**Phase 1: Foundational Data Structures and Simple Algorithms**

1. **Arrays & Hashing**
   * *Problems*:
     + 1. Two Sum
     + 217. Contains Duplicate
     + 53. Maximum Subarray
     + 121. Best Time to Buy and Sell Stock
   * **Goal**: Learn basic array operations, hash maps for efficient lookups, and sliding window patterns.
2. **Strings**
   * *Problems*:
     + 242. Valid Anagram
     + 49. Group Anagrams
     + 125. Valid Palindrome
     + 5. Longest Palindromic Substring
   * **Goal**: Develop string manipulation skills, learn common patterns like two-pointer technique, and work with hash maps.
3. **Linked Lists**
   * *Problems*:
     + 206. Reverse Linked List
     + 21. Merge Two Sorted Lists
     + 141. Linked List Cycle
     + 19. Remove Nth Node From End of List
   * **Goal**: Practice list traversal, pointer manipulation, and cycle detection with the two-pointer (Floyd’s Tortoise and Hare) approach.

**Phase 2: Intermediate Data Structures and Algorithms**

1. **Stacks & Queues**
   * *Problems*:
     + 20. Valid Parentheses
     + 155. Min Stack
     + 739. Daily Temperatures
     + 225. Implement Stack using Queues
   * **Goal**: Get comfortable with the stack and queue operations and understand their use in expression evaluation and order-based problems.
2. **Binary Trees & Binary Search Trees (BSTs)**
   * *Problems*:
     + 104. Maximum Depth of Binary Tree
     + 98. Validate Binary Search Tree
     + 144. Binary Tree Preorder Traversal
     + 102. Binary Tree Level Order Traversal
   * **Goal**: Practice depth-first and breadth-first traversal, binary tree properties, and binary search trees for ordered data.
3. **Recursion & Backtracking**
   * *Problems*:
     + 46. Permutations
     + 77. Combinations
     + 79. Word Search
     + 78. Subsets
   * **Goal**: Understand recursion fundamentals, solve combinatorial problems, and practice backtracking for constraint-based generation.
4. **Binary Search**
   * *Problems*:
     + 704. Binary Search
     + 33. Search in Rotated Sorted Array
     + 153. Find Minimum in Rotated Sorted Array
     + 875. Koko Eating Bananas
   * **Goal**: Master binary search on sorted data, binary search in unknown order, and adapt the technique to meet conditions.

**Phase 3: Advanced Data Structures and Algorithms**

1. **Heaps/Priority Queues**
   * *Problems*:
     + 703. Kth Largest Element in a Stream
     + 215. Kth Largest Element in an Array
     + 347. Top K Frequent Elements
   * **Goal**: Use heaps to find top k elements efficiently, prioritize tasks, and optimize ordering.
2. **Graphs (DFS, BFS)**
   * *Problems*:
     + 200. Number of Islands
     + 695. Max Area of Island
     + 994. Rotting Oranges
     + 133. Clone Graph
   * **Goal**: Master depth-first and breadth-first search for grid problems, adjacency lists, and traversal of connected components.
3. **Dynamic Programming (1D & 2D)**
   * *Problems*:
     + 198. House Robber
     + 62. Unique Paths
     + 300. Longest Increasing Subsequence
     + 322. Coin Change
   * **Goal**: Start with easier DP problems, learning to break problems into subproblems and use memoization and tabulation.

**Phase 4: Advanced Algorithms and Patterns**

1. **Union-Find / Disjoint Set Union (DSU)**
   * *Problems*:
     + 547. Number of Provinces
     + 684. Redundant Connection
   * **Goal**: Solve problems related to connectivity and components using DSU, especially useful in graph problems.
2. **Trie (Prefix Tree)**
   * *Problems*:
     + 208. Implement Trie (Prefix Tree)
     + 211. Design Add and Search Words Data Structure
   * **Goal**: Understand efficient prefix-based storage for strings, common in autocomplete and word search applications.
3. **Sliding Window & Two Pointer Techniques**
   * *Problems*:
     + 3. Longest Substring Without Repeating Characters
     + 76. Minimum Window Substring
   * **Goal**: Master these techniques to solve problems involving contiguous subarrays or strings, often optimizing space or time complexity.