Assignment - Day 4: Advanced Backtracking

Dear Students,

This assignment focuses on advanced backtracking problems and introduces recursive problem-solving within structured search spaces. You are also expected to revise the initial concepts of Dynamic Programming (DP) introduced today in class.

Assignment Problems:

1. 1. Word Break (LeetCode 139)

Link: https://leetcode.com/problems/word-break/

2. 2. Word Break II (LeetCode 140)

Link: https://leetcode.com/problems/word-break-ii/

3. 3. N-Queens II (LeetCode 52)

Link: https://leetcode.com/problems/n-queens-ii/

4. 4. Sudoku Solver (LeetCode 37)

Link: https://leetcode.com/problems/sudoku-solver/

5. 5. Merge Sort (Understand Backtracking)

Objective: Implement Merge Sort using recursion.
Hint: Focus on how recursive splitting and merging demonstrate controlled backtracking of subarray results.

6. 6. Quick Sort (Understand Backtracking)

Objective: Implement Quick Sort recursively and analyze how partitioning uses backtracking to restore subarray order.
Hint: Emphasize understanding the recursion tree and the role of the pivot element during backtracking.

Revision Task (Dynamic Programming Introduction):

As we begin transitioning from Backtracking to Dynamic Programming, revise the following concepts from today's lecture:
1. Memoization Implementation of Fibonacci Series
2. Climbing Stairs Problem using Recursion + Memoization
3. Visualizing the DP Array:
   - Understand how recursion converts into a DP table.

- Focus on identifying overlapping subproblems and optimal substructure.
- Practice dry runs to see how the DP array evolves.

Tip: Try converting your recursive Fibonacci and Climbing Stairs solutions into memoized and tabulated forms to build strong intuition.

Submission Instructions:

- Complete all six problems before the next session.
- Practice the DP revision tasks even if no submission is required.
- Submit your code files for the backtracking problems through the Google Form.
- Deadline: 10 November 2025 (11:59 pm).

Note:

Backtracking teaches the importance of exploring all possible solutions while controlling recursion depth — this skill will directly help you in understanding Dynamic Programming.

Best regards,
Training Team