Meritshot
EDUCATION

How to Recognize
DSA Patterns
in Interview Questions

Who Dream To Switch

Meritshot
EDUCATION

Sliding Window Pattern

Use this when:

- The problem asks about subarrays or substrings — meaning contiguous elements.
- You need to find something within a continuous range, like:
  - maximum/minimum sum,
  - longest substring,
  - number of distinct elements in a window, etc.

How to recognize it in a question

Look for words like:

“contiguous”, “subarray”, “substring”, “window of size k”, “consecutive”, “longest/shortest sequence”

Example

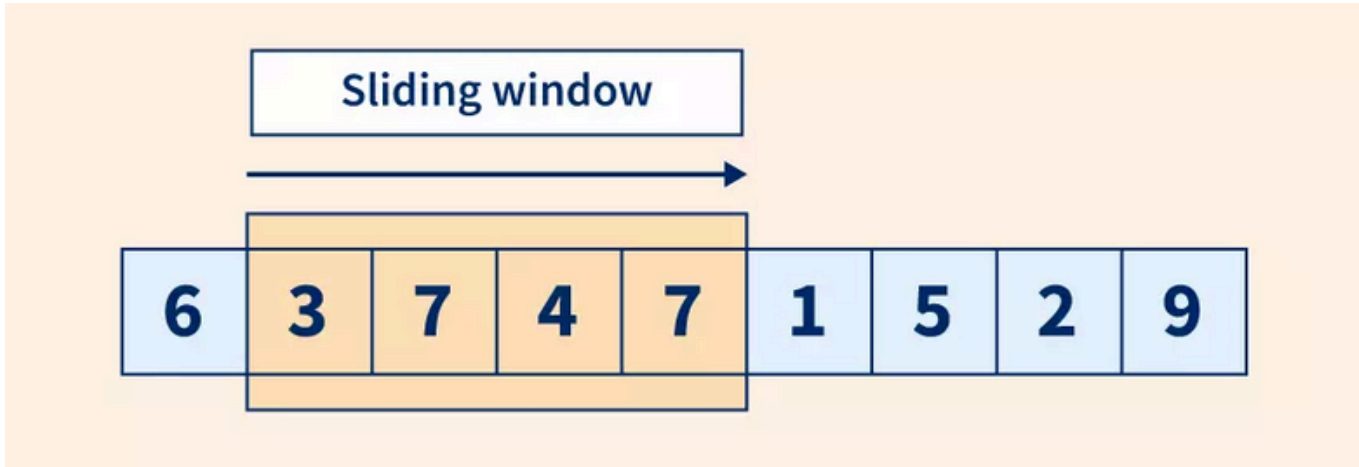
“Find the maximum sum of any subarray of size 3.”
Here you’re sliding a window of size 3 across the array — that’s exactly what this pattern does.

How it works

1. Start with a window (start & end index)
2. Expand the window by adding the next element
3. Shrink it from the start when it exceeds size or condition
4. Keep track of result (like max sum)

Real examples

- Longest substring without repeating characters
- Maximum average subarray
- Minimum size subarray sum



Meritshot
EDUCATION

2/ Two Pointers Pattern

When to use it

- You have sorted data (array or string)
- You're looking for pairs or triplets that meet a condition (sum, difference, etc.)
- You want to compare or move from both ends

How to recognize

Look for words like:
"sorted array", "pair with sum", "remove duplicates", "reverse", "palindrome"

