1. Generate Parentheses

Problem: Given n pairs of parentheses, generate all combinations of well-formed parentheses.
Description: You need to generate valid strings of parentheses using backtracking.
Example:

Input: `n = 3`
Output: `["((()))", "(()())", "(())()", "()(())", "()()()"]`

2. N-Queens Problem

Problem: Place n queens on an n x n chessboard so that no two queens threaten each other.
Description: This is a well-known problem that involves placing queens such that no two are in the same row, column, or diagonal.
Example:
Input: `n = 4`

Output: `[[.Q.., ..Q., Q..., ...Q], [..Q., Q..., ...Q, .Q..]]`

3. **Coin Change Problem**

Problem: Given an array of coins and a target amount, find the minimum number of coins needed to make up the target amount.
Description: This problem involves minimizing the number of coins used to reach a given amount.
Example:
Input: `coins = [1, 2, 5], amount = 11`
Output: 3 (The minimum number of coins is 3, using coins [5, 5, 1])

4. **Minimum Path Sum (Grid-based DP)**
Problem: Given a m x n grid filled with non-negative numbers, find a path from the top-left to the bottom-right corner that minimizes the sum of the numbers along the path. You can only move down or right.
Description: This dynamic programming problem uses a 2D DP array where each cell stores the minimum sum to reach that cell.

5. Longest Path in a Grid (with Obstacles)

Problem: Given a grid where some cells are blocked (e.g., obstacles), find the longest possible path from the top-left to the bottom-right corner. You can only move up, down, left, or right.
Description: This is a dynamic programming problem where you must calculate the longest path while considering obstacles.

Example:

Input: grid = [
  [0, 1, 0, 0],
  [0, 0, 1, 0],
  [0, 0, 0, 0],
  [0, 1, 0, 0]
]

Output: 5 (The longest path is 5 moves)