Comprehensive DSA Course in Python A6 Batch

– Code and Debug

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.

350+ 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.

Personalized Mentorship

Every student is different, and we believe in personalized learning. Our instructors and mentors provide individual feedback, helping you improve where you need it most.

Daily Assignments and Doubt-Solving Sessions

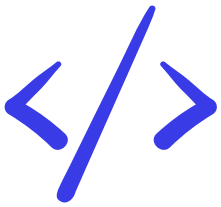
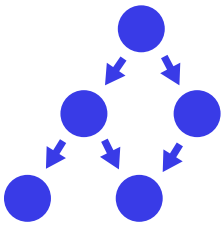
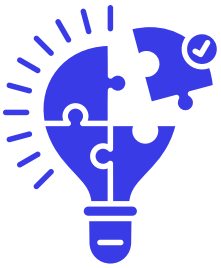
Practice is key to success, and we reinforce this with daily assignments to challenge your skills. Additionally, our regular doubt-clearing sessions ensure that no question goes unanswered.

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

BATCH DETAILS

We ensure that our batches are structured for maximum flexibility and effectiveness, catering to both beginners and experienced learners. Below are the details for our upcoming batch:

BATCH START DATE

The new **A6 Batch** kicks off on **20th January, 2025**. This carefully planned start date allows you enough time to prepare and organize your schedule for this intensive, structured course.

CLASS TIMINGS

Classes will be held from **7:00 PM to 8:30 PM**, Monday to Friday. These timings are designed to be accessible for students and working professionals alike, allowing you to

engage in learning after your daily commitments.

DURATION

This is a **4-month course**, offering a comprehensive dive into Data Structures and Algorithms. Each concept is broken down for thorough understanding, followed by rigorous practice.

DAYS

Monday to Friday: Interactive live sessions will be held every weekday, ensuring consistent learning. The weekends are reserved for extra doubt-clearing sessions or catching up with assignments.

ASSIGNMENTS

Daily Practice Assignments: Practice makes perfect, and we provide assignments after each class to reinforce what you've learned. These exercises range from basic to advanced, ensuring you're well-prepared for real-world challenges.

DOUBT CLEARING SESSIONS

Daily Doubt Sessions: After every class, you can clear any questions or confusion you have. Our instructors are available to address your queries and help you master difficult concepts.

Special Sunday Doubt Session: If you need additional support, we offer dedicated doubt-clearing sessions every Sunday, where you can ask questions, revise concepts, and get personalized help.

FLEXIBLE LEARNING

Can't attend live sessions due to work or personal reasons? Our flexible learning approach with recorded sessions allows you to revisit any class, ensuring that your learning experience is seamless.

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. Python Installation

- Installation of Python
- Installation of VS Code
- Installing VS Code extensions for code productivity

2. Basic Syntax, Variables and Data Types

- Basic syntax of print statement
- Introduction to Variables and Data Types
- Type casting or Type Conversion
- Truthy and Falsy Values

3. Operators in Python

- 6 Different types of operators in Python
- Using Operators using Truthy and Falsy Values

4. Conditional Statements

- Introduction to IF ELSE statements
- Using ELIF statements
- Code flow of multiple IF / ELIF statements
- Solving basic interview questions

5. Loops in Python

- Starting with While Loops
- Logic buildup using While loops
- Starting with For Loops
- Solving questions like Prime numbers in range, factors count, factorial and much more advance level questions based on loops
- Number patterns, Start patterns and more

6. Introduction to List and Tuple Data Type

- What are Lists? And its uses
- Mutable vs Immutable
- Indexing in Lists
- Iterate by Index and Iterate by Value
- Updating List Elements

- List Methods
- Logic buildup questions based on Lists
- Tuple vs Lists

7. Introduction to String Data Type

- What are Strings?
- Indexing in Strings
- Iterate by Index and Iterate by Value
- String Methods (Split, Join and more)
- Logic buildup questions based on Strings

