Data Structures :

- **Arrays**
    - Kadane's Algorithm (Maximum Subarray Sum)
    - Two Pointers (for array manipulation)
    - Sliding Window (for subarray problems)
    - Prefix Sum (range sum queries)

- **Strings**
    - String matching (Naive, KMP, Rabin-Karp)
    - Palindrome checking
    - Anagram checking

- **Linked Lists**
    - Reversal (In-place reversal, recursion)
    - Detecting cycles (Floyd’s Tortoise and Hare)
    - Merge two sorted lists

- **Stacks & Queues**
    - Implementing stacks and queues using arrays/linked lists
    - Monotonic Stack (next greater element, stock span problem)
    - BFS (Queue-based traversal)
    - DFS (Stack-based traversal)

Advanced Data Structures :

- **Heaps/Priority Queues**
  - Implementing priority queues (Min/Max Heap)
  - Top K elements
  - Kth largest/smallest element

- **Hash Tables (Maps/HashMap)**
  - Frequency counting (hashmap-based solutions)
  - Two-sum problem (using hashmap)
  - Implementing a hashmap from scratch

- **Trees**
  - Binary Trees (DFS, BFS)
  - Binary Search Tree (BST) operations (insert, delete, search)
  - Balanced trees (AVL, Red-Black Tree)
  - Trie (Prefix Tree)

- **Graphs**
  - Graph representation (adjacency list/matrix)
  - BFS, DFS for graph traversal
  - Dijkstra's algorithm (Shortest path)
  - Kruskal's and Prim's algorithm (Minimum spanning tree)

- **Disjoint Set Union (Union-Find)**
  - Union-find operations (union by rank, path compression)

Algorithms :

- **Sorting**
    - Merge Sort, Quick Sort, Heap Sort
    - Counting Sort, Radix Sort (non-comparative sorting)
    - Bubble Sort, Selection Sort (basic sorting algorithms)

- **Searching**
    - Binary Search (for sorted arrays or rotated sorted arrays)
    - Modified Binary Search (for finding boundaries or conditions)

- **Recursion**
    - Factorial, Fibonacci
    - Tree traversal (in-order, pre-order, post-order)

- **Backtracking**
    - N-Queens problem
    - Sudoku solver
    - Permutations and combinations

- **Divide and Conquer**
    - Merge Sort, Quick Sort
    - Binary Search
    - Closest pair of points

Advanced Algorithms :

- **Dynamic Programming (DP)**
    - Fibonacci numbers (memoization vs tabulation)
    - Knapsack problem (0/1 knapsack, unbounded knapsack)
    - Longest Common Subsequence (LCS)
    - Longest Increasing Subsequence (LIS)
    - Coin Change problem

- **Greedy Algorithms**
    - Activity Selection
    - Huffman Encoding
    - Fractional Knapsack problem

Advanced Topics :

- **Bit Manipulation**
    - XOR problems (find unique numbers)
    - Bitwise operators (AND, OR, NOT, shifting)
    - Counting set bits (Hamming weight)

- **Matrix Manipulation**
    - Spiral traversal
    - Matrix multiplication
    - Matrix exponentiation

- **Topological Sorting**
    - Kahn's Algorithm (for Directed Acyclic Graphs)
    - Depth-First Search (DFS) based approach
- **String Matching Algorithms**
    - KMP (Knuth-Morris-Pratt)
    - Rabin-Karp
    - Boyer-Moore
- **Sliding Window**
    - Maximum/Minimum subarray sum in a sliding window

    - Longest substring with unique characters
    - String pattern matching
- **Graphs**
    - Floyd-Warshall (All pairs shortest path)
    - Topological Sorting (for Directed Acyclic Graphs)
    - Bellman-Ford Algorithm (for negative weights)

Java-Specific Topics :

- **Collections Framework**
    - Lists, Sets, Maps (HashMap, TreeMap, HashSet, TreeSet)
    - Queue (LinkedList, PriorityQueue)
    - Deque (ArrayDeque, LinkedList)
    - Concurrent Collections (ConcurrentHashMap)

- **Multithreading and Concurrency**
    - Thread synchronization
    - Thread pool (Executor framework)
    - Locks and semaphores

- **Streams and Lambda Expressions**
    - Using Stream API for data manipulation
    - Functional interfaces
    - Collectors, filters, maps, etc.

- **Memory Management**
    - Garbage collection in Java
    - Stack vs Heap memory

Problem Solving Paradigms :

- **Divide and Conquer**

- **Backtracking**

- **Dynamic Programming (DP)**

- **Greedy Algorithms**

- **Graph Algorithms**

Miscellaneous

- **Concurrency and Parallelism**

- **Database Design**

- **System Design**
    - Designing scalable systems (Load balancing, sharding, caching)
    - CAP Theorem (Consistency, Availability, Partition tolerance)

- **Mathematical Algorithms**
    - Sieve of Eratosthenes (for prime numbers)
    - Greatest Common Divisor (GCD)
    - Least Common Multiple (LCM)
    - Fast Exponentiation (Exponentiation by Squaring)