Snakes and Ladders

This problem is essentially finding the shortest path in an unweighted graph where nodes are board positions (1 to n^2) and edges represent dice rolls (1–6 steps forward). Snakes and ladders create directed edges to other nodes.

Key Idea:

1. Convert 2D Boustrophedon board to 1D mapping:
   - Board is labeled starting from bottom-left.
   - Direction alternates per row (left→right, then right→left).
2. Use BFS (Breadth-First Search):
   - Each BFS level represents one dice roll.
   - For each position, consider next 6 moves.
   - If position has snake/ladder (`board[r][c] != -1`), move to that destination.
   - Track visited positions to prevent loops.
3. Return minimum rolls when reaching n^2.

Algorithm Steps:

1. Map board to 1D:
   - Create `posToCoord()` function to convert a 1D index to 2D coordinates based on zig-zag pattern.
2. BFS traversal:
   - Start at position 1.
   - For each dice roll (1 to 6):
     - Compute next position.
     - If `board[r][c] != -1`, move to that destination.
     - Mark visited and enqueue.
3. Return steps when reaching n^2.
   - If BFS ends without visiting n^2, return -1.

Complexity:

Time: $O(n^2)$ (all cells visited once).

BFS with 6 possible moves per cell

Space: $O(n^2)$ for visited set and queue.