MASTER DSA WITH JAVASCRIPT

🧰 MODULE 1: JavaScript Essentials for DSA (No Libraries)

> Build the language fundamentals needed to implement DSA efficiently.

1. JavaScript Basics

- Primitive vs Reference Types
- Type Coercion and Type Conversion
- Variable Declaration (`var`, `let`, `const`)
- Comparison Operators (`==` vs `===`)
- Truthy and Falsy Values
- Control Flow (if, else, switch)
- Loops (for, while, do-while)

2. Functions

- Function Declarations vs Expressions
- Arrow Functions
- Parameters vs Arguments
- Return Values
- Call Stack Basics

3. Arrays

- Array Creation
- Basic Operations: `push`, `pop`, `shift`, `unshift`, `splice`, `slice`
- Iteration: `for`, `for...of`, `forEach`

- Multi-dimensional Arrays

4. Objects

- Key-Value Storage
- Nested Objects
- Accessing/Deleting Keys
- Looping with `for...in`

5. Strings

- Indexing
- String Methods (`charAt`, `slice`, `substr`, `substring`, `split`, `join`)
- Immutability

🔁 Practice

- Reverse a string
- Count vowels
- Custom implementation of `map()`, `filter()`, and `reduce()`

⏱️ MODULE 2: Time and Space Complexity (Big-O)

> Learn how to measure performance of algorithms.

1. Big-O Notation

- O(1), O(n), O(n²), O(log n), O(2^n)
- Time vs Space
- Best, Worst, and Average Case

2. Analyzing Loops and Recursion

- Nested Loops
- Multiple Input Scenarios
- Recurrence Relations (for recursive algorithms)

🔁 Practice

- Manually calculate Big-O for functions
- Compare performance of linear vs binary search

📚 MODULE 3: Core Data Structures (Pure JavaScript)

1. Arrays

- Static vs Dynamic Arrays
- Two-pointer Technique
- Sliding Window Technique

🔁 Practice

- Rotate array
- Maximum sum subarray of size k
- Move all zeros to the end

---

2. Strings

- Character Frequency
- Substrings and Subsequences

🔁 Practice

- Anagram checker
- Longest Palindromic Substring
- Longest Substring without Repeating Characters

---

3. Hash Tables (Using Objects / Maps)

- Hashing Basics
- Collision Handling (Separate Chaining with Arrays)

🔁 Practice

- Two Sum Problem
- Group Anagrams
- Isomorphic Strings

---

4. Stacks (Implemented with Arrays)

- Push, Pop, Peek
- Applications in Expression Evaluation

🔁 Practice

- Valid Parentheses
- Min Stack
- Infix to Postfix conversion

---

5. Queues

- Enqueue, Dequeue, Front
- Circular Queue
- Queue using Stack

🔁 Practice

- Implement queue using 2 stacks
- Sliding window maximum
- Recent calls counter

---

6. Linked Lists

- Singly Linked List
- Doubly Linked List
- Reversing a List
- Detecting Loops (Floyd's Cycle)

🔁 Practice

- Reverse a Linked List

- Remove nth node from end
- Merge Two Sorted Lists

---

7. Trees

- Binary Trees
- Binary Search Trees (BST)
- Tree Traversals (In-order, Pre-order, Post-order, Level-order)

🔁 Practice

- Check if Tree is Balanced
- Lowest Common Ancestor
- Convert Sorted Array to BST

---

8. Tries (Prefix Tree)

- Insert
- Search
- Starts With

🔁 Practice

- Implement Trie
- Word Search (Board)
- Autocomplete System

---

9. Heaps (Binary Heap using Array)

- Min Heap and Max Heap Implementation
- Heapify, Insert, Extract

🔁 Practice

- K Largest Elements
- Median of Data Stream
- Merge K Sorted Arrays

10. Graphs

- Graph Representations (Adjacency List)
- BFS, DFS
- Directed/Undirected, Weighted/Unweighted

🔁 Practice

- Detect Cycle in Directed Graph
- Clone Graph
- Number of Islands

🧱 MODULE 4: Algorithms in JavaScript

1. Searching

- Linear Search
- Binary Search
- Ternary Search

🔁 Practice

- Search in Rotated Array
- First/Last Occurrence
- Peak Element

---

2. Sorting

- Bubble Sort
- Insertion Sort
- Selection Sort
- Merge Sort
- Quick Sort

🔁 Practice

- Sort Colors (Dutch National Flag)
- Find Kth Largest
- Merge Intervals

---

3. Recursion

- Base Case & Recursive Case
- Backtracking

🔁 Practice

- Factorial, Fibonacci (with and without memo)
- N-Queens Problem
- Sudoku Solver

---

4. Dynamic Programming

- Memoization
- Tabulation
- 1D, 2D DP

🔁 Practice

- Climbing Stairs
- Longest Increasing Subsequence
- 0/1 Knapsack
- Edit Distance

---

5. Greedy Algorithms

- Greedy Choice Property
- Activity Selection
- Huffman Coding (Basic)

🔁 Practice

- Jump Game

- Fractional Knapsack
- Gas Station

---

6. Divide & Conquer

- Merge Sort
- Quick Sort
- Binary Search Variants

🔁 Practice

- Count Inversions
- Search in 2D Matrix

---

7. Sliding Window / Two Pointers

- Fixed and Variable Window
- Shrinking Window

🔁 Practice

- Longest Subarray with Sum K
- Minimum Window Substring
- Trapping Rain Water

---

8. Topological Sort (Graphs)

- Kahn’s Algorithm (BFS)
- DFS-based Topological Sort

🔁 Practice

- Course Schedule
- Alien Dictionary

🧩 MODULE 5: Real-World DSA Challenges

> Apply your knowledge to real-world and interview-style challenges.

🔷 Practice Platforms

- LeetCode (Target: 200+ problems)
- InterviewBit (Topic-wise progression)
- HackerRank (Warmup to Intermediate)
- Codeforces/CodeChef (Competitive)

🎯 MODULE 6: Interview Preparation & Strategy

1. Problem-Solving Strategy

- Read → Understand → Plan → Code → Test → Optimize

2. System Design Basics (Optional)

- Load Balancer, Caching, Database Indexing
- Scalability, Consistency

3. Behavioral Prep

- STAR Method
- Discussing Past Projects
- Explain Solutions Clearly