Intuition:

The intuition behind the problem is to find the latest day where it is still possible to cross the grid by incrementally adding cells and checking if a path exists from the top row to the bottom row, utilizing binary search to optimize the search process.

Explanation:

Binary Search:

1. Initialize left as 0 (representing the earliest day) and right as row * col (representing the latest possible day).
2. While left is less than right - 1, do the following:
    - Calculate the mid-day as mid = (left + right) / 2.
    - Check if it is possible to cross the grid on day mid by calling the isPossible function.
        - The isPossible function checks if it is possible to cross the grid by considering the first mid cells from the given list. It uses BFS to explore the grid and checks if the bottom row can be reached.
        - If it is possible to cross, update left to mid and store the latest possible day in a variable (let's call it latestPossibleDay).
        - If it is not possible to cross, update right to mid.
3. Return latestPossibleDay as the latest day where it is still possible to cross the grid.

Breadth-First Search (BFS):
The isPossible function:

1. Create a grid with dimensions (m + 1) x (n + 1) and initialize all cells to 0.
2. Mark the cells from the given list as blocked by setting their corresponding positions in the grid to 1.
3. Create a queue to perform BFS.
4. For each cell in the first row of the grid, if it is not blocked, add it to the queue and mark it as visited.
5. While the queue is not empty, do the following:
    - Dequeue a cell from the front of the queue.
    - Get the current cell's coordinates (row and column).
    - Explore the neighboring cells (up, down, left, and right) by checking their validity and whether they are blocked or not.
        - If a neighboring cell is valid, not blocked, and has not been visited, mark it as visited and enqueue it.
        - If a neighboring cell is in the last row, return true as it means a

path to the bottom row has been found.
6. If we finish the BFS without finding a path to the bottom row, return false.