Tree (BST) Based

➡ Leverage left < root < right

Problem Type	Leetcode
Validate BST	🔸 98
Lowest Common Ancestor (BST)	🔸 235
Search in BST	🔸 700
Insert/Delete in BST	🔸 701, 450
Kth Smallest Element	🔸 230
Convert Sorted Array/List to BST	🔸 108, 109

🔹 3. Tree Path Sum Problems

➡ Path from root to node or node-to-node

Problem Type	Leetcode
Path Sum I / II	🔸 112, 113

Problem Type	Leetcode
Binary Tree Maximum Path Sum	124
Sum Root to Leaf Numbers	129
Count Paths Sum = k	437

🔹 4. Tree Views (Left, Right, Top, Bottom)

➡️ BFS or DFS with column/depth tracking

Problem Type	Leetcode
Right View	199
Left View	GFG
Top View	GFG
Bottom View	GFG
Vertical Order Traversal	987

🔹 5. Diameter, Height, Balanced Tree

➡️ Use postorder for height-based logic

Problem Type	Leetcode
Diameter of Binary Tree	543
Height / Depth	104, 111
Is Balanced Binary Tree	110
Min Depth of Binary Tree	111

🔹 6. Tree Reconstruction (Build Tree)

➡️ Build from inorder + preorder/postorder

Problem Type	Leetcode
Construct Binary Tree from Pre + Inorder	105
Construct from In + Postorder	106

Problem Type	Leetcode
Serialize + Deserialize Binary Tree	🔸 297

🔹 **7. Mirror, Symmetry, Invert Tree**

➡️ Use simple DFS

Problem Type	Leetcode
Invert Binary Tree	🔸 226
Symmetric Tree	🔸 101
Check Mirror Tree	🔸 GFG

🔹 **8. Ancestors, LCA (Binary Tree + BST)**

➡️ Path tracing or recursive LCA

Problem Type	Leetcode
Lowest Common Ancestor in BST	🔸 235
LCA in Binary Tree	🔸 236
All Ancestors of Node	🔸 GFG

🔹 **9. Morris Traversal (O(1) space)**

➡️ Advanced, for interview edge

Problem Type	Leetcode
Morris Inorder Traversal	🔸 Concept
Recover Binary Search Tree	🔸 99

🔹 **10. Tree to Linked List / Flattening**

➡️ Inorder or Preorder flattening

Problem Type	Leetcode
Flatten Binary Tree to Linked List	🔸 114

Problem Type	Leetcode
Convert BST to Greater Tree	🔸 538
Convert BST to DLL	🔸 GFG

🔹 **11. Tree DP (Postorder DP on Tree)**

➡️ DP on children → parent

Problem Type	Leetcode
House Robber III	🔸 337
Longest Zigzag Path in Tree	🔸 1372
Maximum Depth of N-ary Tree	🔸 559
Diameter of N-ary Tree	🔸 GFG
Distribute Coins in Binary Tree	🔸 979

🔹 **12. Binary Tree Coloring / Game Theory**

➡️ Strategy + Tree traversal

Problem Type	Leetcode
Binary Tree Coloring Game	🔸 1145
Tree Coloring DP (advanced)	🔸 GFG

🧠 **Summary Table**

Pattern	Key Questions
Traversals	94, 144, 145, 102
BST	98, 235, 700
Path Sums	112, 124, 437
Views	199, 987
Diameter/Height	543, 104, 110
Reconstruction	105, 106, 297

Pattern	Key Questions
Mirror/Inversion	101, 226
LCA	235, 236
Morris Traversal	99
Flattening	114, 538
Tree DP	337, 979
Game Trees	1145

✅ **Practice Strategy**

1. Start with **basic DFS/BFS and recursion**
2. Master **BST-specific problems**
3. Practice **LCA, path sums, diameter, height**
4. Learn **views, reconstruction, flattening**
5. Try **Tree DP** and **advanced Morris / Serialization**

📌 **Pro Tip**:
Tree problems me mostly postorder DFS ya level order BFS ka variation use hota hai.
Dry run karna bahut helpful hota hai, especially in **LCA, path sum, tree DP** type problems.

☑ Graphs – Har Important Variation aur Pattern (Interview Focused)

🔹 1. Graph Traversal (DFS + BFS)

➡ DFS aur BFS sabse basic aur widely used traversal techniques hain.

Problem Type	Leetcode
DFS Traversal	🔸 200 – Number of Islands
BFS Traversal	🔸 200 – Number of Islands
Path finding using BFS	🔸 1091 – Shortest Path in Binary Matrix
Detect Cycle in Graph	🔸 GFG – Cycle Detection (DFS/BFS)
Connected Components	🔸 694 – Number of Distinct Islands

🔹 2. Shortest Path Algorithms

➡ Dijkstra's, Bellman-Ford, Floyd-Warshall — Distances from one node to others.

