Week 1 :

- **Analysis of Algorithm**
  - Background analysis through a Program and its functions.
- **Asymptotic Notations**
  - Best, Average and Worst case explanation through a program.
- **Arrays**
- Introduction and Advantages
- Types of Arrays
  - Fixed-sized array
  - Dynamic-sized array
- Operations on Arrays
  - Searching
  - Insertions
  - Deletion
  - Arrays vs other DS
  - Reversing - Explanation with complexity
- **Problems**
  - Left Rotation of the array by 1
  - Left Rotation of the array by D places
  - Leaders in an Array
  - Maximum Difference Problem
  - Stock Buy and Sell Problem
  - Trapping Rainwater Problem
  - Maximum subarray sum
  - Longest even-odd subarray
  - Maximum Circular sum subarray.
  - Majority Element
  - Sliding Window Technique
  - Prefix sum technique etc.
- **Basic Recursion**

Week 2 :

- **Basic Bit Manipulation**
- Bitwise Operators in C++
  - Operation of AND, OR, XOR operators
  - Operation of Left Shift, Right Shift and Bitwise Not
- Bitwise Operators in Java
  - Operation of AND, OR
  - Operation of Bitwise Not, Left Shift
  - Operation of Right Shift and unsigned Right Shift

- Problem: Check Kth bit is set or not
    - Method 1: Using the left Shift.
    - Method 2: Using the right shift
- Problem: Count Set Bits
    - Method 1: Simple method
    - Method 2: Brian and Kerningham Algorithm
    - Method 3: Using Lookup Table
- Problem: To check whether a number is a power of 2 or not
- Problem: Odd occurrences in an array.
- Problem: Two numbers having odd occurrences in an array.
- Problem: Generate power set using bitwise operators.
- **Hashing**
- Introduction and Time complexity analysis
- Application of Hashing
- Discussion on Direct Address Table
- Working and examples on various Hash Functions
- Introduction and Various techniques on Collision Handling
- Chaining and its implementation
- Open Addressing and its Implementation
- Chaining V/S Open Addressing
- Double Hashing
- C++
    - Unordered Set
    - Unordered Map
- Java
    - HashSet
    - HashMap
- Problems
    - Count Distinct Elements
    - Count of the frequency of array elements
    - The intersection of two arrays
    - Union of two unsorted arrays
    - Pair with given sum in an unsorted array
    - Subarray with zero-sum
    - Subarray with given sum
    - Longest subarray with a given sum
    - Longest subarray with an equal number of 0’s and 1’s
    - Longest common span with the same sum in a binary array
    - Longest Consecutive Subsequence
    - Count Distinct elements in every window

Week 3

- **Strings**
- Discussion of String DS
- Problems
    - Given a string, check if they are an anagram of each other.
    - Given a string, find the leftmost character that repeats.
    - Given a string, find the leftmost character that does not repeat.
    - â€‹Given a string, find the lexicographic rank of it in O(n) time.
    - Implementation of the previously discussed lexicographic rank problem.
    - Given a text string and a pattern string, find if a permutation of the pattern exists in the text.
    - Given two strings, check if they are rotations of each other or not.
    - Various Pattern Searching Algorithms.
- **Linked Lists**
- Introduction
    - Implementation in CPP
    - Implementation in Java
    - Comparison with Array DS
- Doubly Linked List
- Circular Linked List
- Loop Problems
    - Detecting Loops
    - Detecting loops using Floyd cycle detection

Week 4

- **Linked List**
- Problem:
- Middle of Linked List
- Nth node from the end of linked list
- Deleting a Node without accessing Head pointer of Linked List
- An iterative method to Reverse a linked list
- Recursive method to reverse a linked list
- Segregating even-odd nodes of linked list
- The intersection of two linked list
- Pairwise swap nodes of linked list
- Clone a linked list using a random pointer

- LRU Cache Design
- **Stacks**
- Understanding the Stack data structure
- Applications of Stack
- Implementation of Stack in Array and Linked List
    - In C++
    - In Java
- **Problem:**
    - Balanced Parenthesis
    - Two stacks in an array
    - K Stacks in an array
    - Stock span problem with variations
    - Previous Greater Element
    - Next Greater Element
    - Largest Rectangular Area in a Histogram
- **Queues**
- Introduction and Application
- Implementation of the queue using array and LinkedList
    - In C++ STL
    - In Java
    - Stack using queue
- **Problem:**
    - Reversing a Queue
    - Generate numbers with given digits
    - Maximums of all subarrays of size k

