Problem 2596: Check Knight Tour Configuration

Problem Information

Difficulty: Medium
Acceptance Rate: 59.80%
Paid Only: No
Tags: Array, Depth-First Search, Breadth-First Search, Matrix, Simulation

Problem Description

There is a knight on an `n x n` chessboard. In a valid configuration, the knight starts **at the top-left cell** of the board and visits every cell on the board **exactly once**.

You are given an `n x n` integer matrix `grid` consisting of distinct integers from the range `[0, n * n - 1]` where `grid[row][col]` indicates that the cell `(row, col)` is the `grid[row][col]th` cell that the knight visited. The moves are **0-indexed**.

Return `true` _if_ `grid` _represents a valid configuration of the knight 's movements or_ `false` _otherwise_.

Note that a valid knight move consists of moving two squares vertically and one square horizontally, or two squares horizontally and one square vertically. The figure below illustrates all the possible eight moves of a knight from some cell.

Example 1:

Input: grid = [[0,11,16,5,20],[17,4,19,10,15],[12,1,8,21,6],[3,18,23,14,9],[24,13,2,7,22]]
Output: true **Explanation:** The above diagram represents the grid. It can be shown that it is a valid configuration.

Example 2:

Input: grid = [[0,3,6],[5,8,1],[2,7,4]] **Output:** false **Explanation:** The above diagram represents the grid. The 8th move of the knight is not valid considering its position after the 7th move.

Constraints:

* `n == grid.length == grid[i].length` * `3 <= n <= 7` * `0 <= grid[row][col] < n * n` * All integers in `grid` are **unique**.

Code Snippets

C++:

```
class Solution {
public:
bool checkValidGrid(vector<vector<int>>& grid) {


}
};
```

Java:

```
class Solution {
public boolean checkValidGrid(int[][] grid) {


}
}
```

Python3:

```
class Solution:
def checkValidGrid(self, grid: List[List[int]]) -> bool:
```