Assignment - Day 5: 1-D Dynamic Programming Problems

Dear Students,

This assignment focuses on building a strong foundation in 1-D Dynamic Programming (DP).

Assignment Problems:

1. Calculate nCr (GeeksforGeeks)

Link: https://www.geeksforgeeks.org/problems/ncr1019/1
Objective: Calculate the value of nCr (n choose r) using modular arithmetic (mod $10^9 + 7$).
Hint:
- Use the recursive relation: nCr = n-1Cr-1 + n-1Cr
- Implement with DP or memoization to avoid recomputation.
- Consider handling edge cases when r > n.

2. Tribonacci Number (LeetCode 1137)

Link: https://leetcode.com/problems/n-th-tribonacci-number/
Objective: Find the N-th Tribonacci number where T(n) = T(n-1) + T(n-2) + T(n-3) with base cases T(0)=0, T(1)=1, T(2)=1.
Hint:
- Solve recursively first, then optimize using memorization.
- Observe the similarity to Fibonacci and note how DP reduces time complexity.

3. Min Cost Climbing Stairs (LeetCode 746)

Link: https://leetcode.com/problems/min-cost-climbing-stairs/
Objective: Given an array where each element represents cost on a step, return the minimum cost to reach the top of the floor.
Hint:
- Define dp[i] as the minimum cost to reach step i.
- Recurrence: dp[i] = cost[i] + min(dp[i-1], dp[i-2])
- Try solving via both recursion + memorization.

4. Boredom (Codeforces 455A)

Link: https://codeforces.com/problemset/problem/455/A
Objective: Given an array of integers, choose numbers such that no two chosen numbers differ by 1, and maximize the sum of chosen numbers.
Hint:
- Sort or compress input into frequency map.
- Use recurrence similar to 'House Robber': dp[i] = max(dp[i-1], dp[i-2] + i * freq[i])
- Think about transitions in terms of 'choose or skip'.

Key Concepts to Revise:

Before attempting the assignment, make sure to review:

1. Memoization conversion.
2. Base cases and transition design (Reccurance) in DP.
3. Difference between recursion tree and DP array visualization.

Submission Instructions:

- Solve and submit all four problems.
- Use recursion → memoization approaches for understanding.
- Submit your solution files via the Google Form.
- Deadline: 12 November 2025 (02:00 am)

Note:

Understanding 1-D DP is the stepping stone to advanced 2-D problems — focus on how overlapping subproblems are handled and how recursive calls map to DP states.

Best regards,
Training Team