**Day 1: Arrays**

□ Set Matrix Zeros

□ Pascal's Triangle

□ Next Permutation

□ Kadane's Algorithm

□ Sort an array of 0's, 1's and 2's

□ Stock Buy and Sell

**Day 2: Arrays Part-II**

□ Rotate Matrix

□ Merge Overlapping Subintervals

□ Merge two sorted arrays without extra space

□ Find the duplicate in an array of N+1 integers

□ Repeat and Missing Number

□ Inversion of Array (Pre-req: Merge Sort)

**Day 3: Arrays Part-III**

□ Search in a 2 D matrix

□ Pow(x, n)

□ Majority Element (>n/2 times)

□ Majority Element (n/3 times)

□ Grid Unique Paths

□ Reverse Pairs (Leetcode)

**Day 4: Arrays Part-IV**

□ 2Sum Problem

□ 4-Sum Problem

□ Longest Consecutive Sequence

□ Largest Subarray with K sum

□ Count number of subarrays with given xor K

□ Longest Substring without repeat

**Day 5: Linked List**

□ Reverse a LinkedList

□ Find the middle of LinkedList

□ Merge two sorted Linked List (use method used in mergeSort)

□ Remove N-th node from back of LinkedList

□ Add two numbers as LinkedList

□ Delete a given Node when a node is given.(0(1) solution)

**Day 6: Linked List Part-II**

□ Find intersection point of Y LinkedList

□ Detect a cycle in Linked List

□ Reverse a LinkedList in groups of size k.

□ Check if a LinkedList is palindrome or not.

□ Find the starting point of the Loop of LinkedList

□ Flattening of a LinkedList

**Day 7: Linked List and Arrays**

□ Rotate a LinkedList

□ Clone a Linked List with random and next pointer

□ 3 sum

□ Trapping Rainwater

□ Remove Duplicate from Sorted array

□ Max consecutive ones

**Day 8: Greedy Algorithm**

□ N meetings in one room

□ Minimum number of platforms required for a railway

□ Job sequencing Problem

□ Fractional Knapsack Problem

□ Greedy algorithm to find minimum number of coins

□ Assign Cookies

**Day 9: Recursion**

□ Subset Sums

□ Subset-II

□ Combination sum-1

□ Combination sum-2

□ Palindrome Partitioning

□ K-th permutation Sequence

**Day 10: Recursion and Backtracking**

□ Print all permutations of a string/array

□ N queens Problem

□ Sudoko Solver

□ M Coloring Problem

□ Rat in a Maze

□ Word Break (print all ways)

**Day 11: Binary Search**

□ The N-th root of an integer

□ Matrix Median

□ Find the element that appears once in a sorted array, and the rest element appears twice (Binary search)

□ Search element in a sorted and rotated array/ find pivot where it is rotated

□ Median of 2 sorted arrays

□ K-th element of two sorted arrays

□ Allocate Minimum Number of Pages

□ Aggressive Cows

**Day 12: Heaps**

□ Max heap, Min Heap Implementation (Only for interviews)

□ Kth Largest Element

□ Maximum Sum Combination

□ Find Median from Data Stream

□ Merge K sorted arrays

□ K most frequent elements

**Day 13: Stack and Queue**

□ Implement Stack using Arrays

□ Implement Queue using Arrays

□ Implement Stack using Queue (using single queue)

□ Implement Queue using Stack (0(1) amortized method)

□ Check for balanced parentheses

□ Next Greater Element

□ Sort a Stack

**Day 14: Stack and Queue Part-II**

□ Next Smaller Element

□ LRU cache (IMPORTANT)

□ LFU cache

□ Largest rectangle in a histogram

□ Sliding Window maximum

□ Implement Min Stack

□ Rotten Orange (Using BFS)

□ Stock span problem

□ Find the maximum of minimums of every window size

□ The Celebrity Problem

**Day 15: String**

□ Reverse Words in a String

□ Longest Palindrome in a string

□ Roman Number to Integer and vice versa

□ Implement ATOI/STRSTR

□ Longest Common Prefix

□ Rabin Karp

**Day 16: String Part-II**

□ Z-Function

□ KMP algo / LPS(pi) array

□ Minimum characters needed to be inserted in the beginning to make it palindromic

