🔥 COMPLETE DSA + CODING INTERVIEW MASTER CHECKLIST

(Pattern-based | Covers 95–100% questions asked in MNCs)

🧱 0. CORE FOUNDATIONS (Must-Know)

You should be able to answer these **without thinking**.

✅ Basics

- Time & Space Complexity
- Big-O / Big-Ω / Big-Θ
- Best / Worst / Average case
- Recursion stack space
- Iterative vs Recursive trade-offs

🧩 1. ARRAYS (MOST IMPORTANT – ~30% of tests)

🔹 Patterns in Arrays

1. **Linear Scan**
2. **Prefix Sum**
3. **Carry Forward**
4. **Two Pointers**
5. **Sliding Window**
6. **Binary Search on Array**
7. **Sorting + Scan**
8. **Subarray Logic**
9. **Kadane's Algorithm**
10. **Difference Array**
11. **Matrix Traversal**

🔹 Subtopics (100%)

- Max / Min / Second Max

- Reverse / Rotate array
- Frequency count
- Missing / Repeating number
- Subarray sum / count
- Max subarray sum
- Longest subarray problems
- Merge intervals
- 2D arrays (matrix)
- Spiral traversal
- Row/column wise operations

✅ **If you master array patterns → half the exam is done**

🔤 **2. STRINGS**

🔹 **String Patterns**

1. **Frequency Map**
2. **Two Pointers**
3. **Sliding Window**
4. **String Matching**
5. **Palindrome Logic**
6. **Hashing on characters**

🔹 **Subtopics**

- Reverse string
- Palindrome check
- Anagram
- Longest substring (with/without repeat)
- String compression
- Remove duplicates
- String rotation

- Pattern matching (basic)
- ASCII manipulation

🧱 3. HASHING (THE SPEED BOOSTER)

🔹 **Patterns**

1. **Frequency Table**
2. **Lookup Optimization**
3. **Set-based elimination**
4. **Prefix sum + Hashing**

🔹 **Subtopics**

- Two Sum
- Majority element
- First non-repeating
- Count distinct
- Subarray with sum = k
- Longest consecutive sequence

💡 If brute force is slow → hashing is the answer

👉 4. TWO POINTERS & SLIDING WINDOW

🔹 **Two Pointer Patterns**

- Opposite ends
- Same direction
- Slow & fast pointers

🔹 **Sliding Window Patterns**

- Fixed window
- Variable window (expand–shrink)

🔹 **Questions Covered**

- Pair sum
- Remove duplicates
- Container with most water
- Longest substring problems
- Window max/min
- Min window substring

🔁 5. RECURSION & BACKTRACKING

🔹 Recursion Patterns

1. **Base case + recursive case**
2. **Reduce problem size**
3. **Divide & conquer**

🔹 Backtracking Patterns

1. **Pick / Not Pick**
2. **Explore → Undo**
3. **Path-based recursion**

🔹 Subtopics

- Factorial, Fibonacci
- Reverse array/string
- Subsets
- Permutations
- Combination sum
- N-Queens
- Sudoku solver

🔗 6. LINKED LIST

🔹 Patterns

1. **Pointer manipulation**
2. **Fast & slow pointer**
3. **Dummy node trick**

🔹 **Subtopics**

- Reverse LL
- Middle of LL
- Detect cycle
- Remove nth node
- Merge two lists
- Intersection point

📚 **7. STACK & QUEUE**

🔹 **Stack Patterns**

1. **Monotonic Stack**
2. **Bracket matching**
3. **Previous / Next greater**

🔹 **Queue Patterns**

1. **BFS structure**
2. **Sliding window using deque**

🔹 **Subtopics**

- Valid parentheses
- Next greater element
- Stock span
- Min stack
- Queue using stack
- Sliding window maximum

🔍 8. BINARY SEARCH (VERY HIGH YIELD)

🔹 **Patterns**

1. **Classic binary search**
2. **First / Last occurrence**
3. **Binary search on answer**
4. **Search in rotated array**

🔹 **Subtopics**

- Lower/upper bound
- Peak element
- Square root
- Book allocation
- Aggressive cows
- Min days / max capacity problems

🧠 9. SORTING & DIVIDE & CONQUER

🔹 **Patterns**

1. **Merge sort logic**
2. **Quick sort partition**
3. **Sort + scan**

🔹 **Subtopics**

- Merge sort
- Quick sort
- Inversion count
- Sort colors
- Custom sorting

🪙 10. GREEDY (FREQUENT IN MNCs)

🔹 **Patterns**

1. **Local optimum → Global optimum**
2. **Sort by key + select**

🔹 **Subtopics**

- Activity selection
- Job sequencing
- Minimum coins
- Fractional knapsack
- Interval scheduling

🧩 11. BIT MANIPULATION (SHORT & SCORING)

🔹 **Patterns**

1. **XOR trick**
2. **Bit masking**
3. **Power of two logic**

🔹 **Subtopics**

- Single number
- Count set bits
- Toggle bits
- Check odd/even
- Subsets using bits

🌳 12. TREES

🔹 **Tree Patterns**

1. **DFS (pre/in/post)**
2. **BFS (level order)**
3. **Bottom-up recursion**

4. **Height-based logic**

🔹 **Subtopics**

- Traversals
- Height / depth
- Diameter
- Balanced tree
- LCA
- Path sum
- Views (left/right/top)

🕸️ **13. GRAPHS**

🔹 **Graph Patterns**

1. **BFS / DFS**
2. **Visited array**
3. **Cycle detection**
4. **Topological sort**

🔹 **Subtopics**

- Number of islands
- Connected components
- Cycle detection (directed/undirected)
- Topo sort
- Shortest path (basic)

📈 **14. DYNAMIC PROGRAMMING (INTERVIEW DECIDER)**

🔹 **DP Patterns (MOST IMPORTANT)**

1. **1D DP**
2. **2D DP**

3. **Pick / Not Pick**
4. **Grid DP**
5. **DP on subsequences**
6. **DP on partitions**

🔹 **Subtopics**

- Fibonacci
- Climbing stairs
- House robber
- Knapsack (0/1)
- LCS
- LIS
- Matrix DP
- Subset sum

🧪 **15. INTERVIEW-SPECIFIC SKILLS**

✅ **Must Practice**

- Dry run explanation
- Edge cases
- Time/space justification
- Clean code
- Optimal vs brute force comparison

🏆 **FINAL TRUTH (VERY IMPORTANT)**

❝ Coding tests are NOT random.
They are **pattern-based**.
If you master these patterns → you clear exams ❞