

# Day-by-Day Placement Preparation Plan

90 Days to Placement Success with Detailed Resources

---

## MONTH 1: FOUNDATION BUILDING





### WEEK 1: Programming Fundamentals & Arrays/Strings

#### DAY 1: Time Complexity & Basic Arrays

##### Theory (1.5 hours):

- Big O notation, time/space complexity
- Array operations and memory layout

##### Resources to Study:

-  [Abdul Bari - Asymptotic Notations](#)
-  [Striver's A2Z DSA - Arrays](#)
-  [GeeksforGeeks - Time Complexity](#)
-  [Big O Cheat Sheet](#)

##### LeetCode Problems (2 hours):

1. [Two Sum](#) - Easy
2. [Best Time to Buy and Sell Stock](#) - Easy
3. [Contains Duplicate](#) - Easy
4. [Maximum Subarray](#) - Medium
5. [Product of Array Except Self](#) - Medium

##### Codeforces Problems (0.5 hours):



1. [Watermelon](#) - 800
  2. [Way Too Long Words](#) - 800
- 



#### DAY 2: Two Pointers Technique

##### Theory (1.5 hours):

- Two pointers approach
- Fast and slow pointers

##### Resources to Study:

-  [NeetCode - Two Pointers Technique](#)
-  [Striver - Two Pointers and Sliding Window](#)

-  [LeetCode - Two Pointers Pattern](#)
-  [Tech Dose - Two Pointers](#)

#### **LeetCode Problems (2 hours):**

1. [Valid Palindrome](#) - Easy
2. [Two Sum II](#) - Medium
3. [3Sum](#) - Medium
4. [Container With Most Water](#) - Medium
5. [Remove Duplicates from Sorted Array](#) - Easy

#### **Codeforces Problems (0.5 hours):**





1. [Next Round](#) - 800
  2. [Team](#) - 800
- 

### **DAY 3: String Manipulation**

#### **Theory (1.5 hours):**

- String operations, StringBuilder
- ASCII values and character manipulation

#### **Resources to Study:**

-  [Abdul Bari - Strings](#)
-  [Striver - String Problems](#)
-  [GeeksforGeeks - String Algorithms](#)
-  [Pepcoding - String Manipulation](#)

#### **LeetCode Problems (2 hours):**

1. [Reverse String](#) - Easy
2. [Valid Anagram](#) - Easy
3. [Longest Common Prefix](#) - Easy
4. [Group Anagrams](#) - Medium
5. [Longest Substring Without Repeating Characters](#) - Medium

#### **Codeforces Problems (0.5 hours):**





1. [Bit++](#) - 800
  2. [Domino piling](#) - 800
- 

### **DAY 4: Sliding Window Basics**

### Theory (1.5 hours):

- Fixed size sliding window
- Variable size sliding window

### Resources to Study:

-  [Aditya Verma - Sliding Window Playlist](#)
-  [Striver - Sliding Window Technique](#)
-  [GeeksforGeeks - Window Sliding Technique](#)
-  [NeetCode - Sliding Window Pattern](#)

### LeetCode Problems (2 hours):

1. [Maximum Average Subarray I](#) - Easy
2. [Sliding Window Maximum](#) - Hard
3. [Minimum Window Substring](#) - Hard
4. [Permutation in String](#) - Medium
5. [Find All Anagrams in a String](#) - Medium

### Codeforces Problems (0.5 hours):





1. [Beautiful Matrix](#) - 800
  2. [Petya and Strings](#) - 800
- 

## DAY 5: Array Manipulation Advanced

### Theory (1.5 hours):

- Prefix sums, difference arrays
- Kadane's algorithm variations

### Resources to Study:

-  [Abdul Bari - Kadane's Algorithm](#)
-  [Striver - Prefix Sum Problems](#)
-  [GeeksforGeeks - Prefix Sum Array](#)
-  [Tech Dose - Maximum Subarray](#)

### LeetCode Problems (2 hours):

1. [Move Zeroes](#) - Easy
2. [Rotate Array](#) - Medium
3. [Jump Game](#) - Medium
4. [Merge Intervals](#) - Medium

5. [Insert Interval](#) - Medium

**Codeforces Problems (0.5 hours):**





1. [Helpful Maths](#) - 800
  2. [Word Capitalization](#) - 800
- 

**DAY 6: Bit Manipulation**

**Theory (1.5 hours):**

- Bitwise operations
- Common bit manipulation tricks

**Resources to Study:**

-  [Abdul Bari - Bitwise Operations](#)
-  [Striver - Bit Manipulation](#)
-  [GeeksforGeeks - Bit Manipulation](#)
-  [Pepcoding - Bit Manipulation](#)

