

These are the questions which i did during my preparation.

# DSA SHEET by NISHANT CHAHAR

## Question Link

### **Stacks**

- 1 [Next Greater Element on right](#)
- 2 [Next Greater Element 2](#)
- 3 [Daily Temperatures](#)
- 4 [maximum difference between left and right smaller](#)
- 5 [Stock Span Problem](#)
- 6 [Largest Rectangular Area Histogram](#)
- 7 [maximu size binary matrix containing 1](#)
- 8 [Valid Parentheses](#)
- 9 [Length of longest valid substrng](#)
- 10 [Count of duplicate Parentheses](#)
- 11 [Decode String](#)
- 12 [Minimum Add To make Parentheses Valid](#)
- 13 [Print Bracket Number](#)
- 14 [Asteroid Collision](#)
- 15 [Backspace String Compare](#)
- 16 [Print Binary Number](#)
- 17 [Score Of String](#)
- 18 [Remove K digits From number](#)
- 19 [Car fleet](#)
- 20 [First negative Integer in k sized window](#)
- 21 [Addition](#)
- 22 [Gas Station](#)
- 23 [Maximum sum of smallest and second smallest](#)
- 24 [Min Stack](#)
- 25 [K stacks in a single array](#)
- 26 [Validate Stack](#)
- 27 [K reverse in a queue](#)
- 28 [largest Pair sum in unsorted array](#)

### **Linked Lists**

- 1 [reverse LinkedList](#)
- 2 [K reverse](#)
- 3 [Floyd cycle](#)
- 4 [Merge LinkedList](#)
- 5 [Clone a linkedlist](#)
- 6 [find modular node](#)
- 7 [Remove duplicate from sorted](#)
- 8 [Find the middle element](#)
- 9 [Nth element from end](#)
- 10 [LRU Cache](#)

## Binary Tree

- 1 [Inorder Traversal](#)
- 2 [Preorder Traversal](#)
- 3 [Postorder Traversal](#)
- 4 [Print ancestor of given tree](#)
- 5 [Binary Tree Level Order](#)
- 6 [Average of levels](#)
- 7 [All Nodes at distance K](#)
- 8 [Count bst in a given range](#)
- 9 [Binary search tree to greater sum](#)
- 10 [Binary Tree Cameras](#)
- 11 [Binary Tree Maximum Path Sum](#)
- 12 [Binary Tree to BST](#)
- 13 [right side view](#)
- 14 [Left View](#)
- 15 [Vertical order](#)
- 16 [Top View](#)
- 17 [Bottom View](#)
- 18 [Diagonal Traversal](#)
- 19 [leftmost and rightmost node](#)
- 20 [kth smallest element](#)
- 21 [Binary Tree Tilt](#)
- 22 [Print all nodes that dont have siblings](#)
- 23 [House robber 3](#)
- 24 [Boundary Traversal](#)

## Binary search tree

- 1 [Lowest common ancestor in BST](#)
- 2 [Lowest common ancestor](#)
- 3 [square root decomposition](#)
- 4 [Delete Node in BST](#)
- 5 [Construct from inorder and preorder](#)
- 6 [Construct from inorder and postorder](#)
- 7 [construct bst using postorder](#)
- 8 [Inorder and level order](#)
- 9 [serialize and deserialise](#)
- 10 [Distribute coins in a binary tree](#)
- 11 [duplicate subtree in a binary tree](#)

## Mixed from tree (General tree, AVL,BST)

- 1 [AVL tree](#)
- 2 [image multiplication](#)
- 3 [Binary TREE longest consecutive sequence](#)
- 4 [diameter of a tree](#)
- 5 [Kth smallest element of BST](#)
- 6 [clone a binary tree with random pointer](#)

- 7 [Flatten binary tree to linked list](#)
- 8 [Convert a binary tree to circular doubly linked list](#)
- 9 [Conversion of sorted DLL to BST](#)
- 10 [Merge Two BST](#)
- 11 [Pair violating BST property](#)
- 12 [Flip binary tree to match preorder](#)
- 13 [inorder sucesor](#)
- 14 [Rabbits in forest](#)

