|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.N** | Data Structures | | Algorithms | |
| **Topic List** | **Source** | **Topic List** | **Source** |
| 01 | Time & Space Complexity | Module-1 | Graph Representations (Adjacency matrix, Adjacency List, Edge list) | 1-4, 1-6, 1-8 |
| 02 | STL Vector | Module-2 | Breadth First Search (**BFS**) (Traversal, Level Tracking, Path printing) | 2-2, 2-4, 2-7 |
| 03 | Prefix Sum & Binary Search | Module-3 | Depth First Search (**DFS**) | 3-2 |
| 04 | Singly Linked List (create, print) | Module-5 | 2D Gridin DFS and BFS | 3-5, 3-7 |
| 05 | Operations on Singly Linked List (input, Insert: head, tail, position. Delete: head, tail, position) | Module-6 | Cycle Detection (undirected: BFS, DFS. Directed: DFS) | 5-2, 5-3, 5-5 |
| 06 | Singly Linked List Recap (Insert, delete, input) | Module-7 | **Dijkstra** Algorithm (Naïve, Optimized) | 6-4, 6-7 |
| 07 | Doubly Singly Linked List (insert, delete, input) | Module-9 | **Bellman Ford** Algorithm | 7-4 |
| 08 | STL List and Cycle Detection | Module-10 | **Floyd Warshall** Algorithm | 7-8 |
| 09 | Linked List related problem solving (leetcode) | Module-11 | Disjoint Set Union (**DSU**) | 10 |
| 10 | Stack Implementation and STL | Module-13 | Kruskal’s Algorithm for **MST** | 11-4 |
| 11 | Queue Implementation and STL | Module-14 |  |  |
| 12 | Problem Solving Using Stack & Queue (leetcode & coding ninjas) | Module-15 |  |  |
| 13 | Binary Tree Implementation (depth: pre, in, post) | Module-17 |  |  |
| 14 | Binary Tree Operations (input, level, height) | Module-18 |  |  |
| 15 | Binary Tree related problem solving (coding ninjas) | Module-19 |  |  |
| 16 | Binary Search Tree Implementation (search, insertion, convert) | Module-21 |  |  |
| 17 | Heap Implementation (insert, delete) | Module-22 |  |  |
| 18 | STL Priority Queue, Map and Set | Module-23 |  |  |