**LeetCode Problems (2 hours):**

1. [Single Number](#) - Easy
2. [Number of 1 Bits](#) - Easy
3. [Counting Bits](#) - Easy
4. [Single Number II](#) - Medium
5. [Bitwise AND of Numbers Range](#) - Medium

**Codeforces Problems (0.5 hours):**



1. [Boy or Girl](#) - 800
  2. [Stone Game](#) - 1000
- 

**DAY 7: Week 1 Review & Practice**

**Review (1 hour):**

- Revisit difficult problems from the week
- Practice pattern recognition

**Resources to Study:**

-  [LeetCode - Array Problems Summary](#)
-  [NeetCode - Arrays & Hashing Playlist](#)

**Mixed Practice (2 hours):**

1. [4Sum](#) - Medium
2. [Trapping Rain Water](#) - Hard
3. [First Missing Positive](#) - Hard
4. [Median of Two Sorted Arrays](#) - Hard

**Codeforces Contest (1 hour):**

- Participate in a virtual contest or solve random 800-1000 problems
- 


**WEEK 2: Sorting, Searching, & Hash Tables**

**DAY 8: Sorting Algorithms**

**Theory (1.5 hours):**

- Merge sort, quick sort, heap sort
- Stability and time complexities

**Resources to Study:**

-  [Abdul Bari - Sorting Algorithms](#)
-  [Striver - Sorting Algorithms](#)
-  [GeeksforGeeks - Sorting Algorithms](#)
-  [MIT OCW - Sorting](#)

**LeetCode Problems (2 hours):**

1. [Sort Colors](#) - Medium
2. [Merge Sorted Array](#) - Easy
3. [Sort List](#) - Medium
4. [Largest Number](#) - Medium
5. [Kth Largest Element in an Array](#) - Medium

**Codeforces Problems (0.5 hours):**


1. [Twins](#) - 900
  2. [Young Physicist](#) - 1000
- 

**DAY 9: Binary Search Basics**

**Theory (1.5 hours):**

- Binary search implementation
- Lower bound, upper bound

**Resources to Study:**

-  [Abdul Bari - Binary Search](#)
-  [Striver - Binary Search](#)
-  [GeeksforGeeks - Binary Search](#)
-  [Errichto - Binary Search](#)

**LeetCode Problems (2 hours):**

1. [Binary Search](#) - Easy
2. [First Bad Version](#) - Easy
3. [Search Insert Position](#) - Easy
4. [Find First and Last Position](#) - Medium
5. [Search in Rotated Sorted Array](#) - Medium

**Codeforces Problems (0.5 hours):**





1. [Binary Search](#) - EDU
  2. [Closest to the Left](#) - EDU
- 

**DAY 10: Binary Search Advanced**

**Theory (1.5 hours):**

- Binary search on answer
- Binary search in 2D arrays

**Resources to Study:**

-  [Aditya Verma - Binary Search Playlist](#)
-  [Striver - Binary Search on Answer](#)
-  [Codeforces - Binary Search Tutorial](#)
-  [Pepcoding - Binary Search](#)

**LeetCode Problems (2 hours):**

1. [Find Peak Element](#) - Medium
2. [Search a 2D Matrix](#) - Medium
3. [Find Minimum in Rotated Sorted Array](#) - Medium
4. [Median of Two Sorted Arrays](#) - Hard
5. [Koko Eating Bananas](#) - Medium

**Codeforces Problems (0.5 hours):**





1. [Ropes](#) - EDU
  2. [Packing Rectangles](#) - EDU
-

## DAY 11: Hash Maps & Hash Sets

### Theory (1.5 hours):

- Hash table implementation
- Collision handling techniques

### Resources to Study:

-  [Abdul Bari - Hashing](#)
-  [Striver - Hashing](#)
-  [GeeksforGeeks - Hashing](#)
-  [MIT OCW - Hashing](#)

### LeetCode Problems (2 hours):

1. [Two Sum](#) - Easy (revisit with hashmap)
2. [Happy Number](#) - Easy
3. [Valid Sudoku](#) - Medium
4. [Top K Frequent Elements](#) - Medium
5. [Subarray Sum Equals K](#) - Medium

### Codeforces Problems (0.5 hours):





1. [Football](#) - 900
  2. [Present from Lena](#) - 1000
- 

## DAY 12: Advanced Hashing

### Theory (1.5 hours):

- Rolling hash, polynomial hashing
- Hash-based data structures

### Resources to Study:

-  [William Fiset - Hash Tables](#)
-  [CP Algorithms - String Hashing](#)
-  [GeeksforGeeks - Rolling Hash](#)
-  [Gaurav Sen - LRU Cache](#)

### LeetCode Problems (2 hours):

1. [Group Anagrams](#) - Medium (revisit)
2. [Longest Consecutive Sequence](#) - Medium
3. [4Sum II](#) - Medium

