

Chapter 12

■ Mental Model

The one framework that makes every Stack & Queue problem clickable

■ The Master Mental Model

STACK = "I care about the MOST RECENT thing" QUEUE = "I care about the OLDEST thing" Every Stack/Queue problem is secretly asking: which of these two orders matters? When you need to REVERSE something, MATCH something, or TRACK what just happened → Stack. When you need to PROCESS things in the order they arrived, EXPLORE level by level → Queue.

■ The BRACE Framework for Stack Problems

B **Bracket / Matching** Does this involve matching pairs? → Stack

R **Reversal** Do I need to reverse order at some point? → Stack

A **Undo / History** Does each step depend on the previous? → Stack

C **Calculation (expression)** Is this arithmetic or logic evaluation? → Stack

E **Elimination (monotonic)** Does a new element invalidate old ones? → Monotonic Stack

■ The BFS Wave for Queue Problems

Imagine throwing a stone into a pond. The ripples spread outward in circles. Each circle = one level. BFS using a Queue works EXACTLY like this. Level 0: source node (the stone) Level 1: all neighbors of source Level 2: all neighbors of level 1 (not yet visited) The Queue AUTOMATICALLY handles level-by-level processing because of FIFO. Nodes added at level 1 are processed completely before level 2 nodes.

■ The One Thing To Remember Forever

A Stack is time reversed. Whatever you put in last, you get out first. Every powerful use of a stack exploits this reversal: • Bracket matching: reversing your 'opening' history to match closings • Expression evaluation: reversing operator precedence • DFS: reversing path to backtrack • Undo: reversing your action history Whenever a problem requires you to look back at recent history or reverse a sequence — that's your signal. Reach for the Stack.

■ You're Interview Ready When...

- You can implement Stack and Queue from scratch in under 5 minutes
- You recognize monotonic stack patterns without hints
- You can explain amortized $O(1)$ for Queue-via-Two-Stacks confidently
- You know exactly which Java class to use (ArrayDeque, not Stack)
- Valid Parentheses, Min Stack, Next Greater Element feel trivial
- You can dry-run Trapping Rain Water on a whiteboard

"A stack is just an array that forgot it had random access. A queue is just a line that forgot it had cutting privileges." — Remember this, and you'll never forget which is which.