8. Introduction to Dictionary Data Type

- What are Dictionary? And when it is used?
- Iterate by Key and Values in Dictionary
- Dictionary Methods such as update, pop, items and more
- Logic buildup questions based on Dictionary
- Frequency Dictionary questions

9. Introduction to Set Data Type

- What are Set? And when it is used?
- Difference between Sets and Dictionary
- Sets Methods
- Logic buildup questions based on Sets

10. Functions in Python

- What are Functions?
- Syntax of writing a function
- With parameters and with return Functions
- Default Parameters
- Required Parameters
- Named Parameters
- Args and Kwargs
- Returning multiple elements
- Local vs Global Variables
- Pass by Reference and Pass by Value

11. Exception Handling

- How to handle exceptions
- Raising an error
- Try Except Clauses

12. OOPS Concepts

- What is OOP?
- Procedural programming vs OOP

- Classes
- Objects
- Inheritance, Polymorphism, Abstraction, Encapsulation
- Working with List of Objects
- Banking Application
- Library Application

13. File Handling

- Different modes to open a file
- Read, Write and Append
- Questions based on File Handling
- Combining two or more files
- Binary Files

14. Introduction to Data Structures and Algorithms

- Overview of DSA
- Importance of DSA in Problem Solving
- Big O Notation and Time Complexity

15. Basic Maths Logic Buildup

- Count Digits
- Reverse a number
- Check palindrome
- GCD or HCF
- Armstrong Number
- Print all Divisors of a number
- Check Prime

16. Recursion Introduction

- 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
- Fibonacci Series

17. Hashing Concept

- Learn basics of hashing
- Count Frequency in a range
- Highest / Lowest Frequency Elements

18. Different types of Sorting

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

19. Problems on Arrays / Lists

Easy Level Problems

- Largest Element in an Array
- Second Largest Element in an Array without sorting
- Check if the array is sorted
- Remove duplicates from Sorted array
- Left Rotate an array by one place
- Right rotate an array by K places
- Move Zeros to end
- Linear Search
- Merge 2 sorted Arrays
- Find missing number in an array
- Maximum Consecutive Ones
- Find the number that appears once, and other numbers twice.

Medium Level Problems

- Longest subarray with given sum K(positives)
- Longest subarray with sum K (Positives + Negatives)
- 2Sum Problem
- Sort an array of 0's 1's and 2's
- Majority Element (>n/2 times)
- Kadane's Algorithm, maximum subarray sum
- Print subarray with maximum subarray sum
- Stock Buy and Sell
- Rearrange the array in alternating positive and negative items
- Next Permutation
- Leaders in an Array problem
- Longest Consecutive Sequence in an Array
- Set Matrix Zeros
- Rotate Matrix by 90 degrees
- Print the matrix in spiral manner
- Count subarrays with given sum

Hard Level Problems

- Pascals Triangle
- Majority Element (n/3 times)
- 3-Sum Problem
- 4-Sum Problem
- Largest Subarray with 0 Sum

- Count number of subarrays with given xor K
- Merge Overlapping Subintervals
- Merge two sorted arrays without extra space
- Find the repeating and missing number
- Count Inversions
- Reverse Pairs
- Maximum Product Subarray

20. Binary Search

Problems based on 1D Lists

- Binary Search to find X in sorted array
- Implement Lower Bound
- Implement Upper Bound
- Search Insert Position
- Floor/Ceil in Sorted Array
- Find the first or last occurrence of a given number in a sorted array
- Count occurrences of a number in a sorted array with duplicates
- Search in Rotated Sorted Array I
- Search in Rotated Sorted Array II
- Find minimum in Rotated Sorted Array
- Find out how many times has an array been rotated
- Single element in a Sorted Array
- Find peak element

Problems based on Answers

- Find square root of a number in log n
- Find the Nth root of a number using binary search
- Koko Eating Bananas
- Minimum days to make M bouquets
- Find the smallest Divisor
- Capacity to Ship Packages within D Days
- Kth Missing Positive Number
- Aggressive Cows
- Book Allocation Problem
- Split array - Largest Sum
- Painter's partition
- Minimize Max Distance to Gas Station
- Median of 2 sorted arrays
- Kth element of 2 sorted arrays

