329. Longest Increasing Path in a Matrix

QuestionEditorial Solution

My Submissions

- Total Accepted: **24583**
- Total Submissions: **71231**
- Difficulty: **Hard**
- Contributors: **Admin**

Given an integer matrix, find the length of the longest increasing path.

From each cell, you can either move to four directions: left, right, up or down. You may NOT move diagonally or move outside of the boundary (i.e. wrap-around is not allowed).

Example 1:

```
nums = [
  [9,9,4],
  [6,6,8],
  [2,1,1]
]
```

Return `4`

The longest increasing path is `[1, 2, 6, 9]`.

Example 2:

```
nums = [
  [3,4,5],
  [3,2,6],
  [2,2,1]
]
```

Return `4`

The longest increasing path is `[3, 4, 5, 6]`. Moving diagonally is not allowed.

```
class Solution {
public:
    int longestIncreasingPath(vector<vector<int>>& matrix) {
        if(matrix.size()==0) return 0;
        int m=matrix.size();
        int n=matrix[0].size();
        vector<vector<int>> check(m,vector<int>(n,0));
        int res = 1;
```

```
        for(int i=0;i<m;i++)
        {
            for(int j=0;j<n;j++)
            {
                res = max(res, dfs(matrix, m, n, i, j, check));
            }
        }
        return res;
    }

    int dfs(vector<vector<int>>& matrix, int m, int n, int row, int col,
vector<vector<int>>& check)
    {
        int max_len = check[row][col];
        if(check[row][col]>0) return max_len;
        if(row>0 && matrix[row-1][col]>matrix[row][col]) max_len =
max(max_len,dfs(matrix,m,n,row-1,col,check));
        if(row<m-1 && matrix[row+1][col]>matrix[row][col]) max_len =
max(max_len,dfs(matrix,m,n,row+1,col,check));
        if(col>0 && matrix[row][col-1]>matrix[row][col]) max_len =
max(max_len,dfs(matrix,m,n,row,col-1,check));
        if(col<n-1 && matrix[row][col+1]>matrix[row][col]) max_len =
max(max_len,dfs(matrix,m,n,row,col+1,check));
        check[row][col] = ++max_len;
        return max_len;
    }

};
```