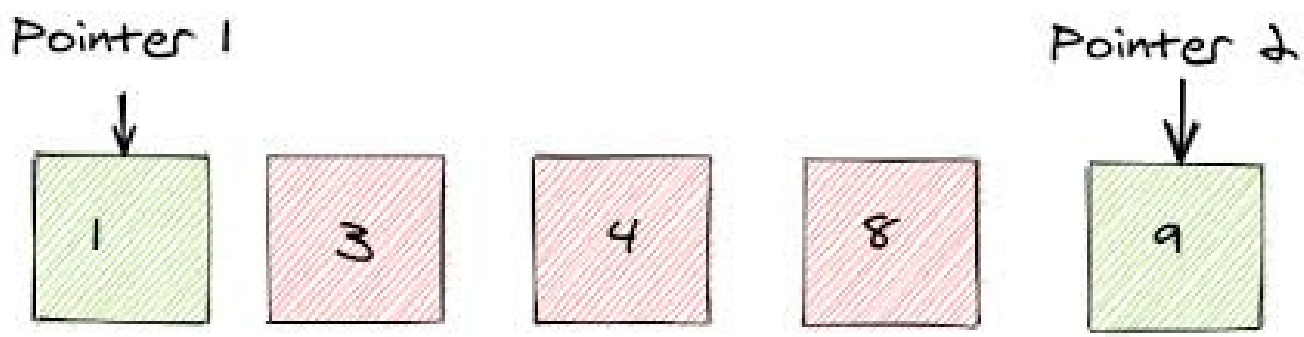
Example
"Given a sorted array, find two numbers that add up to 10."
You start one pointer at the beginning, another at the end — then move them inward.

How it works

- Place left at start and right at end.
- If sum < target → move left++
- If sum > target → move right--
- If equal → found it!

Real examples

- Two sum in sorted array
- Remove duplicates from a sorted array
- Container with the most water
- Valid palindrome check



3/ Fast & Slow Pointers Pattern (Tortoise & Hare)

Use when:

- The problem involves loops or cycles (especially in linked lists)
- You need to find the middle or the cycle starting point

How to recognize

Look for words like:
“cycle”, “loop”, “find middle”, “repeated number”, “linked list”

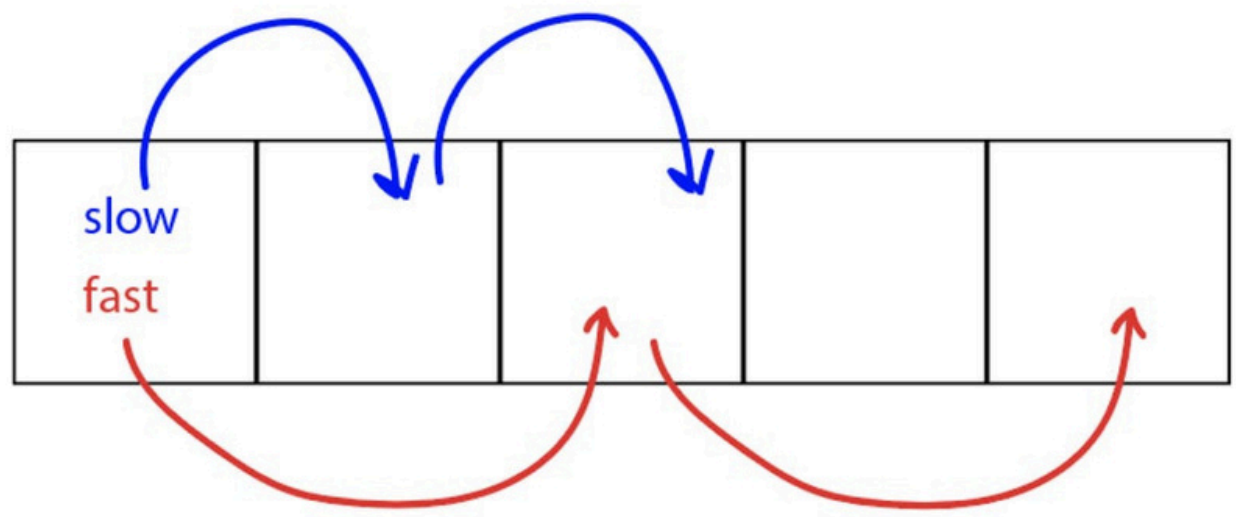
Example
“Detect if a linked list has a cycle.”
Fast moves 2 steps, slow moves 1. If they ever meet, there’s a cycle.

How it works

1. Initialize slow and fast at head.
2. Move slow = slow.next, fast = fast.next.next
3. If fast == slow → cycle found.

Real examples

- Detect cycle in linked list
- Find middle of linked list
- Find starting point of cycle
- Happy number problem



Meritshot
EDUCATION

4/ Merge Intervals Pattern

Use when:

- You’re given intervals or ranges (like meeting times, bookings, ranges of data)
- You must merge overlapping ones or insert a new one.