21. Strings

Easy Level Problems

- Remove outermost Paranthesis
- Reverse words in a given string / Palindrome Check
- Largest odd number in a string
- Longest Common Prefix

- Isomorphic String
- Check whether one string is a rotation of another
- Check if two strings are anagram of each other

Medium Level Problems

- Sort Characters by frequency
- Maximum Nesting Depth of Paranthesis
- Roman Number to Integer and vice versa
- Implement Atoi
- Count Number of Substrings
- Longest Palindromic Substring [Do it without DP]
- Sum of Beauty of all substring
- Reverse Every Word in A String

Hard Level Problems

- Minimum number of bracket reversals needed to make an expression balanced
- Count and say
- Hashing In Strings
- Rabin Karp
- Z-Function
- KMP algo / LPS(pi) array
- Shortest Palindrome
- Longest happy prefix
- Count palindromic subsequence in given string

22. Singly and Doubly Linked List

Singly Linked List Problems

- Introduction to LinkedList
- Inserting a node in LinkedList
- Deleting a node in LinkedList
- Find the length of the linkedlist
- Search an element in the LL
- Design Linked List
- Middle of a LinkedList [TortoiseHare Method]
- Reverse a LinkedList [Iterative]
- Reverse a LL [Recursive]
- Detect a loop in LL
- Find the starting point in LL
- Length of Loop in LL
- Check if LL is palindrome or not
- Segrregate odd and even nodes in LL
- Remove Nth node from the back of the LL
- Delete the middle node of LL
- Sort LL
- Sort a LL of 0's 1's and 2's by changing links
- Find the intersection point of 2 LL
- Add 1 to a number represented by LL

- Add 2 numbers in LL

Doubly Linked List Problems

- Introduction to DLL
- Insert a node in DLL
- Delete a node in DLL
- Reverse a DLL
- Delete all occurrences of a key in DLL
- Find pairs with given sum in DLL
- Remove duplicates from sorted DLL

23. 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
- Count the number of set bits
- Set/Unset the rightmost unset bit
- Swap two numbers
- Divide two integers without using multiplication, division and mod operator

Problems based on bit manipulation

- Count number of bits to be flipped to convert A to B
- Find the number that appears odd number of times
- Power Set
- Find xor of numbers from L to R
- Find the two numbers appearing odd number of times

24. 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 Paranthesis
- Print all subsequences/Power Set
- Learn All Patterns of Subsequences
- Count all subsequences with sum K
- Check if there exists a subsequence with sum K
- Combination Sum

- Combination Sum-II
- Subset Sum-I
- Subset Sum-II
- Combination Sum - III
- Letter Combinations of a Phone number

Commonly asked Interview Questions

- Palindrome Partitioning
- Word Search
- N Queen
- Rat in a Maze
- Word Break
- M Coloring Problem
- Sudoko Solver
- Expression Add Operators

25. 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
- Next Smaller Element
- Number of NGEs to the right
- Trapping Rainwater
- Sum of subarray minimum
- Asteroid Collision
- Sum of subarray ranges
- Remove k Digits
- Largest rectangle in a histogram
- Maximal Rectangles

26. Sliding Window and Two Pointers

- Longest Substring Without Repeating Characters
- Max Consecutive Ones III
- Fruit Into Baskets
- Longest repeating character replacement
- Binary subarray with sum

- Count number of nice subarrays
- Number of substring containing all three characters
- Maximum point you can obtain from cards
- Longest Substring with At Most K Distinct Characters
- Subarray with k different integers
- Minimum Window Substring
- Minimum Window Subsequence

27. 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 [use priority queue]
- Kth smallest element in an array [use priority queue]
- Merge M sorted Lists
- Replace each array element by its corresponding rank
- Task Scheduler
- Hands of Straights
- Design twitter
- Connect n ropes with minimal cost
- Kth largest element in a stream of running integers
- Maximum Sum Combination
- Find Median from Data Stream
- K most frequent elements

