DSA

▼ QUEUE

- [x] ~~Priority Queue~~
- [ ] Applications of Queue
- [x] ~~Types of Queue~~
- [ ] Circular Queue Implementation
- [ ] Applications of Circular Queue
- [x] ~~Applications of Priority Queue~~
- [x] ~~Double-Ended Queue (Deque)~~
- [x] ~~Applications of Double-Ended Queue~~
- [ ] Bounded Queues
- [x] ~~Queue Implementation~~
- [x] ~~Implement Queue Using Stack~~
- [x] ~~Convert Stack into Queue~~
- [ ] Reverse a Queue
- [ ] Circular Queue Using Linked List
- [x] ~~Implement Double-Ended Queue Using Linked List~~
- [ ] Circular Buffers
- [ ] Monotonic Queue

▼ STACK

- [x] ~~Purpose of Stack Pointer~~
- [ ] Reverse a Stack Using Recursion
- [ ] Sort a Stack
- [ ] Stack That Rejects Duplicate Values

- [ ] Delete Specific Node from Stack
- [ ] Complexity of Push Element into Stack
- [x] ~~Monotonic Stack~~
- [x] ~~Min Stack~~
- [ ] Implement a Stack with Methods to Push, Pop, and Get Current Highest Number in O(1) Complexity
- [ ] Palindrome Using Stack
- [x] ~~Call Stack~~
- [ ] Applications of Stack
- [ ] Stack Implementation
- [ ] Stack Using Linked List
- [ ] Stack Overflow vs Underflow
- [ ] Reverse a String Using Stack
- [ ] Parenthesis Checking Using Stack
- [ ] How Stack is Used in Undo-Redo Operations
- [ ] Implement Stack Using Queue

SORTING

- [x] ~~Time Complexity of Sorting Algorithms~~
- [x] ~~Space Complexity of Quick Sort and Merge Sort~~
- [x] ~~Worst Case Complexity of Quick Sort~~
- [x] ~~Average Case Complexity of Quick Sort~~
- [x] ~~Space Complexity of Merge Sort~~
- [ ] Check if Array is Sorted with Linear Time Complexity
- [x] ~~Why Complexity of Merge Sort is O(n log n)~~
- [x] ~~Disadvantages of Merge Sort~~
- [x] ~~Use of Different Types of Sorting Algorithms~~

- [x] ~~Divide and Conquer~~
- [ ] Why Bubble Sort is a Stable Sorting Algorithm
- [x] ~~Why Merge Sort is Preferred for Linked Lists~~
- [ ] Stable Sorting
- [ ] In-Place Sorting
- [ ] Disadvantages of Quick Sort Over Merge Sort
- [ ] Advantages of Merge Sort Over Quick Sort
- [ ] Merge Sort vs Quick Sort
- [ ] Pivot Selection in Quick Sort
- [ ] Importance of Pivot Value in Quick Sort
- [ ] Does Pivot Affect Performance?
- [ ] Choosing Appropriate Sorting Algorithm
- [ ] Omega and Theta Notation for Sorting Complexities
- [ ] Best Sorting Algorithm for Partially Sorted Small Arrays
- [x] ~~Merge Two Sorted Arrays into a Single Sorted Array in O(n) Time~~
- [ ] Sort Array of Students Based on Age
- [x] ~~Sort an Array of Objects Based on a Property (e.g., .amount)~~
- [ ] Perform Merge Sort on Array of Strings
- [ ] Sort a String Using Merge Sort
- [ ] Merge Sort Implementation
- [ ] Quick Sort Implementation
- [ ] Insertion Sort
- [ ] Selection Sort
- [ ] Bubble Sort
- [ ] Heap Sort