## **Arrays & strings ( STL )**

- 1 [Array of doubled Pair](#)
- 2 [Find smallest size of string containing all char of other](#)
- 3 [Longest consecutive 1's](#)
- 4 [number of subarrays sum exactly k](#)
- 5 [Subarray sum Divisible by k](#)
- 6 [longest substring with unique character](#)
- 7 [subarray with equal number of 0 and 1](#)
- 8 [Substring with equal 0 1 and 2](#)
- 9 [same frequency after one removal](#)
- 10 [K closest point from origin](#)
- 11 [Anagram Pallindrome](#)
- 12 [Minimum number of refueling spots](#)
- 13 [Find all anagrams in a string](#)
- 14 [K anagram](#)
- 15 [smallest number whose digit mult to given no.](#)
- 16 [Group anagram](#)
- 17 [Huffman coding](#)
- 18 [Isomorphic string](#)
- 19 [Check AP sequence](#)
- 20 [Count Pair whose sum is divisible by k](#)
- 21 [smallest subarray with all the occurence of MFE](#)
- 22 [Morning Assembly](#)
- 23 [Kth smallest element in sorted 2d matrix](#)
- 24 [Kth smallest prime fraction](#)
- 25 [Max points on a line](#)
- 26 [Brick wall](#)
- 27 [Array Pair sum divisibility](#)
- 28 [A simple fraction](#)
- 29 [Grid illumination](#)
- 30 [Insert Delete GetRandom O\(1\)](#)
- 31 [Count of substring with k 1](#)
- 32 [Incomplete array](#)
- 33 [Long Pressed Name](#)
- 34 [Range Addition](#)
- 35 [Max range query](#)
- 36 [Magic Squares In Grid](#)

- 37 [Next Greater Element III](#)
- 38 [Orderly Queue](#)
- 39 [maximum subarray](#)
- 40 [K-CON](#)
- 41 [Rotate Array](#)
- 42 [Remove Duplicates from Sorted Array](#)
- 43 [X of a kind in a deck](#)
- 44 [merge k sorted array](#)
- 45 [Grid illumination](#)
- 46 [Kth smallest after removing natural numbers](#)
- 47 [rearrange character string such that no two are same](#)
- 48 [Longest consecutive sequence](#)
- 49 [length of largest subarray with continuous element](#)
- 50 [length of largest subarray with cont element 2](#)
- 51 [Anagram mapping](#)
- 52 [Employee Free time](#)
- 53 [Line reflection](#)

## Heap

- 1 [Binary heap](#)
- 2 [Build heap from array](#)
- 3 [Island perimeter](#)
- 4 [skyline problem](#)
- 5 [Pairs of coinciding points](#)
- 6 [trapping rain water](#)
- 7 [Trapping Rain Water II](#)
- 8 [Sort a nearly sorted array](#)
- 9 [bulb switcher](#)
- 10 [max frequency stack](#)
- 11 [Sliding window maximum](#)
- 12 [Swim in rising water](#)
- 13 [Heap sort](#)
- 14 [Product of Array Except Self](#)
- 15 [K empty slots](#)

16

## 17 mathematics

18

- 19 [Sieve of Eratosthenes](#)
- 20 [Segmented sieve](#)
- 21 [Squares of a Sorted Array](#)
- 22 [Fast Exponentiation](#)
- 23 [Fibonacci Number](#)
- 24 [Container With Most Water](#)