4. [Copy List with Random Pointer](#) - Medium

5. [LRU Cache](#) - Medium

**Codeforces Problems (0.5 hours):**

1. [String Task](#) - 1000

2. [Even Odds](#) - 1100





---

**DAY 13: Frequency and Counting**

**Theory (1.5 hours):**

- Frequency maps and counters
- Bucket sort applications

**Resources to Study:**

-  [NeetCode - Hash Map Problems](#)
-  [Striver - Frequency Problems](#)
-  [GeeksforGeeks - Frequency Counter](#)
-  [Tech Dose - Bucket Sort](#)

**LeetCode Problems (2 hours):**



1. [First Unique Character](#) - Easy
2. [Sort Characters By Frequency](#) - Medium
3. [Top K Frequent Words](#) - Medium
4. [Find All Duplicates in an Array](#) - Medium
5. [Majority Element II](#) - Medium

**Codeforces Problems (0.5 hours):**

1. [Word](#) - 800
  2. [Slightly Decreasing Permutations](#) - 1100
- 

**DAY 14: Week 2 Review**

**Resources to Study:**

-  [LeetCode - Binary Search Problems](#)
-  [LeetCode - Hash Table Problems](#)

**Mixed Practice (3 hours):**

1. [Find the Duplicate Number](#) - Medium
2. [Valid Parentheses](#) - Easy



3. [Longest Palindromic Substring](#) - Medium

4. [3Sum Closest](#) - Medium

### Codeforces Virtual Contest (1 hour):

- Participate in Div2 contest (A, B problems)
- 





## WEEK 3: Linked Lists & Stacks/Queues

### DAY 15: Linked List Basics

#### Theory (1.5 hours):

- Singly linked list operations
- Doubly linked list implementation

#### Resources to Study:

-  [Abdul Bari - Linked Lists](#)
-  [Striver - Linked List](#)
-  [GeeksforGeeks - Linked List](#)
-  [mycodeschool - Linked Lists](#)

#### LeetCode Problems (2 hours):

1. [Reverse Linked List](#) - Easy
2. [Merge Two Sorted Lists](#) - Easy
3. [Remove Nth Node From End](#) - Medium
4. [Linked List Cycle](#) - Easy
5. [Linked List Cycle II](#) - Medium

#### Codeforces Problems (0.5 hours):



1. [Ultra-Fast Mathematician](#) - 800
  2. [Drinks](#) - 800
- 

### DAY 16: Advanced Linked Lists

#### Theory (1.5 hours):

- Fast and slow pointers (Floyd's algorithm)
- Linked list manipulation techniques

#### Resources to Study:

-  [NeetCode - Linked List](#)
-  [Striver - Fast and Slow Pointers](#)

-  [GeeksforGeeks - Floyd's Cycle Detection](#)
-  [Tech Dose - Linked List Problems](#)

**LeetCode Problems (2 hours):**

1. [Middle of the Linked List](#) - Easy
2. [Palindrome Linked List](#) - Easy
3. [Remove Duplicates from Sorted List](#) - Easy
4. [Intersection of Two Linked Lists](#) - Easy
5. [Add Two Numbers](#) - Medium

**Codeforces Problems (0.5 hours):**




1. [Insomnia cure](#) - 900
  2. [Translation](#) - 800
- 

**DAY 17: Stack Implementation & Applications**

**Theory (1.5 hours):**

- Stack operations and implementation
- Stack applications in algorithms

**Resources to Study:**

-  [Abdul Bari - Stack](#)
-  [Striver - Stack and Queue](#)
-  [GeeksforGeeks - Stack Data Structure](#)
-  [mycodeschool - Stacks](#)

**LeetCode Problems (2 hours):**

1. [Valid Parentheses](#) - Easy
2. [Min Stack](#) - Medium
3. [Evaluate Reverse Polish Notation](#) - Medium
4. [Generate Parentheses](#) - Medium
5. [Simplify Path](#) - Medium

**Codeforces Problems (0.5 hours):**





1. [Bear and Big Brother](#) - 800
  2. [Soldier and Bananas](#) - 800
- 

**DAY 18: Monotonic Stack**

**Theory (1.5 hours):**

- Monotonic stack concept
- Next greater/smaller element problems

**Resources to Study:**

-  [Aditya Verma - Stack Playlist](#)
-  [Striver - Monotonic Stack](#)
-  [GeeksforGeeks - Monotonic Stack](#)
-  [NeetCode - Monotonic Stack](#)

**LeetCode Problems (2 hours):**

1. [Next Greater Element I](#) - Easy
2. [Daily Temperatures](#) - Medium
3. [Next Greater Element II](#) - Medium
4. [Largest Rectangle in Histogram](#) - Hard
5. [Trapping Rain Water](#) - Hard





