# **🧭 Full DSA Roadmap for Students**

## **📘 Module 0: Introduction to DSA**

**Topics:**

* What is Data Structures and Algorithms?
* Why is DSA important?  
  + For problem-solving, efficiency, job interviews, competitive coding, logical thinking
* Real-life examples  
  + Maps, Auto-complete, Uber ETA, etc.
* How to learn DSA effectively  
  + Practice → Understand → Optimize → Repeat
* Tools to use  
  + VS Code
  + JS Console
  + LeetCode
  + SkillCaptain

## **✅ Module 0.5: Basic Math for DSA**

**Topics:**

* Number Properties & Divisibility  
  + Prime numbers
  + GCD & LCM (Euclidean Algorithm)
  + Divisibility rules
* Modular Arithmetic  
  + Basics of % operator
  + Properties: mod with addition and multiplication
  + Fast Exponentiation (Binary exponentiation)
* Bit Manipulation Basics  
  + Binary representation
  + AND, OR, XOR, NOT
  + Use-cases: check even/odd, count bits, XOR swap
* Basic Math Formulae  
  + Sum of N natural numbers
  + Sum of squares, cubes (briefly)

## **📘 Module 1: Big-O Notation & Core Fundamentals**

**Topics:**

* Time and Space Complexity
* Big-O Notations:  
  + O(1), O(n), O(log n), O(n log n), O(n²)
* Case analysis  
  + Best, worst, average cases
* Analyze JS code snippets

## **📘 Module 2: Arrays & Problem-Solving Patterns**

**Topics:**

* Array Operations  
  + Traversal, Insert, Delete
  + Built-in methods (push, pop, splice, slice, etc.)
* Two Pointer Technique  
  + Reverse array
  + Pair sum
  + Move zeros
* Sliding Window  
  + Max sum subarray
  + Longest substring without repeat
* Common Interview Questions  
  + Duplicates
  + Missing number
  + Intersection

## **📘 Module 3: Linked Lists**

**Topics:**

* Singly Linked List  
  + Insert at head/tail
  + Delete
  + Search
* Doubly Linked List
* Problems:  
  + Reverse a list (iterative & recursive)
  + Merge two sorted lists
  + Detect cycle (Floyd’s Tortoise and Hare)

## **📘 Module 4: Stacks and Queues**

**Topics:**

* Stack  
  + LIFO, Use cases (undo, brackets)
  + Implementation with array
  + Problems: Valid Parentheses, Next Greater Element
* Queue  
  + FIFO, Use cases (scheduling, printers)
  + Circular Queue concept
  + Implementation with array
* Practice Problems  
  + Min Stack
  + Queue using Stacks
  + Stack using Queues

## **📘 Module 5: Hashing & Sets**

**Topics:**

* Hash Tables in JS  
  + Map, Object
* Set for uniqueness
* Frequency Counter Pattern
* Common Problems  
  + Anagrams
  + First non-repeating character
  + Longest consecutive sequence
  + Count frequency
  + Group anagrams

## **📘 Module 6: Algorithms**

**Recursion**

* Stack frames, base cases, tail recursion
* Factorial, Fibonacci, Array sum
* When to use recursion vs iteration

**Backtracking**

* Subsets, Permutations
* N-Queens
* Rat in a Maze
* Sudoku Solver
* Binary Strings without consecutive 1s

**Searching Algorithms**

* Linear Search
* Binary Search (sorted input required)

**Sorting Algorithms**

* Bubble, Selection, Insertion (intro sorts)
* Merge Sort (Divide & Conquer)
* Time/space comparisons

**Kadane’s Algorithm**

* For max subarray sum

## **📘 Module 7: Trees**

**Topics:**

* Basics  
  + Node, Root, Parent, Child, Leaf, Height, Depth
  + Binary Tree vs BST
* Tree Traversal Techniques  
  + Inorder (LNR), Preorder (NLR), Postorder (LRN)
  + Level Order Traversal (using Queue)
  + Recursive & Iterative implementation
* Common Problems  
  + Max depth
  + Check balanced tree
  + Lowest Common Ancestor (LCA)
  + Path sum
  + Same tree check

## **📘 Module 8: Graphs**

**Topics:**

* Introduction  
  + Graph Terminology: nodes, edges, directed/undirected, weighted
  + Adjacency List vs Matrix representation
* Graph Traversal Techniques  
  + BFS (Breadth First Search)
  + DFS (Depth First Search) – recursive & iterative
* Applications & Problems  
  + Detect Cycle (directed & undirected)
  + Count connected components
  + Shortest Path (BFS-based for unweighted graphs)
  + Word Ladder
  + Number of Islands
  + Clone Graph

## **📘 Module 9: Greedy Algorithms**

**Topics:**

* Activity Selection
* Minimum Coins
* Jump Game
* Interval Scheduling
* When Greedy fails vs when it works

## **📘 Module 10: Dynamic Programming (DP)**

**Topics:**

* Introduction  
  + Memoization vs Tabulation
  + Overlapping subproblems & optimal substructure
* Classic Problems  
  + Fibonacci (recursion → memoization → tabulation)
  + Climbing Stairs
  + 0/1 Knapsack
  + Subset Sum
  + Longest Common Subsequence
  + Minimum Path Sum