□ Check for Anagrams

□ Count and say

□ Compare version numbers

**Day 17: Binary Tree**

□ Inorder Traversal

□ Preorder Traversal

□ Postorder Traversal

□ Morris Inorder Traversal

□ Morris Preorder Traversal

□ LeftView Of Binary Tree

□ Bottom View of Binary Tree

□ Top View of Binary Tree

□ Preorder inorder postorder in a single traversal

□ Vertical order traversal

□ Root to Node Path in Binary Tree

□ Max width of a Binary Tree

**Day 18: Binary Tree part-II**

□ Level order Traversal / Level order traversal in spiral form

□ Height of a Binary Tree

□ Diameter of Binary Tree

□ Check if the Binary tree is height-balanced or not

□ LCA in Binary Tree

□ Check if two trees are identical or not

□ Zig Zag Traversal of Binary Tree

□ Boundary Traversal of Binary Tree

**Day 19: Binary Tree part-III**

□ Maximum path sum

□ Construct Binary Tree from inorder and preorder

□ Construct Binary Tree from Inorder and Postorder

□ Symmetric Binary Tree

□ Flatten Binary Tree to LinkedList

□ Check if Binary Tree is the mirror of itself or not

□ Check for Children Sum Property

**Day 20: Binary Search Tree**

□ Populate Next Right pointers of Tree

□ Search given Key in BST

□ Construct BST from given keys

□ Construct a BST from a preorder traversal

□ Check is a BT is BST or not

□ Find LCA of two nodes in BST

□ Find the inorder predecessor/successor of a given Key in BST.

**Day 21: Binary Search Tree Part-II**

□ Floor in a BST

□ Ceil in a BST

□ Find K-th smallest element in BST

□ Find K-th largest element in BST

□ Find a pair with a given sum in BST

□ BST iterator

□ Size of the largest BST in a Binary Tree

□ Serialize and deserialize Binary Tree

**Day 22: Binary Trees[Miscellaneous]**

□ Binary Tree to Double Linked List

□ Find median in a stream of running integers.

□ K-th largest element in a stream.

□ Distinct numbers in Window.

□ K-th largest element in an unsorted array.

□ Flood-fill Algorithm

**Day 23: Graph**

□ Clone a graph (Not that easy as it looks)

□ DFS

□ BFS

□ Detect A cycle in Undirected Graph using BFS

□ Detect A cycle in Undirected Graph using DFS

□ Detect A cycle in a Directed Graph using DFS

□ Detect A cycle in a Directed Graph using BFS

□ Topological Sort BFS

□ Topological Sort DFS

□ Number of islands(Do in Grid and Graph Both)

□ Bipartite Check using BFS

□ Bipartite Check using DFS

**Day 24: Graph Part-II**

□ Strongly Connected Component(using KosaRaju’s algo)

□ Dijkstra’s Algorithm

□ Bellman-Ford Algo

□ Floyd Warshall Algorithm

□ MST using Prim's Algo

□ MST using Kruskal’s Algo

**Day 25: Dynamic Programming**

□ Max Product Subarray

□ Longest Increasing Subsequence

□ Longest Common Subsequence

□ 0-1 Knapsack

□ Edit Distance

□ Maximum sum increasing subsequence

□ Matrix Chain Multiplication

**Day 26: Dynamic Programming Part-II**

□ Minimum sum path in the matrix, (count paths and similar type do, also backtrack to find the Minimum path)

□ Coin change

□ Subset Sum

□ Rod Cutting

□ Egg Dropping

□ Word Break

□ Palindrome Partitioning (MCM Variation)

□ Maximum profit in Job scheduling

**Day 27: Trie**

□ Implement Trie (Prefix Tree)

□ Implement Trie - 2 (Prefix Tree)

□ Longest String with All Prefixes

□ Number of Distinct Substrings in a String

□ Power Set (this is very important)

□ Maximum XOR of two numbers in an array

□ Maximum XOR With an Element From Array