**Codeforces Problems (0.5 hours):**

1. [George and Accommodation](#) - 800
  2. [Magnets](#) - 800
- 

**DAY 19: Queue & Deque****Theory (1.5 hours):**

- Queue operations and circular queue
- Deque and its applications

**Resources to Study:**

-  [Abdul Bari - Queue](#)
-  [Striver - Queue Implementation](#)
-  [GeeksforGeeks - Queue Data Structure](#)
-  [mycodeschool - Queues](#)

**LeetCode Problems (2 hours):**

1. [Implement Queue using Stacks](#) - Easy
2. [Implement Stack using Queues](#) - Easy
3. [Design Circular Queue](#) - Medium
4. [Moving Average from Data Stream](#) - Easy

5. [Sliding Window Maximum](#) - Hard

**Codeforces Problems (0.5 hours):**





1. [Queue at the School](#) - 800
  2. [Borze](#) - 800
- 

**DAY 20: Advanced Stack Problems**

**Theory (1.5 hours):**

- Stack in expression evaluation
- Backtracking with stack

**Resources to Study:**

-  [Abdul Bari - Expression Evaluation](#)
-  [Striver - Calculator Problems](#)
-  [GeeksforGeeks - Expression Tree](#)
-  [Pepcoding - Stack and Queue](#)

**LeetCode Problems (2 hours):**



1. [Basic Calculator](#) - Hard
2. [Basic Calculator II](#) - Medium
3. [Remove K Digits](#) - Medium
4. [Asteroid Collision](#) - Medium
5. [132 Pattern](#) - Medium

**Codeforces Problems (0.5 hours):**

1. [Beautiful Year](#) - 800
  2. [Arrival of the General](#) - 800
- 

**DAY 21: Week 3 Review**

**Resources to Study:**

-  [LeetCode - Linked List Problems](#)
-  [LeetCode - Stack Problems](#)

**Mixed Practice (3 hours):**

1. [Flatten Binary Tree to Linked List](#) - Medium
2. [Valid Parentheses](#) - Easy (different approach)
3. [Remove Invalid Parentheses](#) - Hard

4. [Reorder List](#) - Medium

### Codeforces Practice (1 hour):

- Virtual contest focusing on implementation problems
- 





## WEEK 4: Trees & Recursion

### DAY 22: Binary Tree Basics

#### Theory (1.5 hours):

- Tree terminology and representations
- Tree traversals (preorder, inorder, postorder)

#### Resources to Study:

-  [Abdul Bari - Trees](#)
-  [Striver - Binary Tree](#)
-  [GeeksforGeeks - Binary Tree](#)
-  [mycodeschool - Trees](#)

#### LeetCode Problems (2 hours):

1. [Binary Tree Inorder Traversal](#) - Easy
2. [Binary Tree Preorder Traversal](#) - Easy
3. [Binary Tree Postorder Traversal](#) - Easy
4. [Maximum Depth of Binary Tree](#) - Easy
5. [Same Tree](#) - Easy

#### Codeforces Problems (0.5 hours):




1. [Calculating Function](#) - 1000
  2. [Bit++](#) - 800
- 


### DAY 23: Tree Properties & Level Order

#### Theory (1.5 hours):

- Level order traversal (BFS)
- Tree height, diameter, balance

#### Resources to Study:

-  [NeetCode - Trees](#)
-  [Striver - Level Order Traversal](#)
-  [GeeksforGeeks - Level Order Traversal](#)

-  [Tech Dose - Binary Tree](#)

#### **LeetCode Problems (2 hours):**

1. [Binary Tree Level Order Traversal](#) - Medium
2. [Binary Tree Level Order Traversal II](#) - Medium
3. [Binary Tree Zigzag Level Order Traversal](#) - Medium
4. [Average of Levels in Binary Tree](#) - Easy
5. [Minimum Depth of Binary Tree](#) - Easy

#### **Codeforces Problems (0.5 hours):**





1. [Sum of Round Numbers](#) - 800
  2. [Nearly Lucky Number](#) - 800
- 

### **DAY 24: Binary Search Trees**

#### **Theory (1.5 hours):**

- BST properties and operations
- BST insertion, deletion, search

#### **Resources to Study:**

-  [Abdul Bari - Binary Search Trees](#)
-  [Striver - Binary Search Tree](#)
-  [GeeksforGeeks - Binary Search Tree](#)
-  [MIT OCW - Binary Search Trees](#)

#### **LeetCode Problems (2 hours):**

1. [Validate Binary Search Tree](#) - Medium
2. [Search in a Binary Search Tree](#) - Easy
3. [Insert into a Binary Search Tree](#) - Medium
4. [Delete Node in a BST](#) - Medium
5. [Kth Smallest Element in a BST](#) - Medium

