**📌 Study Plan for Data Structures and Algorithms (DSA)**

**1️⃣ Hash Tables & Hashing**

Start with hash tables and hashing techniques because they form the foundation for efficient searching and indexing.

**Topics to Cover:**

✅ Basics of Hash Tables  
✅ Types of Hash Functions  
✅ Applications of Hash Tables  
✅ Hash Collision and Resolution Methods  
✅ Chaining (Separate Chaining)  
✅ Open Addressing  
✅ Linear Probing vs. Quadratic Probing  
✅ Double Hashing  
✅ Load Factor  
✅ Rehashing  
✅ Hash Table vs. HashSet  
✅ Hash Table Implementation with Collision Handling  
✅ Find Occurrences of Characters using Hash Table  
✅ Use Hash Tables to Find Two Numbers in an Array that Sum to a Target  
✅ Implement a Hash Table to Count Character Frequency in a String  
✅ Find the First Non-Repeating Character using Hash Table  
✅ Hash Table Time Complexity  
✅ Hashing vs. Encryption  
✅ Popular Hashing Algorithms (SHA1, MD5, CRC32)

🔹 **Practice Problems:**  
✅ Leetcode Two Sum  
✅ Find First Non-Repeating Character in a String  
✅ Count Occurrences of Each Character in a String  
✅ Implement a Hash Table

**2️⃣ Stacks**

Stacks are important for problems involving Last-In-First-Out (LIFO) order, recursion, and expression evaluation.

**Topics to Cover:**

✅ Basics of Stacks  
✅ Stack Using Linked List  
✅ Applications of Stack  
✅ Stack Underflow & Overflow  
✅ Parenthesis Checking using a Stack  
✅ Stack Used in Undo-Redo Operations  
✅ Convert Stack into a Queue  
✅ Implement a Stack with push, pop, and getMax in O(1)  
✅ Remove Middle Element from a Stack  
✅ Sort a Stack  
✅ Palindrome Checking using Stack  
✅ Reverse a String using Stack  
✅ Valid Parentheses Problem  
✅ Stack Implementation using Queue  
✅ Stack that Rejects Duplicate Values  
✅ Purpose of Stack Pointer

🔹 **Practice Problems:**  
✅ Valid Parentheses (Leetcode 20)  
✅ Reverse a String using Stack  
✅ Remove Middle Element from Stack  
✅ Convert Stack into Queue

**3️⃣ Queues**

Queues are useful for scheduling, process management, and BFS traversal.

**Topics to Cover:**

✅ Basics of Queues  
✅ Applications of Queue  
✅ Types of Queues  
✅ Circular Queue Implementation  
✅ Double-Ended Queue (Deque)  
✅ Bounded Queue  
✅ Priority Queue  
✅ Implement Double-Ended Queue using Linked List  
✅ Enqueue, Dequeue, Display Operations

🔹 **Practice Problems:**  
✅ Reverse a Queue  
✅ Convert Stack into Queue

**4️⃣ Sorting Algorithms**

Sorting is fundamental for searching and organizing data.

**Topics to Cover:**

✅ Time Complexity of Sorting Algorithms  
✅ Insertion Sort  
✅ Selection Sort  
✅ Merge Sort  
✅ Quick Sort  
✅ Bubble Sort  
✅ Stable Sorting Algorithms  
✅ In-Place Sorting  
✅ Quick Sort vs. Merge Sort  
✅ Worst-Case Complexity of Quick Sort  
✅ Disadvantage of Quick Sort over Merge Sort  
✅ Why Merge Sort is Preferred for Linked Lists  
✅ Merge Sort on an Array of Strings  
✅ Importance of Pivot in Quick Sort  
✅ How Pivot Selection Affects Quick Sort Performance  
✅ Choosing the Appropriate Sorting Algorithm  
✅ Sorting an Array of Objects based on a Property  
✅ Check if an Array is Sorted in O(n) Time

🔹 **Practice Problems:**  
✅ Implement Merge Sort  
✅ Implement Quick Sort  
✅ Check if an Array is Sorted in O(n)

**5️⃣ Linked Lists**

Linked lists are fundamental for understanding dynamic memory and efficient insertions/deletions.

**Topics to Cover:**

✅ Basics of Linked Lists  
✅ Merge Two Sorted Linked Lists  
✅ Remove Middle Element from Linked List  
✅ Circular Queue using Linked List  
✅ Implement Double-Ended Queue using Linked List

🔹 **Practice Problems:**  
✅ Merge Two Sorted Linked Lists  
✅ Remove Middle Element from Linked List

**6️⃣ Arrays**

Arrays are the most basic data structure and are used in almost every problem.

**Topics to Cover:**

✅ Reverse an Array  
✅ Remove Duplicates from an Array in O(n)  
✅ Find the First Missing Number in an Array  
✅ Find the Subarray with the Maximum Sum (Kadane’s Algorithm)  
✅ Merge Two Sorted Arrays in O(n) Time

🔹 **Practice Problems:**  
✅ Reverse an Array  
✅ Remove Duplicates from an Array  
✅ Find First Missing Number

**7️⃣ Recursion**

Recursion is used in tree traversal, divide-and-conquer algorithms, and backtracking.

**Topics to Cover:**

✅ Basics of Recursion  
✅ Tail vs. Head Recursion  
✅ Binary Recursion

🔹 **Practice Problems:**  
✅ Implement Factorial using Recursion  
✅ Solve Fibonacci using Recursion  
✅ Reverse a String using Recursion

**8️⃣ General DSA Topics**

Before diving into advanced problems, understanding the core concepts of data structures is necessary.

**Topics to Cover:**

✅ Time and Space Complexity Analysis  
✅ Linear vs. Non-Linear Data Structures  
✅ Divide and Conquer Approach  
✅ Memory Pool  
✅ Virtual Memory  
✅ Purpose of Stack Pointer

🔹 **Practice Problems:**  
✅ Analyze Time Complexity of Algorithms  
✅ Implement Divide and Conquer Approach

**9️⃣ Practice Problems (Leetcode / Blind 75)**

Once you have covered all the topics, start solving **Blind 75** problems to strengthen problem-solving skills.  
✅ Two Sum  
✅ Valid Anagram  
✅ Reverse a Queue  
✅ Convert a Stack into a Queue  
✅ Find the First Non-Repeating Character in a String  
✅ Find the Occurrence of Each Character in a String  
✅ Valid Parentheses (Leetcode 20)  
✅ Get the Count of Invalid Pairs in a Parenthesis String  
✅ Implement a Hash Table Workout with Practical Problems