How to recognize
Look for words like:
“start time”, “end time”, “overlap”, “merge”, “interval”, “schedule”

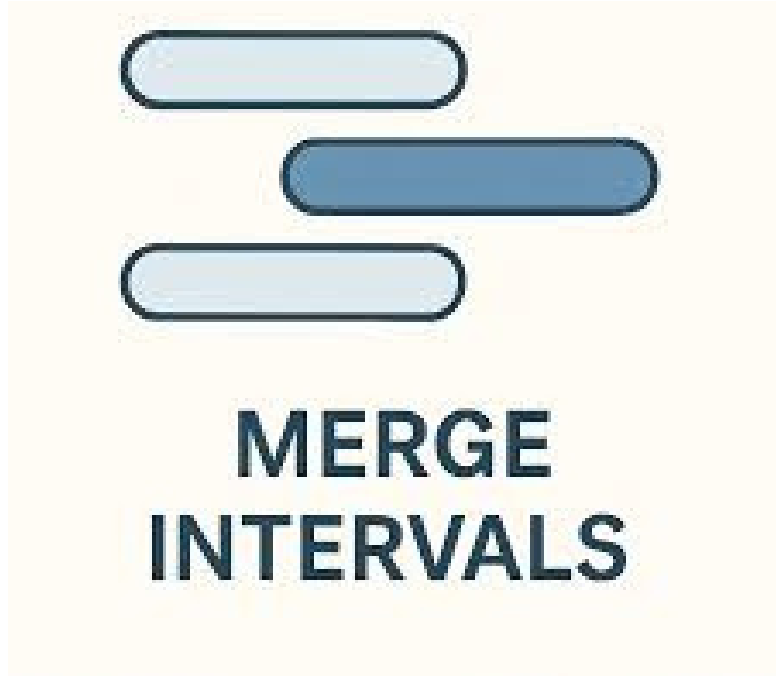
Example
“Given meeting times, find how many meeting rooms are needed.”

How it works

1. Sort intervals by start time
2. Compare each interval with the previous one
3. If they overlap, merge them by updating end time
4. Otherwise, start a new merged interval

Real examples

- Merge overlapping intervals
- Insert a new interval
- Minimum meeting rooms
- Employee free time



5/ Cyclic Sort Pattern

Use when:

- The array contains numbers from 1 to N
- You need to find missing, duplicate, or misplaced numbers
- The array is unsorted but all numbers are in a known range.

How to recognize

Look for words like:
“1 to n”, “find missing number”, “find duplicate”, “first missing positive”

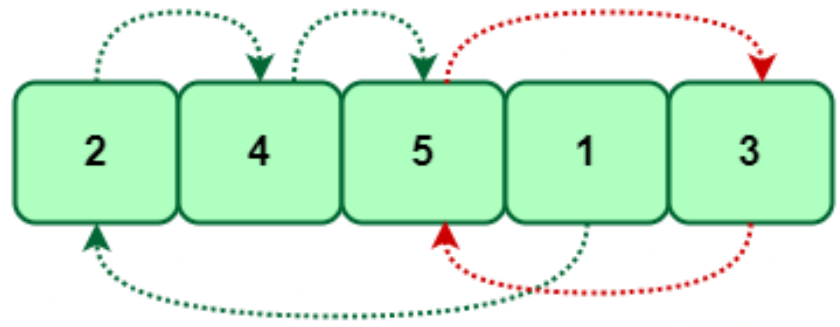
Example
“Find the missing number in the array [3, 0, 1].”

How it works

- For each element, place it in its correct index (num → index = num - 1)
- After sorting, check which index doesn’t match the expected value.