Week 5
Linked List

- Problem:
- Middle of Linked List
- Nth node from the end of linked list
- Deleting a Node without accessing Head pointer of Linked List
- An iterative method to Reverse a linked list
- Recursive method to reverse a linked list
- Segregating even-odd nodes of linked list
- The intersection of two linked list
- Pairwise swap nodes of linked list
- Clone a linked list using a random pointer
- LRU Cache Design
- **Stacks**
- Understanding the Stack data structure
- Applications of Stack
- Implementation of Stack in Array and Linked List
    - In C++

    - In Java
- **Problem:**
    - Balanced Parenthesis
    - Two stacks in an array
    - K Stacks in an array
    - Stock span problem with variations
    - Previous Greater Element
    - Next Greater Element
    - Largest Rectangular Area in a Histogram
- **Queues**
- Introduction and Application
- Implementation of the queue using array and LinkedList
    - In C++ STL
    - In Java
    - Stack using queue
- **Problem:**
    - Reversing a Queue
    - Generate numbers with given digits
    - Maximums of all subarrays of size k

Week 5

- **Binary Tree**
- **Introduction**
    - Tree
    - Application
    - Binary Tree
    - Tree Traversal
- **Implementation of:**
    - Inorder Traversal
    - Preorder Traversal
    - Postorder Traversal
    - Level Order Traversal (Line by Line)
    - Tree Traversal in Spiral Form
- **Problems:**
    - Size of Binary Tree
    - Maximum in Binary Tree
    - Height of Binary Tree
    - Print Nodes at K distance
    - Print Left View of Binary Tree
    - Children Sum Property
    - Check for Balanced Binary Tree
    - Maximum Width of Binary Tree

    - Convert Binary Tree to Doubly Linked List
    - Construct Binary Tree from Inorder and Preorder
    - The diameter of a Binary Tree
    - LCA problem with an efficient solution
- **Binary Search Tree**
- Background, Introduction and Application
- Implementation of Search in BST
    - In CPP
    - In Java
- Insertion in BST
    - In CPP
    - In Java
- Deletion in BST
    - In CPP
    - In Java
- Floor in BST
    - In CPP
    - In Java
- Self Balancing BST
- AVL Tree
- Red Black Tree
- Set in C++ STL
- Map in C++ STL
- TreeSet in java
- TreeMap in Java
- Problems:
    - The ceiling of a key in BST
    - Ceiling on the left side in an array
    - Find Kth Smallest in BST
    - Check for BST
    - Fix BST with Two Nodes Swapped
    - Pair Sum with given BST
    - Vertical Sum in a Binary Tree
    - Vertical Traversal of Binary Tree
    - Top View of Binary Tree
    - Bottom View of Binary Tree

Week 6

- **Heaps**
- Introduction & Implementation
- Binary Heap

    - Insertion
    - Heapify and Extract
    - Decrease Key, Delete and Build Heap
- Heap Sort
- Priority Queue in C++
- PriorityQueue in Java
- Problems:
    - Sort K-Sorted Array
    - Buy Maximum Items with Given Sum
    - K Largest Elements
    - Merge K Sorted Arrays
    - Median of a Stream
- **Graph Algorithms**
- Introduction to Graph
- Graph Representation
    - Adjacency Matrix
    - Adjacency List in CPP and Java
    - Adjacency Matrix VS List
- Breadth-First Search
    - Applications
- Depth First Search
    - Applications

Week 7

- **Graph Algorithms**
- **Problems:**
    - Shortest Path in an Unweighted Graph
    - Number of Islands
    - Snake-Ladder
    - Detecting Cycle
        - In the Undirected Graph
        - In the Directed Graph
    - Topological Sorting
        - Kahn's BFS Based Algorithm
        - DFS Based Algorithm
- Shortest Path in Directed Acyclic Graph
- **Greedy Algorithms**
- Introduction
- Activity Selection Problem
- Fractional Knapsack
- Job Sequencing Problem

Week 8

- **Dynamic Programming**
- Introduction
- Dynamic Programming
    - Memoization
    - Tabulation
- **Problems:**
    - Longest Common Subsequence
    - Coin Change Count Combinations
    - Edit Distance Problem
        - Naive Approach
        - DP Approach
    - Longest Increasing Subsequence Problem
        - Naive Approach
        - Efficient Approach
    - Maximum Cuts
    - Minimum coins to make a value
    - Minimum Jumps to reach at the end
    - 0-1 knapsack problem
        - Naive Approach
        - Efficient Approach
    - Optimal Strategy for a Game
    - Variation of Longest Common Subsequence
    - Variation of Longest Increasing Subsequence
    - Egg Dropping Problem