Pattern 1: Balanced Parentheses / Valid Sequence

Key Idea: Use stack to match opening and closing elements.

Common Problems:

- Valid Parentheses (LC 20)
- Minimum Remove to Make Valid Parentheses (LC 1249)
- Longest Valid Parentheses (LC 32)
- Valid Palindrome with Stack (Custom)

Trick:

- Use stack to push **opening** characters.
- Pop on matching **closing** characters.
- Stack should be **empty** at the end for a valid sequence.

Pattern 2: Monotonic Stack (Next Greater/Smaller)

Key Idea: Use stack to maintain a **monotonic** order for finding **next** or **previous** elements.

Common Problems:

- Next Greater Element I (LC 496)
- Next Greater Element II (LC 503)
- Largest Rectangle in Histogram (LC 84)
- Daily Temperatures (LC 739)

Trick:

- **Increasing** stack for **next greater** elements.
- **Decreasing** stack for **next smaller** elements.
- Process elements **right to left** for circular arrays.

📉 Pattern 3: Stock Span / Price Spanning

Key Idea: Use stack to track spans and maintain indices.

Common Problems:

- Online Stock Span (LC 901)
- Largest Rectangle in Histogram (LC 84)
- Trapping Rain Water (LC 42)

Trick:

- Use a **pair** (value, index) to track spans.
- Pop when current is greater than stack top for span calculation.

🏔️ Pattern 4: Contiguous Subarrays / Sliding Window

Key Idea: Use stack to keep track of valid subarrays or windows.

Common Problems:

- Sum of Subarray Minimums (LC 907)
- Remove K Digits (LC 402)
- Sliding Window Maximum (LC 239)

Trick:

- Use stack for efficient window processing.
- Pair with **deque** for maximum or minimum in subarrays.

📦 Pattern 5: Stack for Backtracking / DFS Simulation

Key Idea: Use stack to simulate recursion for DFS or backtracking.

Common Problems:

- Basic Calculator II (LC 227)
- Decode String (LC 394)
- Flatten Nested List Iterator (LC 341)

Trick:

- Use stack to keep track of **state**.
- Push intermediate results and operators for nested structures.

🗃️ Pattern 6: Stack for Linked Lists / LRU Cache

Key Idea: Use stack for reversing, reversing pairs, or LRU logic.

Common Problems:

- Reverse Nodes in k-Group (LC 25)
- LRU Cache (LC 146)
- Add Two Numbers II (LC 445)

Trick:

- Use stack for reversing parts of linked lists.
- Use doubly linked lists for cache structures.

⚙️ How to Identify a Stack Problem?

- **Nested Structures:** Parentheses, brackets, or deeply nested strings.
- **Backtracking:** Need to go back to a previous state.
- **Next Greater/Smaller:** Finding the next or previous element.
- **Maintaining Order:** When you need to keep elements in a sorted order as you process them.
- **Undo Operations:** Like parsing, calculators, or linked list problems.

📚 Must-Know Stack Tricks

- **Use Indexes:** For problems like **Next Greater Element** to maintain context.
- **Dual Stack / Two-Pass:** For complex structures like calculator problems.

- **Avoid Nesting:** Use a single stack cleverly for multiple conditions.
- **Process from End:** For circular array problems.