Real examples

- Missing number
- Find all duplicates
- Find the first missing positive integer
- Find a corrupt pair (missing + duplicate)



6/ In-Place Reversal of Linked List Pattern

When to use it

Use when:

- You need to reverse or reorder a linked list (entirely or partially)
- You can’t use extra memory (O(1) space)

How to recognize

Look for:
“reverse linked list”, “reorder”, “rotate”, “check palindrome (linked list)”

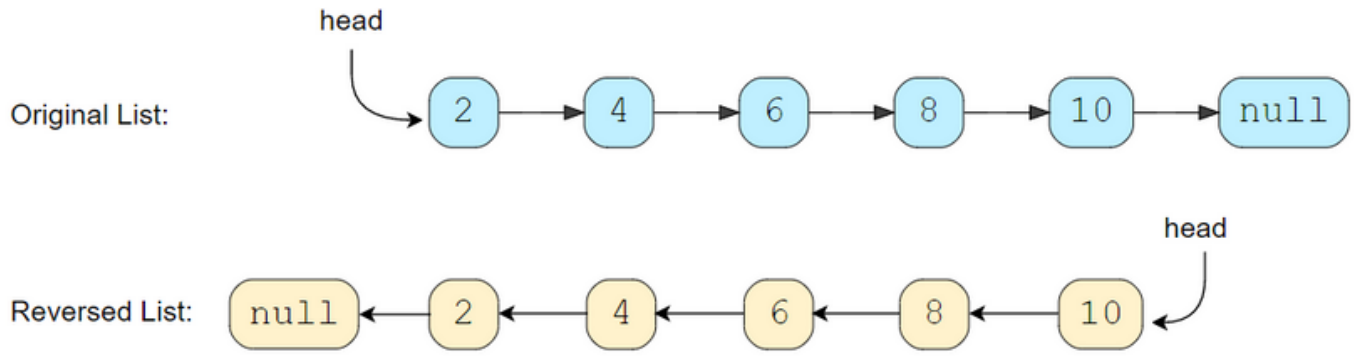
Example
“Reverse a linked list between position m and n.”

How it works

1. Use 3 pointers — prev, current, next
2. Re-point the next of the current node backward
3. Continue until reversed

Real examples

- Reverse the entire linked list
- Reverse nodes in k-groups
- Palindrome linked list
- Rotate the linked list



Meritshot
EDUCATION

Tree / Graph Traversal Pattern (BFS / DFS)

Use when:

- You’re given a tree or graph
- You must visit nodes, find paths, calculate depth, or search

How to recognize

Look for:
“level order”, “depth”, “traverse”, “path sum”, “connected components”, “shortest path”

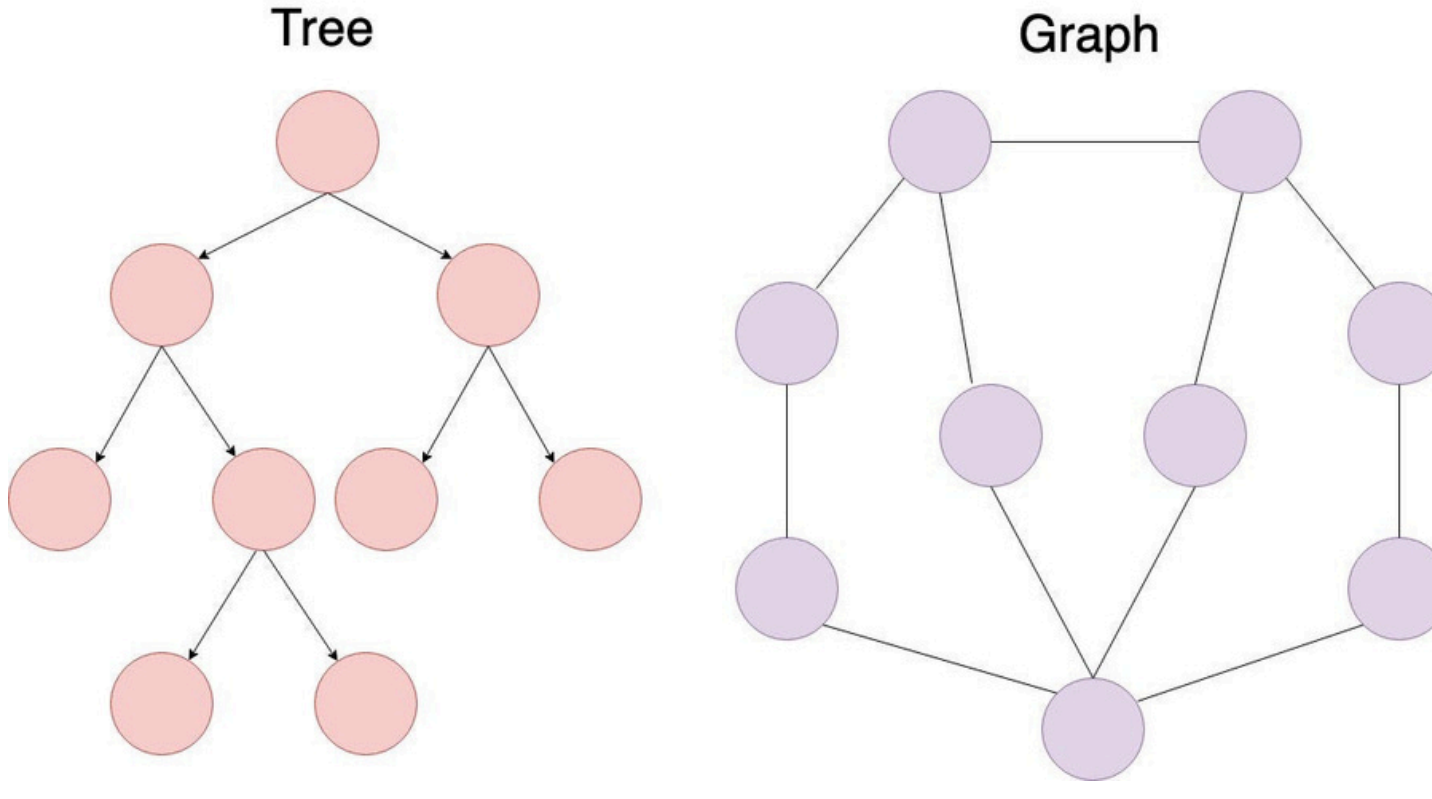
Example
“Print all nodes level by level.”

