Complete DSA Roadmap for Placement Drives :→

Algorithms :→

Searching Algorithms :→
* Linear Search
* Binary Search
* Recursive Binary Search
* Jump Search
* Interpolation Search

Time Complexities

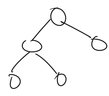
Sorting Algos :→
* Bubble Sort
* Selection Sort
* Insertion Sort
* Merge Sort
* Quick Sort
* Count Sort
* Radix Sort
* Shell Sort
* Bucket Sort
* Heap Sort

Advanced Algos :→
* Kadane's Algo
* Robin Karp Algo
* Sieve of Eratosthenes
* Euclidean Algo for the GCD
* Pythagorean Triplets

Data Structures

Linear
* Arrays
* Strings
* Linked Lists → Singly / Doubly / Circular
* Stacks
* Queues

Non-Linear
* Trees
  - Normal Tree
  - Binary Tree
  - Traversals
    - DFS → Pre Order / In Order / Post Order
    - BFS → BFS or Level Order
  - Not strictly height balanced ← Binary Search Tree
  - Strictly height balanced → AVL Trees / Red Black Trees / K-d Trees / Skewed Trees / Segment Trees / B/B+ Trees (DB) / Binary Indexed Tree (Fenwick Tree)
  - Insertion Left to Right, CBT → Complete Binary Tree → Heap (Max & Min Heaps)
  - Tries
  - Graphs → Types
  - Representation
    - Adj List
    - Adj Matrix
  - Traversal
    - DFS
    - BFS (queue)
  - Cycle Detection



Graph Algos :→
* Topological Sort (Linear Ordering) DAG
* Topological Sort (Kahn's Algorithm)
* Shortest Distance Algos :
  - (I) Dijkstra's
  - (II) Bellman Ford
  - (III) Floyd Warshall's
  - (IV) Minimum Spanning Tree → Prim's Algorithm / Kruskal's Algorithm
  - (V) Disjoint Set
    - Union by rank / size
    - find parent
  - (VI) Strongly Connected Components
  - (VII) Kosaraju's Algorithm
  - (VIII) Problems :→ CF4, Leet Code, Coding Ninjas

* Recursion → Back Tracking
  - (I) N Queens
  - * (II) Rat In A Maze
  - (III) Subsets of a String / Array
  - (IV) Sudoku Solver
  - * (V) Phone Keypad Problem
* Recursion → Dynamic Programming
  - (I) Recursion
  - (II) Memoisation
  - (III) Tabulation
  - (IV) Space Optimization
* Greedy Algorithms : (Max & Min Heap)
  - (I) Chocolate Distribution
  - (II) Minimum No of Coins
  - (III) Activity Selection Problem
  - (IV) Huffman Encoding
  - (V) Fractional Knapsack
  - (VI) Min Cost of Ropes
  - (VII) Job Sequencing
  - (VIII) Nikunj & Donuts
* Bit Masking / Bit Manipulation
  - (I) Position of Set Bit
  - (II) Two Unique Elements
  - (III) Hamming Weight
  - (IV) Power of 2

The 4 Pillars of Object Oriented Programming

(I) **Encapsulation :→** The process of binding the data members, fields inside a class by using the "private" access modifier, is called "Encapsulation". It is done so that the data is not accidentally modified.

To access the data outside the class we use two special methods which are public :
- (I) Setters → set the attributes
- (II) getters → get / retrieve / fetch the attributes

(II) **Inheritance :→** The property by virtue of which a class can inherit / use all the attributes / methods of it's parent is known as 'inheritance'.

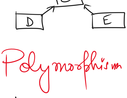
There are generally 5 types of inheritance that we use in the industry :

(I) Single Level: Animal (Super) → Dog (Sub)

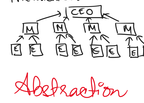
(II) Multi-level: Grand Parent → Parent → Children

(III) Multiple (Not Supported Directly in Java): Father, Mother → Child

(IV) Hybrid:



(V) Hierarchical:



* Polymorphism
* Abstraction
* Association
* Exception Handling
* STL Libraries