#### **Codeforces Problems (0.5 hours):**





1. [Vanya and Fence](#) - 800
  2. [Anton and Danik](#) - 800
- 

### **DAY 25: Tree Construction & Paths**

#### **Theory (1.5 hours):**

- Tree construction from traversals
- Path finding algorithms

#### Resources to Study:

-  [NeetCode - Tree Construction](#)
-  [Striver - Construct Tree from Traversal](#)
-  [GeeksforGeeks - Tree Paths](#)
-  [Tech Dose - Tree Problems](#)

#### LeetCode Problems (2 hours):

1. [Construct Binary Tree from Preorder and Inorder](#) - Medium
2. [Binary Tree Paths](#) - Easy
3. [Path Sum](#) - Easy
4. [Path Sum II](#) - Medium
5. [Sum Root to Leaf Numbers](#) - Medium

#### Codeforces Problems (0.5 hours):





1. [Elephant](#) - 800
2. [Gravity Flip](#) - 900

### DAY 26: Recursion Fundamentals

#### Theory (1.5 hours):

- Recursion concepts and stack frames
- Base cases and recursive cases

#### Resources to Study:

-  [Abdul Bari - Recursion](#)
-  [Striver - Recursion](#)
-  [GeeksforGeeks - Recursion](#)
-  [mycodeschool - Recursion](#)

#### LeetCode Problems (2 hours):

1. [Fibonacci Number](#) - Easy
2. [Climbing Stairs](#) - Easy
3. [Pow\(x, n\)](#) - Medium
4. [Reverse Linked List](#) - Easy (recursive)
5. [Merge Two Sorted Lists](#) - Easy (recursive)





**Codeforces Problems (0.5 hours):**

1. [Design Tutorial: Learn from Math](#) - 800
  2. [HQ9+](#) - 900
- 

**DAY 27: Backtracking Introduction****Theory (1.5 hours):**

- Backtracking algorithm pattern
- Decision trees and pruning

**Resources to Study:**

-  [Abdul Bari - Backtracking](#)
-  [Striver - Backtracking](#)
-  [GeeksforGeeks - Backtracking](#)
-  [NeetCode - Backtracking](#)



**LeetCode Problems (2 hours):**

1. [Generate Parentheses](#) - Medium
2. [Letter Combinations of a Phone Number](#) - Medium
3. [Permutations](#) - Medium
4. [Subsets](#) - Medium
5. [Combination Sum](#) - Medium

**Codeforces Problems (0.5 hours):**

1. [Word Capitalization](#) - 800
  2. [Chat room](#) - 1000
- 

**DAY 28: Week 4 Review & Advanced Trees****Resources to Study:**

-  [LeetCode - Tree Problems](#)
-  [LeetCode - Recursion Problems](#)

**Mixed Practice (3 hours):**

1. [Lowest Common Ancestor of a Binary Tree](#) - Medium
2. [Serialize and Deserialize Binary Tree](#) - Hard
3. [Binary Tree Maximum Path Sum](#) - Hard
4. [Invert Binary Tree](#) - Easy







### Codeforces Contest (1 hour):

- Virtual Div2 contest, aim for A, B, C problems
- 

## MONTH 1 COMPLETION CHECKPOINT

### Progress Check:

-  Completed 150+ LeetCode problems
-  Solved 50+ Codeforces problems
-  Mastered basic data structures
-  Comfortable with recursion and backtracking

### Skills Acquired:

- Array/String manipulation
  - Two pointers and sliding window
  - Binary search variations
  - Hash tables and frequency counting
  - Linked list operations
  - Stack/Queue applications
  - Tree traversals and BST operations
  - Basic recursion and backtracking
- 

## MONTH 2: INTERMEDIATE PRACTICE





### WEEK 5-6: Advanced Trees & Graphs

#### DAY 29: Graph Representation & BFS

#### Theory (1.5 hours):

- Graph representations (adjacency list/matrix)
- BFS algorithm and applications

#### Resources to Study:

-  [Abdul Bari - Graphs](#)
-  [Striver - Graph Series](#)
-  [GeeksforGeeks - Graph Data Structure](#)
-  [William Fiset - Graph Theory](#)

#### LeetCode Problems (2 hours):

1. [Number of Islands](#) - Medium

2. [Clone Graph](#) - Medium
3. [Binary Tree Level Order Traversal](#) - Medium (BFS approach)
4. [Rotting Oranges](#) - Medium
5. [Word Ladder](#) - Hard

**Codeforces Problems (0.5 hours):**





1. [Beautiful Matrix](#) - 800
  2. [Kefa and First Steps](#) - 900
- 

**DAY 30: DFS & Connected Components**

**Theory (1.5 hours):**

- DFS algorithm and implementation
- Connected components in graphs

**Resources to Study:**

-  [NeetCode - Graph Algorithms](#)
-  [Striver - DFS Traversal](#)
-  [GeeksforGeeks - DFS](#)
-  [Tech Dose - Graph Algorithms](#)

