

# DS and Algo Foundation



Detailed  
Course Syllabus

## 1. Analysis of Algorithm

- a. Analysis of Algorithm
- b. Order of Growth
- c. Asymptotic Notations
- d. Big O Notation
- e. Omega Notation
- f. Theta Notation
- g. Analysis of Common Loops
- h. Analysis of Recursion
- i. Space Complexity

## 2. Arrays

- a. Introduction to Arrays
- b. Array Types
- c. Operations on Arrays (Part 1)
- d. Operations on Arrays (Part 2)
- e. Largest Element
- f. Check if Sorted
- g. Second Largest Element
- h. Remove Duplicates from Sorted Array
- i. Move Zeros to End
- j. Reverse an Array
- k. Left Rotate an Array by One

## 3. Recursion

- a. Introduction to Recursion
- b. Applications of Recursion
- c. Recursion Output Practice(Part 1)
- d. Recursion Output Practice(Part 2)
- e. Print N to 1 Using Recursion
- f. Print 1 to N Using Recursion
- g. Tail Recursion
- h. Natural Number Sum using Recursion

- i. Writing Base Cases in Recursion
- j. Palindrome Check using Recursion
- k. Sum of Digits, Rod Cutting and Subsets
- l. Tower of Hanoi

## 4. Hashing

- a. Introduction to Hashing
- b. Hashing Application
- c. Direct Address Table
- d. Hashing Functions
- e. Collision Handling
- f. Chaining
- g. Implementation of Chaining
- h. Open Addressing
- i. Double Hashing
- j. Implementation of Open Addressing
- k. Chaining vs Open Addressing
- l. `Unordered_set` in C++ STL
- m. `Unordered_map` in C++ STL
- n. `HashSet` in Java
- o. `HashMap` in Java
- p. Count Distinct Elements
- q. Count of Frequency of array elements

## 5. String

- a. Introduction to String
- b. Strings in C++
- c. String in Java
- d. Palindrome Check
- e. Check for Anagram
- f. Leftmost Repeating Character
- g. Leftmost Non-repeating Element
- h. Overview of Pattern Searching
- i. Naive Pattern Searching
- j. Improved Naive Pattern Searching for Distinct
- k. Anagram Search
- l. Check if Strings are Rotations

- m. Rabin Karp Algorithm
- n. KMP Algorithm (Part 1 : Constructing LPS Array)
- o. KMP Algorithm (Part 2 : Complete Algorithm)
- p. Lexicographic Rank of a String

## 6. Searching

- a. Binary Search (Iterative)
- b. Binary Search (Recursive)
- c. Analysis of Binary Search
- d. Find Index of First Occurrence
- e. Find Index of Last Occurrence
- f. Count 1s in a Sorted Binary Array
- g. Square Root

## 7. Sorting

- a. Introduction to Linked List
- b. Simple Linked List Implementation in C++
- c. Simple Linked List Implementation in Java
- d. Traversing a Linked List in C++
- e. Traversing a Linked List in Java
- f. Recursive Traversal of Singly Linked List
- g. Search in a Linked List (Iterative and Recursive)
- h. Insert at Begin of Singly Linked List
- i. Insert at the end of Singly Linked List
- j. Delete First Node of Singly Linked List
- k. Delete Last of Singly Linked List
- l. Insert at given position in Singly Linked List
- m. Sorted Insert in a Singly Linked List
- n. Middle of linked list
- o. Nth Node from end of linked list
- p. Reverse a linked list iterative
- q. Recursive reverse a linked list (Part 1)
- r. Recursive reverse a linked list (Part 2)
- s. Remove duplicates from a sorted Singly Linked List

## 8. Circular Linked List

- a. Circular Linked List
- b. Circular Linked List Traversal in C++
- c. Circular Linked List Traversal in Java
- d. Insert at Begin of Circular Linked List
- e. Insert at the end of Circular Linked List
- f. Delete Head of Circular Linked List
- g. Delete Kth of a Circular Linked List

## 9. Doubly Linked List

- a. Doubly Linked List
- b. Insert at Begin of Doubly Linked List
- c. Insert at End Doubly Linked List
- d. Reverse a Doubly Linked List
- e. Delete Head of a Doubly Linked List
- f. Delete Last of a Doubly Linked List
- g. Circular Doubly Linked List

