Question: https://leetcode.com/problems/longest-increasing-path-in-a-matrix/

So what we do is, we perform dfs for each position of the grid; which will calculate the max len of strictly increasing sequence.

Now let's say thet we have already calculated a position while evaluating some other grid.

So we can memorize the value in a cache matrix, by getting the value of each cell using backtracking.

Code:

class Solution {

public static final int[][] dirs = {{0, 1}, {1, 0}, {0, -1}, {-1, 0}};

public int longestIncreasingPath(int[][] matrix) {

int m = matrix.length, n = matrix[0].length;

int[][] cache = new int[m][n];

int max = 1;

for(int i = 0; i < m; i++) {

for(int j = 0; j < n; j++) {

if(cache[i][j]==0)max = Math.max(max, dfs(matrix, i, j, m, n, cache));

}

}

return max;

}

public int dfs(int[][] matrix, int i, int j, int m, int n, int[][] cache) {

if(cache[i][j] != 0) return cache[i][j];

int max = 1;

for(int[] dir: dirs) {

int x = i + dir[0], y = j + dir[1];

if(x < 0 || x >= m || y < 0 || y >= n || matrix[x][y] <= matrix[i][j]) continue;

max = Math.max(max, 1 + dfs(matrix, x, y, m, n, cache));

}

cache[i][j] = max;

return max;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>