How it works

- BFS (Breadth First Search) → use a queue, explore neighbors level by level.
- DFS (Depth First Search) → use recursion or stack, go deep before backtracking.

Real examples

- Binary tree level order traversal
- Minimum depth of a binary tree
- Path sum in a tree
- Number of islands (graph)



Quick Recognition Flow

1. Step 1: Is it about subarrays or substrings? → **Sliding Window**
2. Step 2: Is it sorted or about pairs? → **Two Pointers**
3. Step 3: Linked list with cycle or middle? → **Fast-Slow**
4. Step 4: Intervals overlapping? → **Merge Intervals**
5. Step 5: Range 1–N, missing or duplicate? → **Cyclic Sort**
6. Step 6: Linked list reversal? → **In-place Reverse**
7. Step 7: Tree/Graph traversal? → **BFS/DFS**

Quick Recognition Flow

1. Step 1: Is it about subarrays or substrings? → **Sliding Window**
2. Step 2: Is it sorted or about pairs? → **Two Pointers**
3. Step 3: Linked list with cycle or middle? → **Fast-Slow**
4. Step 4: Intervals overlapping? → **Merge Intervals**
5. Step 5: Range 1–N, missing or duplicate? → **Cyclic Sort**
6. Step 6: Linked list reversal? → **In-place Reverse**
7. Step 7: Tree/Graph traversal? → **BFS/DFS**

Meritshot
EDUCATION

Pattern Recognition Mind Map

Problem Mentions	You Think Of	Pattern
“subarray”, “substring”, “window”	moving start-end window	Sliding Window
“sorted array”, “pairs”, “reverse”, “palindrome”	two pointers	Two Pointers
“cycle”, “loop”, “middle”	linked list with two speeds	Fast & Slow Pointers
“start/end time”, “overlap”, “schedule”	sorting + merging	Merge Intervals
“missing”, “1 to n”, “duplicate”	index = number - 1	Cyclic Sort
“reverse list”, “rotate list”	pointer reversal	In-place Reversal
“tree”, “graph”, “path”, “level”	BFS or DFS traversal	Tree/Graph Traversal

Meritshot
EDUCATION

Your one-step destination for your Career Upskilling

Lets make a community of 20k+ learnerrs

meritshoteducation