

Chapter 2

■ Core Intuition

Build the mental model — deeply, not just definitions

■ Stack — The Pancake Model

Picture a stack of pancakes. You always add a new pancake on TOP. You always eat from the TOP. The bottom pancake was made first but eaten last. KEY INSIGHT: In a stack, the most recently added item is always the most accessible. This reversal property is what makes stacks powerful for problems involving 'tracking history' or 'matching pairs'.

Three Core Operations:

Operation	Description	Time
<code>push(x)</code>	Add x to the top	O(1)
<code>pop()</code>	Remove & return top element	O(1)
<code>peek()</code>	View top element without removing	O(1)

■ Queue — The Ticket Line Model

Picture people lining up for a movie. The first person to arrive stands at the FRONT. New arrivals join the BACK. When tickets open, people leave from the FRONT. KEY INSIGHT: A queue preserves order. Whatever came in first, leaves first. This fairness property makes queues ideal for scheduling, BFS, and any 'process in order' scenario.

Three Core Operations:

Operation	Description	Time
<code>enqueue(x)</code>	Add x to the rear	O(1)
<code>dequeue()</code>	Remove & return front element	O(1)
<code>peek()</code>	View front element without removing	O(1)

■ The Key Difference

Stack

- Access: Only from the TOP
- Insert & Remove: Same end
- Pattern: LIFO (reversal)
- Best for: tracking state, matching

Queue

- Access: Front (remove), Rear (insert)
- Insert & Remove: Opposite ends
- Pattern: FIFO (ordering)
- Best for: level-order, scheduling

■ Quick Test: If the problem says 'most recent', 'last opened', 'undo', or 'matching brackets' → think Stack. If the problem says 'first come first served', 'level by level', 'shortest path' → think Queue.