Problem Type	Leetcode
Dijkstra's Shortest Path	🔸 743 – Network Delay Time
Bellman-Ford Algorithm	🔸 787 – Cheapest Flights Within K Stops
Floyd-Warshall	🔸 GFG – Shortest Path All Pairs

🔹 3. Cycle Detection (Directed + Undirected)

➡ DFS-based cycle detection (directed/undirected)

Problem Type	Leetcode
Cycle Detection in Directed Graph	🔸 267 – Palindrome Permutation II (Directed)
Cycle Detection in Undirected Graph	🔸 GFG
Detect Cycle in Directed Graph (Topological Sort)	🔸 207 – Course Schedule II

🔹 4. Topological Sort (DAG)

➡ Directed Acyclic Graph (DAG) me ordering of vertices

Problem Type	Leetcode
Topological Sort	🔸 207 – Course Schedule
Alien Dictionary	🔸 269
Task Scheduling (Kahn's Algorithm)	🔸 621 – Task Scheduler

🔹 5. Graph Coloring (Bipartite Check)

➡️ Check whether a graph is bipartite (2-colorable)

Problem Type	Leetcode
Bipartite Graph	🔸 785 – Is Graph Bipartite?
Graph Coloring	🔸 GFG – Graph Coloring
Bipartite Matching	🔸 GFG

🔹 6. Minimum Spanning Tree (MST)

➡️ Prim's and Kruskal's Algorithm for finding minimum spanning tree.

Problem Type	Leetcode
Kruskal's Algorithm	🔸 GFG – MST using Kruskal's Algorithm
Prim's Algorithm	🔸 GFG – MST using Prim's Algorithm
Minimum Cost to Connect All Points	🔸 1584

🔹 7. Union-Find / Disjoint Set Union (DSU)

➡️ Efficiently handle dynamic connectivity in a graph (using path compression & union by rank).

Problem Type	Leetcode
Number of Connected Components	🔸 323
Redundant Connection	🔸 684
Kruskal's MST (Union-Find)	🔸 GFG
Account Merge	🔸 721
Connected Components in a Graph	🔸 323

🔹 8. Graph Search (BFS/DFS on Weighted Graphs)

➡️ BFS/DFS with additional edge weights or constraints

Problem Type	Leetcode
Shortest Path in Weighted Graph	🔸 743
Cheapest Flights within K Stops	🔸 787
Path with Maximum Probability	🔸 1514

🔹 9. Finding Strongly Connected Components (SCC)

➡️ Kosaraju's or Tarjan's Algorithm for finding SCCs in a directed graph.

Problem Type	Leetcode
Strongly Connected Components (SCC)	🔸 GFG
Kosaraju's Algorithm	🔸 GFG
Tarjan's Algorithm	🔸 GFG

🔹 10. Flood Fill / Connected Region Search

➡️ Similar to DFS/BFS, but with an area to fill.

Problem Type	Leetcode
Flood Fill	🔸 733 – Flood Fill
Surrounded Regions	🔸 130 – Surrounded Regions
Number of Enclaves	🔸 1020

🔹 11. Articulation Points & Bridges

➡️ Identify critical nodes/edges whose removal increases graph's connected components.

Problem Type	Leetcode
Articulation Points	🔸 GFG
Bridges in a Graph	🔸 GFG

🔹 12. Traveling Salesman Problem (TSP)

➡️ Solve the problem using Brute Force / Dynamic Programming / Approximation.

Problem Type	Leetcode
Traveling Salesman	🔸 329 – TSP Approximation

🧠 Summary Table

Pattern	Key Problems
DFS/BFS	200, 1091
Shortest Path	743, 787
Cycle Detection	GFG (DFS)
Topological Sort	207, 269
Graph Coloring	785
MST (Prim/Kruskal)	GFG
Union-Find (DSU)	684, 721
Weighted Graph Search	743, 787
SCC	GFG, Kosaraju's
Flood Fill	733, 130
Articulation Points	GFG
TSP	329

✅ Practice Strategy

1. Master DFS/BFS traversal.
2. Learn Topological Sort and Cycle Detection techniques.
3. Practice Minimum Spanning Tree (Prim/Kruskal) using Union-Find.
4. Get comfortable with Bipartite Graph, MST, and Shortest Path algorithms.
5. Practice problems on Graph Coloring, Flood Fill, SCC, and Articulation Points.

📌 **Pro Tip:**
Graph problems me sabse zyada DFS/BFS base solution hota hai. Union-Find bhi bahut useful hai for connected components and MST problems.
Kosaraju and Tarjan's algorithms for SCC are often asked in advanced problems.