28. Greedy Algorithms

Easy Level Problems

- Assign Cookies
- Fractional Knapsack Problem
- Greedy algorithm to find minimum number of coins
- Lemonade Change
- Valid Paranthesis Checker

Medium/Hard Level Problems

- N meetings in one room
- Jump Game
- Jump Game 2
- Minimum number of platforms required for a railway
- Job sequencing Problem
- Candy
- Program for Shortest Job First (or SJF) CPU Scheduling
- Program for Least Recently Used (LRU) Page Replacement Algorithm
- Insert Interval
- Merge Intervals
- Non-overlapping Intervals

29. Binary Trees

Different types of Traversals in BT

- Introduction to Trees
- Create Binary Tree
- Binary Tree Traversals in Binary Tree
- Preorder Traversal of Binary Tree
- Inorder Traversal of Binary Tree
- Post-order Traversal of Binary Tree
- Level order Traversal / Level order traversal in spiral form
- Iterative Preorder Traversal of Binary Tree
- Iterative Inorder Traversal of Binary Tree
- Post-order Traversal of Binary Tree using 2 stack
- Post-order Traversal of Binary Tree using 1 stack
- Preorder, Inorder, and Postorder Traversal in one Traversal

Medium Level Problems

- Height of a Binary Tree
- Check if the Binary tree is height-balanced or not
- Diameter of Binary Tree
- Maximum path sum
- Check if two trees are identical or not
- Zig Zag Traversal of Binary Tree
- Boundary Traversal of Binary Tree
- Vertical Order Traversal of Binary Tree
- Top View of Binary Tree
- Bottom View of Binary Tree
- Right/Left View of Binary Tree
- Symmetric Binary Tree

Hard Level Problems

- Root to Node Path in Binary Tree
- LCA in Binary Tree
- Maximum width of a Binary Tree
- Check for Children Sum Property
- Print all the Nodes at a distance of K in a Binary Tree
- Minimum time taken to BURN the Binary Tree from a Node
- Count total Nodes in a COMPLETE Binary Tree
- Requirements needed to construct a Unique Binary Tree
- Construct Binary Tree from inorder and preorder
- Construct the Binary Tree from Postorder and Inorder Traversal
- Serialize and deserialize Binary Tree
- Morris Preorder Traversal of a Binary Tree
- Morris Inorder Traversal of a Binary Tree
- Flatten Binary Tree to LinkedList

30. 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 given Node in Binary Search Tree
- Delete a Node in Binary Search Tree
- Find K-th smallest/largest element in BST
- Check if a tree is a BST or BT
- LCA in Binary Search Tree
- Construct a BST from a preorder traversal
- Inorder Successor/Predecessor in BST
- Merge 2 BST's
- Two Sum In BST | Check if there exists a pair with Sum K
- Recover BST | Correct BST with two nodes swapped
- Largest BST in Binary Tree

31. Graphs

Introduction

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

Problems based on DFS and BFS

- Number of provinces
- Connected Components Problem in Matrix
- Rotten Oranges
- Flood fill
- Cycle Detection in undirected Graph (bfs)
- Cycle Detection in undirected Graph (dfs)
- 0/1 Matrix (Bfs Problem)
- Surrounded Regions (dfs)
- Number of Enclaves [flood fill implementation - multisource]
- Word ladder - 1
- Word ladder - 2
- Number of Distinct Islands [dfs multisource]
- Bipartite Graph (DFS)
- Cycle Detection in Directed Graph (DFS)

Topo Sort Problems

- Topo Sort
- Kahn's Algorithm
- Cycle Detection in Directed Graph (BFS)
- Course Schedule - I
- Course Schedule - II
- Find eventual safe states
- Alien dictionary

