Master Data Structures and Algorithms

About us

We are a group of IITians who recently cracked companies like Flipkart, Jaguar Land Rover, Amazon, Vedantu, JioSaavn, Zomato, Arista etc, and are guiding students.

You can join our telegram group to get in touch and ask your queries:
https://t.me/dream_placement

Youtube Channel: **https://www.youtube.com/c/KayDee24/featured**

The DS-Algo questions list is prepared by us after analyzing all the websites available and taking insights from our own preparation and experiences. This question set if done properly is enough to prepare for maximum companies.

Pre-requisites:

- 1 Coding language (C++, Java or Python) along with basic functionalities like array declaration, loops, if-else statements, and declaring variables.
- Sorting, Hashing (Basic)
- Basic time and space complexities of loop traversals & sorting.
- For C++ users: STL like Vector, Map, Set (to implement hashing), Sort function (to sort).

Note: Vector is nothing but an array just with some flexibilities like inserting as many elements without initially declaring size. Going through geeksforgeeks STL will be enough for getting comfortable with there implementation.

Refer Tushar Roy, Abdul Bari and Jenny's Lectures videos in case you need help for most of your doubts.

1. ARRAYS:

1. Rotate an array
2. Leaders in an array
3. Pascal triangle
4. Max Sum Subarray
5. Find first missing and repeating number
6. Stock Buy and Sell
7. Majority Element (Study all possible methods)
8. Minimum Platforms
9. Merge overlapping intervals
10. Elements on left smaller and on right greater

11. Trapping Rainwater
12. Maximum Index
13. Move all zeros to the end of the array (Also try: Remove duplicates from the sorted array)
14. Max sum subarray of size k
15. Chocolate Distribution Problem
16. Find first element occurring k times
17. Minimum swaps to sort
18. Sort elements of an array by frequency
19. Product of an array except for self
20. Rearrange array alternatively
21. Find repeating element in an array (Also try: Smallest positive missing number)

2D Array:

22. Spiral matrix
23. Rotate matrix
24. Set matrix zeroes
25. Max-sum square sub-matrix

Advance:

-Minimize the heights
-3 partition
-Maximum unsorted subarray
-Next Permutation
-Kth smallest element
-Max Sum Triplet

2. STRINGS:

Note: Strings are very similar to arrays because a string is nothing but an array of characters only.

1. Palindrome string
2. Longest Common Prefix
3. Zig zag string
4. Anagrams
5. Atoi
6. Number following a pattern
7. Count and say
8. Find one extra character in a string
9. Find if strings are rotations of each other or not

Advance:

- Longest Substring without repeating character
-Smallest Window containing all character of another string
-Design a tiny URL or URL shortener
-Longest Palindrome string (Do it in O(1) space and O(n^2) time)

3. Hashing and 2 Pointer

(Famous methods to solve questions in efficient space and time complexities)**:**

Hashing:

1. 2 out of 3
2. Valid Sudoko
3. 2 sum, 3 sum, 4 sum
4. Largest subarray with given sum k
5. Largest subarray with equal 0s and 1s
6. Distinct elements in every window of size k
7. Subarray with given xor

2 Pointer:

1. Container with most water
2. Remove duplicates from a sorted array
3. Two numbers with sum closest to zero
4. 3 elements sum lies closest to a given k
5. No of triangles formed from an array
6. 3 sorted arrays
7. Geek collect the balls

Advance:

-Points in a straight line
-Subarray with B odd numbers
-Pair with given diff
-Max ones after Modification
-Smallest distinct window

4. Linked List:

1. Reverse a Linked List

2. Reverse a Linked List in groups
3. Reverse alternate k nodes
4. Palindrome Linked List
5. Remove duplicates from Linked List
6. Print mid of Linked List
7. Find nth node from end of linked list
8. Remove loop in linked list and find that point
9. Delete node without head pointer
10. Sort list
11. Merge k sorted Linked List
12. Intersection point in Y shaped Linked List
13. Clone a Linked List

5. Stacks and Queues:

Stacks:

1. Implement stack using array
2. Reverse a sentence using stack
3. Balanced parentheses
4. Longest balanced parenthesis
5. Rain Water Trapping
6. Stock span problem
7. Next larger element (Also try next smaller element)
8. Max rectangle area in histogram (Hint: Using the concept of next larger element)
9. Minimum element of stack
10. Celebrity problem
11. Evaluation of expressions

Queue:

1. Queue using 2 stacks
2. Decode string
3. Reverse First K elements of Queue
4. First non-repeating character in a stream
5. Rotten oranges
6. Max of all subarrays of size k (Deque)
7. LRU cache

