PROBLEM

Minimum Points To Reach Destination

Hard Accuracy: 33.0% Submissions: 24K+ Points: 8

Given a **m*n** grid with each cell consisting of **positive**, **negative**, or **zero** point. We can move across a cell only if we have positive points. Whenever we pass through a cell, points in that cell are added to our overall points, the task is to find **minimum initial points** to reach cell **(m-1, n-1)** from **(0, 0)** by following these certain set of rules :

1. From a cell (i, j) we can move to (i + 1, j) or (i, j + 1).
2. We cannot move from (i, j) if your overall points at (i, j) are <= 0.
3. We have to reach at (n-1, m-1) with minimum positive points i.e., > 0.

Example 1:

Input:
m = 3, n = 3
points = {{-2,-3,3},
{-5,-10,1},
{10,30,-5}}
Output:
7
Explanation: 7 is the minimum value to reach the destination with positive throughout the path. Below is the path. (0,0) -> (0,1) -> (0,2) -> (1, 2) -> (2, 2) We start from (0, 0) with 7, we reach (0, 1) with 5, (0, 2) with 2, (1, 2) with 5, (2, 2) with and finally we have 1 point (we needed greater than

Example 2:

Input:
m = 3, n = 2
points = {{2,3},
{5,10},
{10,30}}
Output:
1
Explanation: Take any path, all of them are positive. So, required one point at the start

Your Task:
You don't need to read input or print anything. Complete the function **minPoints()** which takes **m,n** and 2-d vector **points** as input parameters and returns the **minimum initial points** to reach cell **(m-1, n-1)** from **(0, 0).**

Expected Time Complexity: O(n*m)
Expected Auxiliary Space: O(n*m)

Constraints:
$1 \leq m \leq 10^3$
$1 \leq n \leq 10^3$
$-10^3 \leq points[i][j] \leq 10^3$

CODE

```
#User function Template for python3

class Solution:

    def minPoints(self, m, n, points):

        # code here

        dp=[[float('inf') for _ in range(n+1)]for _ in range(m+1)]

        dp[m-1][n]=1

        dp[m][n-1]=1

        for i in range(m-1,-1,-1):

            for j in range(n-1,-1,-1):

                dp[i][j] = max(1, (min( dp[i+1][j],dp[i][j+1] )- points[i][j] ) )

        return dp[0][0]
```