Free DSA Course Master Data Structures and Algorithms with LeetCode

WHO ARE WE?

At **Code & Debug**, we are a dedicated online training platform focused on empowering students to achieve their career aspirations. Our mission is to bridge the gap between theoretical knowledge and real-world application, ensuring that our students are fully equipped to tackle the challenges of modern tech interviews and secure their dream jobs.

We have successfully trained over 15,000 students on a regular basis, guiding them through a structured learning process designed to enhance their problem-solving skills, coding abilities, and confidence. Our courses are meticulously crafted to cover essential concepts in programming, data structures, algorithms, and more, with a focus on hands-on practice and real-time interview scenarios.

WHY CODE & DEBUG?

At **Code & Debug**, we understand the challenges faced by students preparing for technical interviews and career transitions. Here's why we stand out:

Expert Guidance

Our instructors are seasoned professionals with extensive industry experience, offering insights into real-world interview processes and coding challenges. They know exactly what top tech companies are looking for and will help you sharpen your skills accordingly.

Structured Learning Path

We provide a meticulously designed curriculum that starts from the basics and progresses to advanced concepts in Data Structures, Algorithms, and programming. You'll build a solid foundation and master complex topics through a step-by-step approach.

200+ Curated Leetcode Problems

We provide a curated list of Leetcode problems categorized by difficulty and topic, ensuring you're well-prepared for coding interviews. With hands-on practice, you'll gain the confidence to solve complex algorithmic problems.

Doubt-Solving Sessions

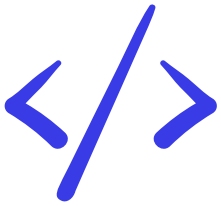
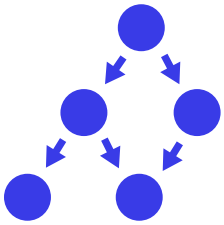
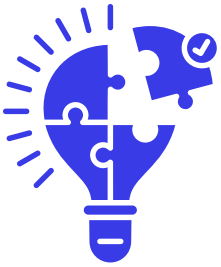
Practice is key to success, and we reinforce this with daily assignments to challenge your skills. Additionally, our regular sunday doubt-clearing sessions ensure that no question goes unanswered.

Focused on Interview Success

From coding problems to behavioral interview prep, we cover all aspects of the hiring process, giving you a comprehensive toolkit to excel in interviews.

With Code & Debug, you're not just learning to code – you're preparing to **build your career**.

What You'll Learn (Key Skills and Concepts)

 Foundational Programming Concepts	 Data Structures Mastery	 Algorithmic Thinking
 Advanced Problem-Solving Techniques	 Interview-Ready Problem Sets	 Dynamic Programming & Recursion

SYLLABUS BREAKDOWN

At **Code & Debug**, we believe in delivering a well-rounded, thorough learning experience that covers every crucial aspect of Data Structures and Algorithms (DSA). Our syllabus is designed to take you from foundational concepts to advanced problem-solving techniques. Here's a detailed breakdown of what you'll be learning throughout the course:

1. Basic Maths Logic Buildup

- Count Digits
- Reverse a Number: Reverse Integer
- Check Palindrome: Palindrome Number
- GCD or HCF
- Armstrong Number
- Print all Divisors of a Number
- Check Prime

2. Recursion Basics

- Print 1 to N without Loop
- Print N times with Recursion
- Print N to 1 without Loop
- Sum of First N Natural Numbers without Loop
- Factorial of N Numbers
- Reverse an Array without Loop
- Check if String is Palindrome: Valid Palindrome
- Fibonacci Series: Fibonacci Number

3. Hashing

- Learn Basics of Hashing
- Count Frequency in a Range
- Highest / Lowest Frequency Elements: Frequency of the Most Frequent Element

4. Different Types of Sorting

- Selection Sort
- Bubble Sort
- Insertion Sort
- Merge Sort
- Recursive Bubble Sort
- Recursive Insertion Sort
- Quick Sort

5. Problems on Arrays

Easy Level Problems

- Merge 2 Sorted Arrays: Union of Two Sorted Arrays
- Find Missing Number in an Array: Missing Number
- Maximum Consecutive Ones: Max Consecutive Ones
- Contains Duplicate: Contains Duplicate

