Snakes And Ladders LeetCode

You are given an n x n integer matrix board where the cells are labeled from 1 to n^2 in a **Boustrophedon style** starting from the bottom left of the board (i.e. board[n - 1][0]) and alternating direction each row.

Example:

The Input Snake and Ladder board:

-1 -1 -1 -1 -1 -1

-1 -1 -1 -1 -1 -1

-1 -1 -1 -1 -1 -1

-1 35 -1 -1 13 -1

-1 -1 -1 -1 -1 -1

-1 15 -1 -1 -1 -1

The Least Number of Moves required to move 36: 4

Approach 1: Function to find the least number of moves to reach the end of the snake and ladder board

- **Explanation:**
    - The **snakeAndLadders** function initializes a queue and a visited matrix to track explored cells.
    - It starts the BFS traversal from the first cell, and at each step, it explores the possible moves (1 to 6).
    - The next cell is determined based on whether it's a regular cell or a Snake/Ladder. The visited matrix prevents revisiting cells.
    - The traversal continues until the last cell is reached, and the number of moves is returned.
    - If the last cell cannot be reached, the function returns -1.
- **Time Complexity:**
    - **The time complexity is O(N^2), where N is the size of the board. In the worst case, all cells may need to be visited.**
    - **Each cell is visited at most once due to BFS traversal.**
- **Space Complexity:**

- **The space complexity is O(N^2) due to the visited matrix.**