```
#include <vector>
#include <iostream>
using namespace std;

int maxMoves(vector<vector<int>> &grid)
{
    int m = grid.size();
    int n = grid[0].size();
    vector<vector<int>> dp(m, vector<int>(n, 0));
    for (int i = 0; i < m; i++)
    {
        dp[i][0] = 1;
    }
    for (int col = 1; col < n; col++)
    {
        for (int row = 0; row < m; row++)
        {
            for (pair<int, int> move : {make_pair(1, 1), make_pair(0, 1), make_pair(-1, 1)})
            {
                int r_offset = move.first;
                int c_offset = move.second;
                int new_row = row + r_offset;
                int new_col = col + c_offset;
                if (new_row >= 0 && new_row < m && grid[new_row][new_col] > grid[row][col])
                {
```

```
                dp[row][col] = max(dp[row][col], dp[new_row][new_col] + 1);
            }
        }
    }
    int max_moves = 0;
    for (int i = 0; i < m; i++)
    {
        max_moves = max(max_moves, dp[i][n - 1]);
    }

    return max_moves;
}

int main()
{
    vector<vector<int>> grid;
    int m = grid.size();
    int n = grid[0].size();
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            cin >> grid[i][j];
        }
```

```
    }
    int result = maxMoves(grid);
    cout << result << endl;
    return 0;
}
```