Medium Level Problems

- 2Sum Problem: Two Sum
- Kadane's Algorithm, Maximum Subarray Sum: Maximum Subarray
- Stock Buy and Sell: Best Time to Buy and Sell Stock
- Container With Most Water: Container With Most Water
- Longest Consecutive Sequence in an Array: Longest Consecutive Sequence
- Set Matrix Zeros: Set Matrix Zeroes
- Rotate Matrix by 90 Degrees: Rotate Image
- Print the Matrix in Spiral Manner: Spiral Matrix

Hard Level Problems

- 3Sum Problem: 3Sum
- Maximum Product Subarray: Maximum Product Subarray

6. Binary Search

Problems based on 1D Lists

- Binary Search to Find X in Sorted Array: Binary Search
- Implement Lower Bound
- Implement Upper Bound
- Search Insert Position: Search Insert Position
- Floor/Ceil in Sorted Array
- Find the First or Last Occurrence of a Given Number in a Sorted Array: Find First and Last Position of Element in Sorted Array
- Count Occurrences of a Number in a Sorted Array with Duplicates
- Search in Rotated Sorted Array I: Search in Rotated Sorted Array
- Search in Rotated Sorted Array II: Search in Rotated Sorted Array II
- Find Minimum in Rotated Sorted Array: Find Minimum in Rotated Sorted Array

7. Strings

Easy Level Problems

- Check if Two Strings are Anagram of Each Other: Valid Anagram
- Group Anagrams: Group Anagrams

Medium Level Problems

- Longest Palindromic Substring [Without DP]: Longest Palindromic Substring

8. Linked List (Singly and Doubly Linked Lists)

Learn Singly Linked List

- Introduction to LinkedList
- Inserting a Node in LinkedList
- Deleting a Node in LinkedList: Delete Node in a Linked List
- Find the Length of the LinkedList
- Search an Element in the LinkedList
- Design Linked List: Design Linked List

Learn Doubly Linked List

- Introduction to Doubly Linked List
- Insert a Node in Doubly Linked List
- Delete a Node in Doubly Linked List
- Reverse a Doubly Linked List

Medium Problems on Singly Linked List

- Middle of a LinkedList [Tortoise-Hare Method]: Middle of the Linked List
- Reverse a LinkedList [Iterative]: Reverse Linked List
- Reverse a LinkedList [Recursive]: Reverse Linked List
- Detect a Loop in LinkedList: Linked List Cycle
- Find the Starting Point of the Loop in LinkedList: Linked List Cycle II
- Merge Two Sorted Lists: Merge Two Sorted Lists
- Reorder List: Reorder List
- Remove Nth Node from the Back of the LinkedList: Remove Nth Node From End of List

9. Bit Manipulation

Learn the basics of bit manipulation

- Introduction to Bit Manipulation
- Check if the i-th Bit is Set or Not
- Check if a Number is Odd or Not
- Check if a Number is Power of 2 or Not: Power of Two
- Count the Number of Set Bits: Number of 1 Bits
- Set/Unset the Rightmost Unset Bit
- Swap Two Numbers
- Divide Two Integers Without Using Multiplication, Division, and Mod Operator: Divide Two Integers

Problems based on bit manipulation

- Count Number of Bits to be Flipped to Convert A to B: Minimum Bit Flips to Convert Number
- Find the Number that Appears Odd Number of Times: Single Number
- Sum of Two Integers: Sum of Two Integers
- Number of 1 Bits: Number of 1 Bits
- Counting Bits: Counting Bits

- Find Missing Number in an Array: Missing Number
- Reverse Bits: Reverse Bits

10. Advance Recursion

Basic logic buildup

- Recursive Implementation of atoi
- Pow(x, n)
- Count Good numbers
- Sort a stack using recursion
- Reverse a stack using recursion

Problems based on SubSequences

- Generate All Binary Strings
- Generate Parentheses: Generate Parentheses
- Print All Subsequences/Power Set: Subsets
- Learn All Patterns of Subsequences
- Count All Subsequences with Sum K
- Check if There Exists a Subsequence with Sum K
- Combination Sum: Combination Sum
- Combination Sum II: Combination Sum II
- Combination Sum III: Combination Sum III
- Combination Sum IV: Combination Sum IV
- Word Search: Word Search
- Word Break: Word Break