▼ HASH TABLE

- [x] ~~Applications of Hash Table~~
- [x] ~~Types of Hash Functions~~
- [x] ~~Hashing vs Encryption~~
- [ ] Rehashing
- [ ] Hash Table vs Hash Set
- [ ] How to Handle Collisions in a Linked List
- [ ] Methods to Resolve Hash Collisions
- [x] ~~Why Hash Table is Used in Database Indexing~~
- [x] ~~Hash Table Time Complexity~~
- [ ] Collision Handling in Hash Table
- [ ] Open Addressing
- [ ] Linear Probing vs Quadratic Probing
- [ ] Double Hashing
- [ ] Load Factor
- [ ] Separate Chaining
- [ ] Quadratic Probing Practical
- [ ] Hash Table to Check if String Contains Duplicates
- [ ] Hash Table to Find Two Numbers in an Array That Add Up to a Target Sum
- [ ] Find the First Non-Repeating Character Using Hash Table
- [ ] Find the Occurrence of Each Character in a String Using Hash Table
- [ ] Remove Duplicates from an Array Using Hash Table in O(n)
- [ ] Find the Least Occurred Number Using Hash Table
- [ ] Find Uncommon Elements from Two Different Arrays Using Hash Table
- [ ] Implement a Hash Table to Count Frequency of Characters in a String

- [ ] Valid Anagram Using Hash Map
- [ ] Chaining with Linked List

▼ TREE

🛠️ Practical Questions

- [x] ~~Implement Binary Tree (not BST)~~
- [x] ~~Level Order Traversal~~
- [x] ~~Postorder Traversal~~
- [x] ~~Preorder Traversal~~
- [x] ~~Printing All Leaf Nodes in a Tree~~
- [x] ~~Checking Subtree~~
- [x] ~~DFS and BFS in Tree~~
- [x] ~~Write a function that counts all the nodes in a binary tree.~~
- [x] ~~Write a function that counts only the leaf nodes (nodes with no children).~~
- [x] ~~Sum of All Nodes in a Binary Tree~~
- [x] ~~Mirror a Binary Tree, Recursively swap left and right children for each node~~
- [x] ~~Find Maximum Value in a Binary Tree~~
- [x] ~~Write a function that returns the minimum depth from root to the nearest leaf~~
- [x] ~~Check if a Tree is Balanced~~
- [ ] Find Lowest Common Ancestor (LCA)
- [ ] Check if Two Trees are Identical
    - [x] ~~Bonus: Trace the recursive calls for a 3-level tree like:~~
- [x] ~~Write a function to count total number of nodes in a binary tree~~
    - ➤ Use recursion: total = 1 + count(left) + count(right)
- [x] ~~Extend the function to count only leaf nodes~~
    - ➤ Leaf node = node with no left and right children

- [ ] Write a function to check if a binary tree is a valid BST
    - ➤ Rule: left < root < right for every subtree
    - ➤ Use a helper function with min and max range for validation
    - [ ] Provide a violating example:

```
   10
  /  \
 5    15
     /
    6  ← violates BST (6 < 10 but on right side)
```

- [ ] Write a function to search for a value in a binary search tree
    - ➤ Use recursion or iteration
    - [ ] Trace the search for 60 in the tree from Question 1
        - ➤ Start at 50 → go right to 70 → go left to 60 → found ✅

📘 Theory Questions

- [ ] Complete Binary Tree
- [ ] Perfect Binary Tree
- [ ] Full Binary Tree
- [ ] Degenerate Tree
- [ ] Balanced vs Unbalanced Tree
- [ ] Height of Tree
- [ ] Depth of a Node
- [ ] Internal Nodes
- [ ] Siblings

- [ ] Degree of Node
- [ ] Degree of Tree
- [ ] Tree vs Graph
- [ ] Binary Tree
- [ ] Terminologies in Tree
- [ ] Applications of Tree
- [ ] Time Complexity of Search in Binary Tree

▼ BST

📘 Theory Topics (BST)