## Searching & Sorting

- 1 [Segregate 0 and 1](#)
- 2 [Segregate 0-1-2](#)
- 3 [Sort Array By Parity](#)
- 4 [Min Jump required with +i or -i allowed](#)
- 5 [Max chunks to make sorted](#)
- 6 [Max Chunks To Make Sorted II](#)
- 7 [Two Sum](#)
- 8 [Two Difference](#)
- 9 [LPS](#)
- 10 [Shortest Palindrome](#)
- 11 [Boats to Save People](#)
- 12 [Min No. of Platform](#)
- 13 [Maximum Swap](#)
- 14 [Optimal Division](#)
- 15 [Max Consecutive Ones II](#)
- 16 [max consecutive ones 3](#)
- 17 [majority element](#)
- 18 [majority element 2](#)
- 19 [majority element general](#)
- 20 [Reverse vowels of a string](#)
- 21 [First missing positive](#)
- 22 [push dominoes](#)
- 23 [moving stones until consecutive 2](#)
- 24 [max product of 3 numbers](#)
- 25 [largest number atleast twice of others](#)
- 26 [maximum product subarray](#)
- 27 [rotate image](#)
- 28 [number of subarrays with bounded maximum](#)
- 29 [partition labels](#)
- 30 [global and local inversions](#)
- 31 [partition array into disjoint intervals](#)
- 32 [valid pallindrome 2](#)
- 33 [consecutive number sum](#)
- 34 [minimum domino rotation for equal row](#)
- 35 [multiply strings](#)
- 36 [smallest range from k lists](#)
- 37 [pascal triangle 2](#)
- 38 [max sum of two non overlapping subarrays](#)
- 39 [maximize distance to closest person](#)
- 40 [Subarrays with k different integers](#)
- 41 [Icing on cake](#)
- 42 [search in rotated sorted array](#)
- 43 [split array largest sum](#)
- 44 [counting sort](#)
- 45 [capacity to ship within D days](#)

- 46 [insertion sort](#)
- 47 [koko eating bananas](#)
- 48 [median of two sorted array](#)
- 49 [merge sort](#)
- 50 [smallest divisor given a threshold](#)
- 51 [wiggle sort](#)
- 52 [best meeting points](#)

## Graph

- 1 [BFS of graph](#)
- 2 [Bipartite graph](#)
- 3 [DFS](#)
- 4 [detect cycle in undirected graph](#)
- 5 [Prim's Algo](#)
- 6 [Dijkstra algo](#)
- 7 [chef and reversing](#)
- 8 [Bus routes](#)
- 9 [evaluate division](#)
- 10 [topological sorting](#)
- 11 [Kahn's algo](#)
- 12 [course schedule 2](#)
- 13 [Strongly Connected Components \(Kosaraju's Algo\)](#)
- 14 [Mother Vertex](#)
- 15 [Rotting Oranges](#)
- 16 [bellman ford](#)
- 17 [Number of Islands](#)
- 18 [DSU](#)
- 19 [Number of Enclaves](#)
- 20 [Most Stones Removed with Same Row or Column](#)
- 21 [Regions Cut By Slashes](#)
- 22 [Kruskal's algo](#)
- 23 [Articulation point](#)
- 24 [Doctor Strange](#)
- 25 [Satisfiability of Equality Equations](#)
- 26 [0-1 matrix](#)
- 27 [Word Ladder](#)
- 28 [Job Sequencing](#)
- 29 [Eulerian Path in an Undirected Graph](#)
- 30 [Euler Circuit in a Directed Graph](#)
- 31 [Castle RUN](#)
- 32 [Sentence Similarity II](#)
- 33 [Number of Distinct Islands](#)
- 34 [Number of Islands II](#)
- 35 [Parallel courses](#)
- 36 [optimize water distribution in village](#)
- 37 [connecting cities with minimum cost](#)