11. Stacks and Queues

Basic Implementation of Stack and Queue

- Implement Stack using Arrays
- Implement Queue using Arrays
- Implement Stack using Queue
- Implement Queue using Stack
- Implement stack using Linkedlist
- Implement queue using Linkedlist
- Check for balanced paranthesis
- Implement Min Stack

Problems based on Stack/Queues

- Next Greater Element
- Next Greater Element 2

12. Sliding Window and Two Pointers

- Longest Substring Without Repeating Characters: Longest Substring Without Repeating Characters
- Longest Repeating Character Replacement: Longest Repeating Character Replacement

- Longest Substring with At Most K Distinct Characters: Longest Substring with At Most K Distinct Characters
- Minimum Window Substring: Minimum Window Substring

13. Heaps

- Introduction to Priority Queues Using Binary Heaps
- Min Heap and Max Heap Implementation
- Convert Min Heap to Max Heap
- Kth Largest Element in an Array: Kth Largest Element in an Array
- Kth Smallest Element in an Array
- Merge M Sorted Lists: Merge k Sorted Lists
- Find Median from Data Stream: Find Median from Data Stream
- K Most Frequent Elements: Top K Frequent Elements

14. Greedy Algorithms

Easy Level Problems

- Assign Cookies: Assign Cookies
- Fractional Knapsack Problem
- Greedy Algorithm to Find Minimum Number of Coins
- Lemonade Change: Lemonade Change
- Valid Parenthesis Checker: Valid Parenthesis String

Medium/Hard Level Problems

- N Meetings in One Room
- Jump Game: Jump Game
- Jump Game II: Jump Game II
- Minimum Number of Platforms Required for a Railway
- Job Sequencing Problem
- Insert Interval: Insert Interval
- Merge Intervals: Merge Intervals
- Non-overlapping Intervals: Non-overlapping Intervals

15. Binary Trees

Different types of Traversals in BT

- Introduction to Trees
- Create Binary Tree
- Binary Tree Traversals
- Preorder Traversal of Binary Tree: Binary Tree Preorder Traversal
- Inorder Traversal of Binary Tree: Binary Tree Inorder Traversal
- Post-order Traversal of Binary Tree: Binary Tree Postorder Traversal
- Level Order Traversal: Binary Tree Level Order Traversal
- Iterative Preorder Traversal of Binary Tree
- Iterative Inorder Traversal of Binary Tree
- Post-order Traversal Using 2 Stacks
- Post-order Traversal Using 1 Stack

- Preorder, Inorder, and Postorder Traversal in One Traversal

Medium Level Problems

- Height of a Binary Tree: Maximum Depth of Binary Tree
- Check if the Binary Tree is Height-Balanced: Balanced Binary Tree
- Diameter of Binary Tree: Diameter of Binary Tree
- Maximum Path Sum: Binary Tree Maximum Path Sum
- Check if Two Trees are Identical: Same Tree
- Invert Binary Tree: Invert Binary Tree

Hard Level Problems

- Requirements Needed to Construct a Unique Binary Tree
- Construct Binary Tree from Inorder and Preorder: Construct Binary Tree from Preorder and Inorder Traversal
- Construct Binary Tree from Postorder and Inorder Traversal: Construct Binary Tree from Inorder and Postorder Traversal
- Serialize and Deserialize Binary Tree: Serialize and Deserialize Binary Tree
- Subtree of Another Tree: Subtree of Another Tree

16. Binary Search Trees

- Introduction to Binary Search Tree
- Search in a Binary Search Tree
- Find Min/Max in BST
- Ceil in a Binary Search Tree
- Floor in a Binary Search Tree
- Insert a Node in Binary Search Tree: Insert into a Binary Search Tree
- Delete a Node in Binary Search Tree: Delete Node in a BST
- Find K-th Smallest/Largest Element in BST: Kth Smallest Element in a BST
- Check if a Tree is a BST: Validate Binary Search Tree
- Lowest Common Ancestor in Binary Search Tree: Lowest Common Ancestor of a Binary Search Tree
- Construct a BST from Preorder Traversal: Construct Binary Search Tree from Preorder Traversal

