Lakshya Online Course

Instructor led Live Online Course

Course Focus: Preparing students to crack coding interviews of all product companies which focus on Coding DSA skills like Google, Microsoft, Amazon, Adobe, Facebook, LinkedIn etc.

- **Program duration:** 64 hours
- **Session schedule:**
  - Weekend – 2 session per week (Sat & Sun) – 12 Weeks
  - Weekend – 3-4 session per week – 8 Weeks
- **Session length:** 2 -3 hours
- In-depth understanding of all the Data Structures & Algorithms
- Covering 200+ Interview problems and its variants
- Daily hands on coding
- Interactive sessions
- Weekly assignments
- Resume preparation

Course Curriculum:

Data Structures

- Arrays, Strings
- Stack, Queue
- Iteration & Recursion
- Linked List
- Binary Tree/ BST
- Heaps
- Hashes
- Graphs
- Time and space complexity
- Dynamic Programming

Optimization of Code

- Understanding complexity(time/space) of an algorithm/program
- Big O notation
- Brute Force Approach
- Backtracking/ Recursion
- Learning concept and learn to identify that a problem can be solved using BS
- Tips to avoid edge cases and overflow and underflow
- Concept and how things change at scale
- How to optimize further with tricks?

- How to identify a problem to be a DP problem
- How to change recursive to DP
- Which problems can be solved using greedy method
- How to get complexity of it
- What template all problems follow?
- How to fit your problems in that template?

Algorithms

- Understanding complexity(time/space) of an algorithm/program
- Big O notation
- Brute Force Approach
- Backtracking/ Recursion
- **Binary search algorithm and when to apply**
  - Learning concept and learn to identify that a problem can be solved using BS
  - Tips to avoid edge cases and overflow and underflow
- **Merge sort & Quick sort and what problems can be solved with them?**
  - Concept and how things change at scale
  - How to optimize further with tricks?
- **Dynamic programming approach : How to think in DP?**
  - How to identify a problem to be a DP problem
  - How to change recursive to DP
- **Greedy approach**
  - Which problems can be solved using greedy method
  - How to get complexity of it
- **Backtracking approach**
  - What template all problems follow?
  - How to fit your problems in that template?

Best Coding Practices

- Understanding execution of recursive functions
- How to scale a solution once you got it right?
- Improving time & space complexity
- Corner cases
- Invalid inputs
- Recursion termination conditions
- Logical errors
- Memory Leaks
- Test cases

Cracking Coding Interviews

- How to start solutioning?
- Do and Don’ts in interviews
- What are bare minimum for a solution?
- When to start optimizations?
- When to think of scale and complexity analysis?