Heap (Actually a tree but can be implemented using STL Priority Queue)**:**

1. Merge k sorted arrays
2. Min Cost Ropes

3. Distinct numbers in a window
4. N max pair combination

6. Recursion & Backtracking

(Focus only on the basics, no need to go very advance)

Recursion:

1. Print 1 to n without loop
2. Print factorial of a number
3. Nth fibonacci number
4. Print all possible subsequences (Best video solution)
5. Print all strings
6. Tower of Hanoi

Backtracking:

Beginners first watch this: https://www.youtube.com/watch?v=S3rnLLHI0PM

1. Subset
2. Subset 2

8. Binary Search

(There are 2 ways to implement this algorithm, either by recursion or loop, you may use either)

1. Square root
2. Rotated sorted array search
3. Find element in a bitonic array
4. Left most and right most index
5. Matrix Search
6. Sorted matrix median
7. Painter Partition Problem / Allocate Books
8. Usage of Upper/Lower bound STL

9. Trees

(We will deal with binary tree only for almost all the questions at our level)

1. Create a Binary Tree and Binary Search Tree (BST)
2. Search in a BST
3. Find height of a tree/ Min Depth
4. Level Order Traversal / Zig-Zag Level Order Traversal
5. Pre/ In / Post Order Traversal
6. Iterative Pre/ In / Post order traversal
7. Given 2 traversals, create binary tree I, II , III

8. Check if 2 trees are identical
9. Count number of leaves of a tree / Sum of a tree
10. Left / Right View of a tree
11. Mirror Tree
12. Check for balanced tree
13. Symmetric Tree
14. Diameter of a tree
15. Lowest Common Ancestor (LCA)
16. All nodes at K distance from root
17. Print path from root to a particular node
18. Vertical order traversal
19. Top / Bottom view
20. Boundary Traversal
21. Sum of all left leaf nodes
22. Swap leaf nodes
23. Max path sum between 2 leaves
24. Remove all the half nodes

10. Graphs

(BFS and DFS are the 2 main pillars of most of the graph questions)
Understand ways of representing graphs first.

1. Breadth First Search (BFS)
2. Depth First Search (DFS)
3. Find no. of islands
4. Maximum area island
5. Snake and Ladder
6. Word Ladder I and II
7. Shortest path in a matrix
8. Detect Cycle in a directed graph
9. Detect Cycle in an undirected graph
10. Topological Sort
11. Shortest path from 1 source to all other sources or any particular source in weighted graph with +Ve Edges - Dijkstra algo | -Ve Edges included - Bellman ford algo

Advance:

12. Minimum Spanning Tree
13. Strongly connected components
14. Articulation points

11. Dynamic Programming

(Just a concept to solve questions efficiently, nothing fancy)

1. Count ways to reach nth stair
2. Thief Problem
3. Longest Increasing Subsequence
4. Longest Common Subsequence
5. Minimum jumps array
6. Largest square formed
7. 0-1 Knapsack
8. Coin Change
9. Coin Change II
10. Rod Cutting
11. Edit Distance
12. Subset Sum
13. Longest Palindromic subsequence
14. Best time to buy and sell stock

Advance:

-Word Break
-Min sum partition

Summary:

– Mostly Max or Min Nikalana hoga

1. Think of a recursion solution
2. Create an array or a matrix and try to fit in the question
3. Fill the array/matrix by feeling the problem ;p

10. Bit manipulation

1. Trailing Zeroes
2. Reverse bits
3. Number of 1 bits
4. Different bits sum pairwise
5. XORing the subarrays

Miscellaneous:

1. KMP algorithm
2. Merge 2 array without extra space
3. Inversion count
4. Median of row wise sorted matrix
5. Median of an infinite stream & Median of 2 sorted sets
6. Rotate a matrix by 90 degrees

7. Tries Implementation

Problem Solving Approach:

1. Read the question carefully and understand some test cases first.
2. Then try to think of solutions, try various approaches and test your approaches on test cases. Give at least 30 minutes to a question before opening the solution on paper.
3. Writing the code and submitting should be the last step once you are confident your approach is working on many test cases.
4. Patience is the key here and people who will struggle with the questions will learn much more as compared to ones who see the solution within a few minutes after reading the question.
5. The goal is to develop problem-solving skills through these questions and not mugging them up, so try build that instinct of learning something new from every question.
6. Your notes you make are your Bhagavad gita, keep revising them so that you don't forget after a few months.