17. Graphs

Introduction

- Graph and Types
- Graph Representation
- Connected Components
- BFS
- DFS

Problems based on DFS and BFS

- Pacific Atlantic Water Flow: Pacific Atlantic Water Flow
- Connected Components in a Matrix: Number of Provinces
- Rotten Oranges: Rotting Oranges

- Flood Fill: Flood Fill

Topo Sort Problems

- Topological Sort
- Kahn's Algorithm
- Cycle Detection in Directed Graph (BFS)
- Course Schedule I: Course Schedule
- Course Schedule II: Course Schedule II
- Alien Dictionary: Alien Dictionary
- Number of Islands: Number of Islands

18. Dynamic Programming

Introduction

- Dynamic Programming Introduction
- What do we use DP?

Problems based on 1D List

- Climbing Stairs: Climbing Stairs
- Maximum Sum of Non-Adjacent Elements: House Robber
- House Robber II: House Robber II

Problems based on 2D/3D List

- Ninja's Training
- Grid Unique Paths: Unique Paths
- Grid Unique Paths II: Unique Paths II

Problems based on Subsequences

- Subset Sum Equal to Target
- Minimum Coins: Coin Change

Problems based on Strings

- Longest Common Subsequence: Longest Common Subsequence
- Print Longest Common Subsequence
- Longest Common Substring
- Decode Ways: Decode Ways

Problems based on Stocks Buy and Sell

- Best Time to Buy and Sell Stock: Best Time to Buy and Sell Stock
- Buy and Sell Stock II: Best Time to Buy and Sell Stock II
- Buy and Sell Stock III: Best Time to Buy and Sell Stock III
- Buy and Sell Stock IV: Best Time to Buy and Sell Stock IV
- Buy and Sell Stocks with Cooldown: Best Time to Buy and Sell Stock with Cooldown
- Buy and Sell Stocks with Transaction Fee: Best Time to Buy and Sell Stock with Transaction Fee

Problems based on LIS

- Longest Increasing Subsequence: Longest Increasing Subsequence
- Printing Longest Increasing Subsequence

19. Tries

- Implement Trie | Insert | Search | StartsWith: Implement Trie (Prefix Tree)
- Implement Trie II (Prefix Tree)
- Longest String with All Prefixes
- Number of Distinct Substrings in a String
- Bit Prerequisites for Trie Problems
- Maximum XOR of Two Numbers in an Array: Maximum XOR of Two Numbers in an Array
- Maximum XOR with an Element from Array: Maximum XOR With an Element From Array
- Design Add and Search Words Data Structure: Design Add and Search Words Data Structure

MEET YOU INSTRUCTOR

Anirudh Khurana – Lead Instructor & Founder

With over 10 years of experience in the tech industry and an in-depth knowledge of data structures and algorithms, Anirudh is a highly skilled educator. Having worked with top-tier companies, Anirudh brings practical insights and problem-solving techniques to the classroom. His personalized teaching style ensures that students grasp complex topics easily and are fully prepared for interviews. Anirudh has trained over 15,000 students, helping them land roles at major tech firms.

HOW TO ENROLL?

Enrolling in our **Free Data Structures and Algorithms Course** is simple and hassle-free. Just follow these easy steps to join the course:

Click on "Enroll Now"
Go to the course page on our website and click the **"Enroll Now"** button to begin the enrollment process.

Fill Out the Enrollment Form
A pop-up window will appear prompting you to enter your details. Please provide your **name**, **email address**, and **phone number**.

Submit Your Information
After entering your details, click on the **"Submit"** button. There's no payment required—enrollment is completely **free**!

Receive Confirmation
Once you've submitted the form, you'll receive a confirmation email with all the necessary information about the course, including how to access the content and any upcoming sessions.

Start Learning
You're all set! Dive into the course materials and start your journey to mastering Data Structures and Algorithms.

For questions and concerns, reach out to us!

support@codeanddebug.in
+91 97129 28220