Shortest Path Problems

- Shortest Path in UG with unit weights
- Shortest Path in DAG
- Djisktra's Algorithm
- Why priority Queue is used in Djisktra's Algorithm
- Shortest path in a binary maze
- Path with minimum effort
- Cheapest flights within k stops
- Network Delay time
- Number of ways to arrive at destination
- Minimum steps to reach end from start by performing multiplication and mod operations with array elements
- Bellman Ford Algorithm
- Floyd Warshal Algorithm
- Find the city with the smallest number of neighbors in a threshold distance

Minimum Spanning Tree and Disjoint Sets

- Minimum Spanning Tree
- Prim's Algorithm
- Disjoint Set [Union by Rank]
- Disjoint Set [Union by Size]
- Kruskal's Algorithm
- Number of operations to make network connected
- Most stones removed with same rows or columns
- Accounts merge
- Number of island II
- Making a Large Island
- Swim in rising water

32. Dynamic Programming

Introduction

- Dynamic Programming Introduction
- What do we use DP?

Problems based on 1D List

- Climbing Stars
- Frog Jump
- Frog Jump with k distances
- Maximum sum of non-adjacent elements
- House Robber

Problems based on 2D/3D List

- Ninja's Training
- Grid Unique Paths : DP on Grids
- Grid Unique Paths 2

- Minimum path sum in Grid
- Minimum path sum in Triangular Grid
- Minimum/Maximum Falling Path Sum
- 3-d DP : Ninja and his friends

Problems based on Subsequences

- Subset sum equal to target
- Partition Equal Subset Sum
- Partition Set Into 2 Subsets With Min Absolute Sum Diff
- Count Subsets with Sum K
- Count Partitions with Given Difference
- 0/1 Knapsack
- Minimum Coins
- Target Sum
- Coin Change 2
- Unbounded Knapsack
- Rod Cutting Problem

Problems based on Strings

- Longest Common Subsequence
- Print Longest Common Subsequence
- Longest Common Substring
- Longest Palindromic Subsequence
- Minimum insertions to make string palindrome
- Minimum Insertions/Deletions to Convert String
- Shortest Common Supersequence
- Distinct Subsequences
- Edit Distance
- Wildcard Matching

Problems based on Stocks Buy and Sell

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

Problems based on LIS

- Longest Increasing Subsequence
- Printing Longest Increasing Subsequence
- Longest Increasing Subsequence
- Largest Divisible Subset
- Longest String Chain
- Longest Bitonic Subsequence
- Number of Longest Increasing Subsequences

Problems based on Partition

- Matrix Chain Multiplication
- Matrix Chain Multiplication | Bottom-Up
- Minimum Cost to Cut the Stick
- Burst Balloons
- Evaluate Boolean Expression to True
- Palindrome Partitioning - II
- Partition Array for Maximum Sum

33. Tries

- Implement TRIE | INSERT | SEARCH | STARTSWITH
- Implement Trie - 2 (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 With an Element From Array

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 the **Data Structures and Algorithms Course (A6 Batch)** at **Code & Debug** is a straightforward process designed to get you started quickly on your learning journey. Follow the steps below to secure your spot:

Fill Out the Registration Form

- Navigate to the **"Enroll Now"** section on our website and fill out the simple registration form with your details including your name, email, and phone number. Ensure that your information is accurate for a seamless enrollment process.

Choose Your Payment Plan

- We offer flexible payment options to accommodate your needs. Select a plan that suits you and proceed to the payment gateway. All transactions are secure and encrypted to ensure your data's safety.

Make the Payment

- Complete the payment process using any major credit/debit card, UPI, net banking, or other available options. After payment, you'll receive an email confirming your enrollment along with further details about the course.

Receive Confirmation

- Upon successful payment, you'll receive a confirmation email with your login details, batch schedule, and instructions on how to access the course. You'll also be added to our student community, where you can engage with peers and instructors.

Get Ready for Class

- Mark your calendar for the **batch start date (20th January, 2025)** and prepare to dive into an immersive learning experience. You'll also receive reminders about upcoming sessions and access to course materials prior to the first class.

For questions and concerns, reach out to us!

info@codeanddebug.in
+91 97129 28220