**📅 15-Day DSA Preparation Roadmap**

**Day 1: Arrays (Basics)**

* **Concepts**: Array operations, traversal, insertion, deletion, searching. Time & space complexity basics.
* **Problems**:
  + Easy: Find max & min in an array, Reverse an array.
  + Medium: Rotate array (by k steps), Move zeroes.
  + Hard: Trapping rainwater, Maximum subarray sum (Kadane’s algorithm).
* **Time Allocation**:
  + Theory → 1 hr
  + Practice → 3 hrs
  + Revision → 30 min
* **Tips**: Always try **brute force first**, then optimize step by step.

**Day 2: Strings (Basics + Hashing)**

* **Concepts**: String manipulation, character frequency, palindrome, anagrams, hash maps.
* **Problems**:
  + Easy: Check palindrome, Count vowels.
  + Medium: Longest substring without repeating characters, Group anagrams.
  + Hard: Minimum window substring, String to integer (atoi).
* **Time Allocation**: 1 hr theory, 3 hrs practice, 30 min revision.
* **Resources**: LeetCode String tag, "Cracking the Coding Interview".

**Day 3: Searching Algorithms**

* **Concepts**: Linear Search, Binary Search, Binary Search on answer.
* **Problems**:
  + Easy: Implement Binary Search.
  + Medium: First & last position in sorted array, Search in rotated sorted array.
  + Hard: Median of two sorted arrays, Aggressive cows (Binary Search on answer).
* **Tips**: Always dry-run binary search on **paper** before coding.

**Day 4: Sorting Algorithms**

* **Concepts**: Bubble, Selection, Insertion, Merge Sort, Quick Sort. Time/space trade-offs.
* **Problems**:
  + Easy: Sort colors (Dutch National Flag).
  + Medium: Merge intervals, Kth largest element.
  + Hard: Count inversions, Minimum difference between subsets.
* **Resources**: Visualgo.net (sorting animations).

**Day 5: Recursion & Backtracking (Basics)**

* **Concepts**: Recursion tree, base & recursive cases, backtracking idea.
* **Problems**:
  + Easy: Factorial, Fibonacci using recursion.
  + Medium: Rat in a maze, N-Queens (small n).
  + Hard: Sudoku solver, Word search.
* **Tips**: Draw recursion trees → helps to debug logic.

**Day 6: Linked List (Singly & Doubly)**

* **Concepts**: Insertion, deletion, reverse linked list, slow & fast pointers.
* **Problems**:
  + Easy: Reverse linked list, Detect cycle.
  + Medium: Merge two sorted lists, Remove nth node from end.
  + Hard: Copy list with random pointer, Flatten linked list.

**Day 7: Stack & Queue**

* **Concepts**: Stack (LIFO), Queue (FIFO), Monotonic Stack, Deque.
* **Problems**:
  + Easy: Valid Parentheses, Implement Queue using Stacks.
  + Medium: Next Greater Element, Sliding Window Maximum.
  + Hard: Largest Rectangle in Histogram, Min Stack with O(1).
* **Strategy**: Think in terms of **"last seen element"** for stack problems.

**Day 8: Revision + Mock Coding Round**

* **Concepts**: Revise Arrays → Strings → Searching → Sorting.
* **Tasks**:
  + Solve 6–8 mixed problems from last 7 days.
  + Revise key formulas (Kadane, Binary Search, Merge Sort steps).
* **Time Allocation**:
  + Revision → 2 hrs
  + Practice → 2 hrs

**Day 9: Trees (Basics)**

* **Concepts**: Binary Tree, Traversals (Inorder, Preorder, Postorder, Level Order).
* **Problems**:
  + Easy: Maximum depth of Binary Tree, Inorder traversal.
  + Medium: Symmetric tree, Diameter of Binary Tree.
  + Hard: Construct tree from inorder & preorder, Serialize/deserialize tree.
* **Tips**: Learn recursion for traversals → iterative is just optimization later.

**Day 10: Binary Search Trees (BST)**

* **Concepts**: Properties of BST, Insertion, Deletion, Search.
* **Problems**:
  + Easy: Search in BST.
  + Medium: Validate BST, Lowest Common Ancestor.
  + Hard: Convert sorted array to BST, Recover BST.

**Day 11: Heap & Priority Queue**

* **Concepts**: Heap structure, min/max heap, priority queue applications.
* **Problems**:
  + Easy: Kth largest element.
  + Medium: Merge k sorted lists, Top k frequent elements.
  + Hard: Median in data stream.
* **Resources**: Heaps are key in many **Greedy + Graph** problems.

**Day 12: Graphs (Basics + BFS/DFS)**

* **Concepts**: Graph representation (adj list, adj matrix), BFS, DFS.
* **Problems**:
  + Easy: Number of islands, BFS traversal.
  + Medium: Detect cycle in graph, Rotten Oranges.
  + Hard: Word Ladder, Course Schedule.
* **Tips**: For BFS → use **queue**, for DFS → recursion/stack.

**Day 13: Graphs (Advanced)**

* **Concepts**: Dijkstra, Topological Sort, Union-Find, MST (Kruskal/Prim).
* **Problems**:
  + Medium: Topological sort, Number of connected components.
  + Hard: Minimum spanning tree, Network delay time.

**Day 14: Dynamic Programming (Basics)**

* **Concepts**: Memoization, Tabulation, Common patterns (Fibonacci, Knapsack).
* **Problems**:
  + Easy: Climbing Stairs.
  + Medium: House Robber, Coin Change.
  + Hard: Longest Increasing Subsequence, Edit Distance.
* **Tips**: Break DP into → **choice + recurrence + base case**.

**Day 15: Dynamic Programming (Advanced) + Final Mock**

* **Concepts**: DP on strings, grids, subsequences.
* **Problems**:
  + Medium: Longest Palindromic Subsequence, Unique Paths II.
  + Hard: Matrix Chain Multiplication, Wildcard Matching.
* **Tasks**: Attempt a **mock test** (mix of arrays → DP → graphs).
* **Strategy**: Focus on patterns, not memorization.

**⏱ Daily Schedule Suggestion**

* **Theory** → 1 hr
* **Coding Practice** → 3 hrs
* **Revision** → 30 min
* **Mock/Notes** → 30 min

**📚 Resources**

* **LeetCode** → Topic-wise problems.
* **GeeksforGeeks** → Quick concept notes.
* **NeetCode 150** (YouTube + LeetCode list) → Focused prep.
* **VisualAlgo.net** → Visualize algorithms.