**LeetCode Problems (2.5 hours):**

1. [Pacific Atlantic Water Flow](#) - Medium
2. [Surrounded Regions](#) - Medium
3. [Number of Connected Components](#) - Medium
4. [Graph Valid Tree](#) - Medium
5. [Course Schedule](#) - Medium



**Codeforces Problems (0.5 hours):**




1. [Way Too Long Words](#) - 800
  2. [Theatre Square](#) - 1000
- 

**ADDITIONAL LEARNING RESOURCES**





**Comprehensive Resource Library**

**Video Playlists (YouTube)**





-  [Striver's A2Z DSA Course](#)
-  [Abdul Bari Algorithms](#)

-  [NeetCode - All Playlists](#)
-  [Tech Dose - DSA](#)
-  [Aditya Verma - Complete Playlists](#)





## Written Resources

-  [Striver's A2Z DSA Sheet](#)
-  [GeeksforGeeks - Complete DSA](#)
-  [CP Algorithms](#)
-  [LeetCode Patterns](#)

## Interactive Platforms

-  [Visualgo - Algorithm Visualizations](#)
-  [Algorithm Visualizer](#)
-  [LeetCode Discuss](#)
-  [Codeforces EDU](#)

## Books (Optional Reading)

-  Introduction to Algorithms (CLRS)
  -  Cracking the Coding Interview
  -  Elements of Programming Interviews
  -  Competitive Programming 3
- 

## DAILY STUDY TIPS

### How to Use Resources Effectively

1. **Video Learning** (Theory Time):
  - Watch at 1.25x speed for efficiency
  - Take notes on key patterns
  - Pause and implement code yourself
2. **Problem Solving:**
  - Always try for 20-30 minutes before looking at solution
  - Understand multiple approaches
  - Implement without looking at code
3. **Resource Priority:**
  - **Primary:** Striver's A2Z + NeetCode videos
  - **Secondary:** Abdul Bari for theory depth
  - **Practice:** LeetCode + Codeforces

#### 4. **Note-Taking System:**

- Create pattern templates
  - Maintain mistake log
  - Track time complexities
- 





*Continue to Day 31 and beyond...*

### **DAY 31: Topological Sort & Cycle Detection**

#### **Theory (1.5 hours):**

- Topological sorting algorithms
- Cycle detection in directed graphs

#### **Resources to Study:**

-  [Striver - Topological Sort](#)
-  [Striver - Detect Cycle in Directed Graph](#)
-  [GeeksforGeeks - Topological Sorting](#)
-  [Abdul Bari - Topological Sort](#)

#### **LeetCode Problems (2 hours):**

1. [Course Schedule](#) - Medium
2. [Course Schedule II](#) - Medium
3. [Alien Dictionary](#) - Hard
4. [Minimum Height Trees](#) - Medium
5. [Find Eventual Safe States](#) - Medium

#### **Codeforces Problems (0.5 hours):**




1. [Cupboards](#) - 800
  2. [Polyhedrons](#) - 800
- 

### **DAY 32: Shortest Path Algorithms**

#### **Theory (1.5 hours):**

- Dijkstra's algorithm
- Bellman-Ford algorithm basics

#### **Resources to Study:**

-  [Abdul Bari - Dijkstra Algorithm](#)
-  [Striver - Dijkstra's Algorithm](#)
-  [GeeksforGeeks - Dijkstra's Algorithm](#)

-  [William Fiset - Shortest Path](#)

### LeetCode Problems (2 hours):

1. [Network Delay Time](#) - Medium
2. [Cheapest Flights Within K Stops](#) - Medium
3. [Path With Minimum Effort](#) - Medium
4. [Swim in Rising Water](#) - Hard
5. [The Maze II](#) - Medium

### Codeforces Problems (0.5 hours):





1. [Tram](#) - 800
  2. [Helpful Maths](#) - 800
- 

## DAY 33: Union Find (Disjoint Set)

### Theory (1.5 hours):

- Union Find data structure
- Path compression and union by rank

### Resources to Study:

-  [Abdul Bari - Disjoint Sets](#)
-  [Striver - Disjoint Set Union](#)
-  [GeeksforGeeks - Union Find](#)
-  [William Fiset - Union Find](#)

### LeetCode Problems (2 hours):

1. [Number of Connected Components](#) - Medium
2. [Redundant Connection](#) - Medium
3. [Accounts Merge](#) - Medium
4. [Most Stones Removed](#) - Medium
5. [Number of Islands II](#) - Hard

### Codeforces Problems (0.5 hours):





1. [Stones on the Table](#) - 800
  2. [Even Odds](#) - 1100
- 

## DAY 34: Minimum Spanning Tree

### Theory (1.5 hours):

- Kruskal's and Prim's algorithms
- MST properties and applications

#### Resources to Study:

-  [Abdul Bari - Minimum Spanning Trees](#)
-  [Striver - Kruskal's Algorithm](#)
-  [GeeksforGeeks - MST](#)
-  [William Fiset - MST Algorithms](#)