## Dynamic programming

- 1 [Minimize Malware Spread](#)
- 2 [climbing stairs](#)
- 3 [Jump game 2](#)
- 4 [Min cost path](#)
- 5 [max size subsquare with all 1](#)
- 6 [0-1 Knapsack](#)
- 7 [fractional knapsack](#)
- 8 [longest increasing subsequence](#)
- 9 [longest increasing subsequence](#)
- 10 [minimum number of increasing subsequence](#)
- 11 [building bridges](#)
- 12 [Box stacking](#)
- 13 [max sum alternating subsequence](#)
- 14 [best time to buy and sell stock](#)
- 15 [best time to buy and sell 2](#)
- 16 [best time to buy and sell 3](#)
- 17 [best time to buy and sell 4](#)
- 18 [best time to buy and sell with cool down](#)
- 19 [buy and sell with transaction time](#)
- 20 [Ugly number](#)
- 21 [Super ugly number](#)
- 22 [Domino and tromino tiling](#)
- 23 [Wildcard pattern matching](#)
- 24 [Regular expression matching](#)
- 25 [Count all pallindromic subsequences](#)
- 26 [Regular expression matching](#)
- 27 [Count all pallindromic subsequence](#)
- 28 [Count distinct pallindromic subsequence](#)
- 29 [Count of binary string without consecutive 1](#)
- 30 [Max sum with no 2 adjacent element](#)
- 31 [Pizza with 3n slices](#)
- 32 [LCS triplet](#)
- 34 [Edit distance](#)
- 35 [Frog jump](#)
- 36 [Friends pairing problem](#)
- 37 [Partition of sets into k subsets](#)
- 38 [Can i win](#)
- 39 [Knight probability](#)
- 40 [Temple offering](#)
- 41 [Highway billboard problem](#)
- 42 [No. of sequence of type  \$a^i+b^j+c^k\$](#)
- 43 [boolean parenthesization](#)
- 44 [Min and max with + and \\*](#)
- 45 [Optimal BST](#)

- 46 [Find water in glass](#)
- 47 [cherry pickup](#)
- 48 [arithmetic slices](#)
- 49 [arithmetic slices 2](#)
- 50 [Largest sum subarray atleast k numbers](#)
- 51 [Maximum sum of 3 non overlapping subarrays](#)
- 52 [Remove min element according to constraint](#)
- 53 [Scramble string](#)
- 54 [Minimum score triangulation](#)
- 55 [2 keys keyboard](#)
- 56 [4 keys keyboard](#)
- 57 [Mobile numeric keypad](#)
- 58 [Word break](#)
- 59 [burst balloons](#)
- 60 [Encode string with shortest length](#)
- 61 [longest repeating subsequence](#)
- 62 [String is k pallindromic or not](#)
- 63 [Count distinct subsequence](#)
- 64 [Shortest uncommon subsequence](#)
- 65 [minimal moves to form a string](#)
- 66 [Paint fence](#)
- 67 [Paint house](#)
- 68 [Paint house 2](#)

## **BFS/DFS**

### [Sliding Puzzle](#)

- 1 [Find the Maximum Flow](#)
- 2 [Maximum Bipartite Matching](#)
- 3 [Reconstruct Itinerary](#)
- 4 [Redundant Connection](#)
- 5 [Redundant connection 2](#)
- 6 [Possible Bipartition](#)
- 7 [Floyd Warshall](#)
- 8 [Johnson's algorithm](#)
- 9 [Journey to the moon](#)
- 10 [Sort item by group accord to dependencies](#)
- 11 [As far from land as possible](#)
- 11 [K-Similar Strings](#)
- 12 [Similar String Groups](#)
- 13 [Coloring A Border](#)
- 14 [Shortest bridge](#)
- 15 [Min swaps required to sort array](#)
- 16 [Walls and gates](#)
- 17 [The maze 2](#)

## **Text processing**

- 1 [KMP](#)
- 2 [Find string roots](#)
- 3 [Z algo](#)



4 [chef and secret password](#)

5 [Manacher's algo](#)

## Number theory

1 [Euclidean algorithm](#)

2 [Extended Euclidean algorithm](#)

3 [Linear diophantine equation](#)

4 [Euler's totient function](#)

5 Divisors upto n

6 [Fermat's little theorem](#)

7 [No min No max](#)

8 [Boring factorials](#)

9 [FFT](#)

## Geometry

1 [Erect the fence](#)

## Game Theory

1 [5 Pirates and 100 coins](#)

2 [Nim game](#)

3 [Buddy nim](#)



















**DONE**





































**HINT**

































































































































































































