- [ ] BST vs Binary Tree
- [ ] Applications of BST
- [ ] Complexity of BST Insertion
- [ ] Complexity of Removing Second Largest Element in BST
- [ ] BST Time Complexity for Search, Insert, and Other Operations
- [ ] Allow Duplicate Elements in BST

🛠️ Practical Topics (BST)

- [x] ~~Implement BST~~
- [x] ~~Deletion in BST~~
- [x] ~~Level Order Traversal in BST~~
- [x] ~~Find Height of BST~~
- [x] ~~Count Single Child Nodes in BST~~
- [x] ~~Find Min in BST using Recursion~~
- [ ] Validate BST
- [ ] Find Kth Smallest Element in BST

- [ ] Find Second Largest in BST
- [ ] Find Third Largest in BST
- [ ] Find Element Closest to Target in BST
- [ ] Check if BST is Balanced

▼ HEAP

- [ ] Heap Concept
- [ ] Min Heap
- [ ] Max Heap
- [ ] Heapify (Up and Down)
- [ ] Applications of Heap
- [ ] Priority Queue and Heap
- [ ] Heapify Complexity
- [ ] Complexity of Heap Sort
- [ ] Limitation of Heap
- [ ] Conversion of Min Heap to Max Heap
- [ ] Find Right Child of a Heap

🛠️ Practical Questions (Implementations & Problem Solving):

- [x] ~~Implement Heap~~
- [x] ~~Heap Sort~~
- [x] ~~Delete Node from Heap~~
- [ ] Find Kth Largest in Array using Heap
- [ ] Top K Frequent Elements using Heap

TRIE

Theory Topics

- [ ] Trie Concept
- [ ] Suffix Trie vs Prefix Trie
- [ ] Advantages of Trie
- [ ] Applications of Trie
- [ ] Types of Trie
- [ ] Trie Serialization and Deserialization

🛠️ Practical Topics

- [x] ~~Implement Trie~~
- [x] ~~Insert New Word to Trie~~
- [x] ~~Search Word in Trie~~
- [x] ~~Prefix Search in Trie~~
- [x] ~~Auto Completion using Trie / Word Suggestion using Trie~~
- [x] ~~Longest Prefix in a Trie~~
- [ ] Compressed Trie
- [ ] Suffix Trie
- [ ] Count Words with Given Prefix
- [x] ~~Longest Common Prefix (LeetCode #14)~~
- [ ] Prefix and Suffix Search (LeetCode #745)

GRAPH

- [x] ~~Adjacency Matrix and List~~
- [x] ~~Find Shortest Distance Between Two Vertices~~
- [x] ~~Depth-First Search (DFS)~~

- [x] ~~Graph Traversal Methods~~
- [ ] Represent a Graph in Memory
- [ ] Prims Algorithm
- [ ] Kruskal Algorithm
- [ ] Clone Graph
- [ ] Bipartite Graph
- [ ] Shortest Path in Weighted Graph (Dijkstra's Algorithm)
- [ ] Check Cycles in a Graph
- [ ] How to Count Cycles in a Graph

- [ ] Graph Concept
- [x] ~~Directed Graph vs Undirected Graph~~
- [x] ~~Weighted Graph vs Unweighted Graph~~
- [x] ~~Connected Graph~~
- [x] ~~Disconnected Graph~~
- [ ] Complete Graph
- [x] ~~Adjacency~~
- [x] ~~Degree of Vertex~~
- [x] ~~Cycle in Graph~~
- [x] ~~Loop in Graph~~
- [ ] Spanning Tree
- [ ] Minimum Spanning Tree
- [x] ~~Shortest Path in Graph~~
- [x] ~~Shortest Path in Unweighted Graph~~
- [x] ~~Applications of Graph~~
- [ ] Applications of Weighted Graph

- [ ] Graph Indexing
- [x] ~~Classification of Graph~~
- [ ] Complexity of BFS
- [ ] Complexity of DFS
- [ ] Backtracking in DFS
- [ ] Graph in Social Media (Mutual Friends)