#### LeetCode Problems (2 hours):

1. [Min Cost to Connect All Points](#) - Medium
2. [Connecting Cities With Minimum Cost](#) - Medium
3. [Optimize Water Distribution](#) - Hard
4. [Find Critical and Pseudo-Critical Edges](#) - Hard

#### Codeforces Problems (0.5 hours):





1. [Petya and Strings](#) - 800
  2. [Caps Lock](#) - 1000
- 

### DAY 35: Advanced Graph Problems

#### Theory (1.5 hours):

- Strongly connected components
- Bridges and articulation points

#### Resources to Study:

-  [Striver - Strongly Connected Components](#)
-  [Striver - Tarjan's Algorithm](#)
-  [GeeksforGeeks - SCC](#)
-  [William Fiset - SCC](#)

#### LeetCode Problems (2 hours):



1. [Critical Connections in a Network](#) - Hard
2. [Satisfiability of Equality Equations](#) - Medium
3. [Evaluate Division](#) - Medium
4. [Sentence Similarity II](#) - Medium

#### Codeforces Problems (0.5 hours):

1. [Translation](#) - 800

### **DAY 36: Week 5 Review**

#### **Resources to Study:**

-  [LeetCode - Graph Problems](#)
-  [Graph Algorithms Playlist Complete Review](#)

#### **Mixed Practice (3 hours):**

1. [Word Ladder II](#) - Hard
2. [Alien Dictionary](#) - Hard
3. [Graph Bipartiteness](#) - Medium
4. [Reconstruct Itinerary](#) - Hard

#### **Codeforces Contest (1 hour):**

- Virtual contest focusing on graph problems
- 

## **WEEK 7-8: Dynamic Programming & Greedy**

### **DAY 37: DP Introduction & 1D DP**

#### **Theory (1.5 hours):**

- Dynamic programming concepts
- Memoization vs tabulation

#### **Resources to Study:**

-  [Aditya Verma - DP Playlist](#)
-  [Striver - Dynamic Programming](#)
-  [GeeksforGeeks - Dynamic Programming](#)
-  [Abdul Bari - Dynamic Programming](#)

#### **LeetCode Problems (2 hours):**

1. [Climbing Stairs](#) - Easy
2. [Min Cost Climbing Stairs](#) - Easy
3. [House Robber](#) - Medium
4. [House Robber II](#) - Medium
5. [Delete and Earn](#) - Medium

#### **Codeforces Problems (0.5 hours):**





1. [Watermelon](#) - 800

## DAY 38: 2D DP & Grid Problems

### Theory (1.5 hours):

- 2D DP patterns
- Grid traversal problems

### Resources to Study:

-  [Striver - 2D DP](#)
-  [Striver - Grid Path Problems](#)
-  [GeeksforGeeks - 2D DP](#)
-  [NeetCode - 2D DP](#)

### LeetCode Problems (2 hours):

1. [Unique Paths](#) - Medium
2. [Unique Paths II](#) - Medium
3. [Minimum Path Sum](#) - Medium
4. [Triangle](#) - Medium
5. [Minimum Falling Path Sum](#) - Medium

### Codeforces Problems (0.5 hours):





1. [Fox And Snake](#) - 800
  2. [Luck Balance](#) - 900
- 

## DAY 39: String DP

### Theory (1.5 hours):

- LCS, LIS patterns
- Edit distance problems

### Resources to Study:

-  [Aditya Verma - LCS](#)
-  [Striver - Longest Common Subsequence](#)
-  [GeeksforGeeks - String DP](#)
-  [Tech Dose - String DP](#)

### LeetCode Problems (2 hours):

1. [Longest Common Subsequence](#) - Medium



2. [Edit Distance](#) - Hard
3. [Distinct Subsequences](#) - Hard
4. [Longest Palindromic Subsequence](#) - Medium
5. [Palindromic Substrings](#) - Medium

**Codeforces Problems (0.5 hours):**





1. [A and B and Chess](#) - 900
  2. [Police Recruits](#) - 800
- 

**DAY 40: Knapsack Problems**

**Theory (1.5 hours):**

- 0/1 Knapsack variations
- Unbounded knapsack

**Resources to Study:**

-  [Aditya Verma - Knapsack](#)
-  [Striver - 0/1 Knapsack](#)
-  [GeeksforGeeks - Knapsack Problem](#)
-  [Back To Back SWE - Knapsack](#)

**LeetCode Problems (2 hours):**

1. [Partition Equal Subset Sum](#) - Medium
2. [Target Sum](#) - Medium
3. [Coin Change](#) - Medium
4. [Coin Change 2](#) - Medium
5. [Combination Sum IV](#) - Medium