## 10. Stack

- a. Stack data structure
- b. Stack Applications
- c. Stack in C++ STL
- d. Stack in Java collections
- e. Stack Implementation
- f. Balanced Parenthesis
- g. Infix, Prefix and Postfix Introduction
- h. Infix to Postfix (Simple Solution)
- i. Infix to Postfix (Efficient Solution)
- j. Evaluation of Postfix
- k. Infix to Prefix (Simple Solution)
- l. Infix to Prefix (Efficient Solution)
- m. Evaluation of Prefix

## 11. Queue

- a. Queue Data Structure
- b. Application of Queue
- c. Implementation of queue using array
- d. Implementation queue using LinkedList
- e. Queue in C++ STL
- f. Queue in Java

## 12. Dequeue

- a. Deque Data Structure
- b. Dequeue in C++ STL
- c. Deque in Java
- d. Design a Data Structure with Min and Max operations

## 13. Tree

- a. Tree Data Structure
- b. Application of Tree
- c. Binary Tree
- d. Tree Traversal
- e. Implementation of Inorder Traversal
- f. Implementation of Preorder Traversal
- g. Implementation of Postorder Traversal
- h. Level Order Traversal
- i. Size of Binary Tree
- j. Maximum in Binary Tree
- k. Height of Binary Tree
- l. Iterative Inorder Traversal
- m. Iterative Preorder Traversal (Simple)
- n. Iterative Preorder Traversal (Space Optimized)

## 14. Binary Search Tree

- a. Binary Search Tree(Background)
- b. Binary Search Tree(Introduction)
- c. Search in BST (Introduction)

- d. Search in BST C++
- e. Search in BST Java
- f. Insert in BST
- g. Insert in BST C++
- h. Insert in BST Java
- i. Deletion in BST
- j. BST deletion in C++
- k. BST Deletion in Java

## 15. Heap

- a. Binary Heap Introduction
- b. Binary Heap Implementation
- c. Binary Heap Insert
- d. Binary Heap (Heapify and Extract)
- e. Binary Heap (Decrease Key, Delete and Build Heap)
- f. Priority Queue in C++
- g. PriorityQueue in Java

## 16. Graph

- a. Introduction to Graph
- b. Graph Representation (Adjacency Matrix)
- c. Graph Representation (Adjacency List)
- d. Adjacency List implementation in CPP
- e. Adjacency List implementation in Java
- f. Adjacency Matrix and List Comparison
- g. Breadth First Search
- h. Applications of BFS
- i. Depth First Search
- j. Applications of DFS
- k. Detect Cycle in Undirected Graph
- l. Detect Cycle in a Directed Graph (Part 1)
- m. Detect Cycle in a Directed Graph (Part 2)
- n. Topological Sorting (DFS Based Algorithm)
- o. Topological Sorting (Kahn's BFS Based Algorithm)

## 17. Greedy Algorithms

- a. Introduction to Greedy Algorithms
- b. Activity Selection Problem
- c. Fractional Knapsack
- d. Job Sequencing Problem

## 18. Dynamic Programming Algorithms

- a. Introduction to DP
- b. Dynamic Programming Memoization
- c. Dynamic Programming Tabulation
- d. Longest Common Subsequence (Part 1)
- e. Longest Common Subsequence (Part 2)
- f. Coin Change Count Combinations
- g. Edit Distance Problem
- h. Edit Distance Problem DP solution
- i. Longest Increasing Subsequence Problem
- j. Longest Increasing Subsequence  $O(n \log n)$
- k. Maximum Cuts
- l. Minimum coins to make a value
- m. Minimum Jumps to reach at end
- n. 0-1 knapsack problem
- o. 0-1 knapsack problem DP Solution
- p. Variation of LCS
- q. Variation of LIS (Part 1)
- r. Variations of LIS (Part 2)
- s. Count BSTs with n keys

## 19. Backtracking Algorithm

- a. Concepts of Backtracking
- b. Rat In a Maze
- c. N Queen Problem
- d. Sudoku Problem