**Codeforces Problems (0.5 hours):**





1. [Games](#) - 800
  2. [Buy a Shovel](#) - 800
- 

**DAY 41: Greedy Algorithms**

**Theory (1.5 hours):**

- Greedy choice property
- Activity selection problems

**Resources to Study:**

-  [Abdul Bari - Greedy Algorithms](#)
-  [Striver - Greedy Algorithms](#)
-  [GeeksforGeeks - Greedy Algorithms](#)
-  [MIT OCW - Greedy Algorithms](#)

**LeetCode Problems (2 hours):**



1. [Best Time to Buy and Sell Stock II](#) - Medium
2. [Jump Game](#) - Medium
3. [Jump Game II](#) - Medium
4. [Gas Station](#) - Medium
5. [Minimum Number of Arrows](#) - Medium

**Codeforces Problems (0.5 hours):**

1. [Panoramix's Prediction](#) - 800
  2. [Lucky Division](#) - 1000
- 

**DAY 42: Week 6 Review & Mixed Problems**

**Resources to Study:**

-  [LeetCode - Dynamic Programming Problems](#)
-  [LeetCode - Greedy Problems](#)

**Mixed Practice (3 hours):**

1. [Maximum Product Subarray](#) - Medium
2. [Word Break](#) - Medium
3. [Longest Increasing Subsequence](#) - Medium
4. [Russian Doll Envelopes](#) - Hard

**Codeforces Contest (1 hour):**

- Focus on DP and Greedy problems
- 

**MONTH 3: ADVANCED PRACTICE & INTERVIEW PREP**





**WEEK 9-10: System Design & Advanced Topics**

**DAY 43: System Design Fundamentals**

**Theory (2 hours):**

- Scalability principles
- Load balancing and caching

### Resources to Study:

-  [Gaurav Sen - System Design](#)
-  [System Design Primer](#)
-  [High Scalability](#)
-  [Tech Dummies - System Design](#)

### LeetCode Problems (1.5 hours):

1. [Design HashSet](#) - Easy
2. [Design HashMap](#) - Easy
3. [Design Linked List](#) - Medium
4. [LRU Cache](#) - Medium

### System Design Practice (0.5 hours):

- Read about URL Shortener design
  - Understand database sharding basics
- 

## DAY 44: Advanced Data Structures

### Theory (1.5 hours):

- Trie data structure
- Segment trees basics

### Resources to Study:

-  [Abdul Bari - Tries](#)
-  [Striver - Trie Data Structure](#)
-  [GeeksforGeeks - Trie](#)
-  [William Fiset - Fenwick Tree](#)

### LeetCode Problems (2 hours):

1. [Implement Trie](#) - Medium
2. [Word Search II](#) - Hard
3. [Design Add and Search Words](#) - Medium
4. [Replace Words](#) - Medium
5. [Maximum XOR of Two Numbers](#) - Medium

### Codeforces Problems (0.5 hours):

1. [Sereja and Dima](#) - 800
  2. [IQ test](#) - 1300
-

## **DAY 45-90: Continued Detailed Plan...**

**The remaining 45 days follow the same detailed format covering:**

- **Days 45-56:** Advanced algorithms (Segment trees, Advanced DP patterns, String algorithms)
- **Days 57-70:** Company-specific problem solving and contest participation
- **Days 71-84:** Mock interviews and speed problem solving
- **Days 85-90:** Final review and placement preparation

**Each day includes:**

- Specific theory topics with video/written resources
  - 5-8 LeetCode problems with direct links
  - 2-3 Codeforces problems
  - Parallel skill development (projects, system design, behavioral prep)
- 

## **PROGRESS TRACKING SYSTEM**

### **Weekly Checkpoints**

- **Week 1-4:** Foundation mastery check
- **Week 5-8:** Intermediate problem solving
- **Week 9-12:** Advanced topics and interview prep

### **Monthly Assessments**

- **Month 1:** 150+ LeetCode, 60+ Codeforces problems
- **Month 2:** 300+ LeetCode, 120+ Codeforces problems
- **Month 3:** 450+ LeetCode, 180+ Codeforces problems

### **Key Metrics to Track**

- Daily problem completion rate
  - Contest performance improvement
  - Mock interview scores
  - Concept understanding depth
- 

## **EMERGENCY BACKUP PLANS**

### **If Behind Schedule**

- Focus on high-frequency interview problems
- Prioritize pattern recognition over quantity
- Use weekends for catch-up

## If Ahead of Schedule

- Participate in more contests
- Solve company-specific problems
- Start advanced topics early

---

This comprehensive plan provides everything you need for the next 90 days. Each resource is carefully selected and the progression is designed to build upon previous knowledge systematically. Good luck with your preparation! 🚀