**Problem: Dungeon Game**

**Scenario**

The princess is captured and imprisoned in the bottom-right corner of a dungeon, which is laid out in a 2D grid. Our valiant knight starts in the top-left room and must fight through the dungeon to rescue the princess. Each room in the dungeon has either threats (represented by negative integers), magic orbs (positive integers), or is empty (0). The knight can only move rightward or downward in each step. The goal is to determine the knight's minimum initial health so that he can rescue the princess.

**Problem Statement**

The demons had captured the princess and imprisoned her in **the bottom-right corner** of a dungeon. The dungeon consists of m x n rooms laid out in a 2D grid. Our valiant knight was initially positioned in **the top-left room** and must fight his way through dungeon to rescue the princess.

The knight has an initial health point represented by a positive integer. If at any point his health point drops to 0 or below, he dies immediately.

Some of the rooms are guarded by demons (represented by negative integers), so the knight loses health upon entering these rooms; other rooms are either empty (represented as 0) or contain magic orbs that increase the knight's health (represented by positive integers).

To reach the princess as quickly as possible, the knight decides to move only **rightward** or **downward** in each step.

Print *the knight's minimum initial health so that he can rescue the princess*.

**Note** that any room can contain threats or power-ups, even the first room the knight enters and the bottom-right room where the princess is imprisoned.

**Input Format**

1. The first line contains two integers, n and m, representing the number of rows and columns in the dungeon, respectively.
2. The next n lines each contain m space-separated integers, representing the dungeon grid where dungeon[i][j] represents the health change upon entering room (i, j).

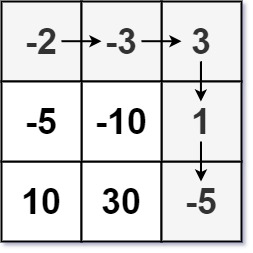
**Constraints**

* 1 <= dungeon.length, dungeon[0].length <= 200
* -1000 <= dungeon[i][j] <= 1000

**Output Format**

* Print an integer representing the knight's minimum initial health.

**Sample Input**



3 3

-2 -3 3

-5 -10 1

10 30 -5

**Sample Output**

7

**Explanation**

The knight's initial health must be at least 7 if he follows the optimal path: RIGHT -> RIGHT -> DOWN -> DOWN.

**Additional Test Cases**

**Test Case 1**

**Input:**

1 1

0

**Output:**

1

**Test Case 2**

**Input:**

2 3

-2 -3 3

10 30 -5

**Output:**

3

**Test Case 3**

**Input:**

3 3

1 -3 3

0 -10 -2

3 -5 0

**Output:**

2

**Test Case 4**

**Input:**

3 3

1 -1 3

0 -1 -4

3 -5 -3

**Output:**

5

**Test Case 5**

**Input:**

3 3

1 -1 3

10 -1 -4

3 -5 -3

**Output:**

1

**Solution Approach**

To solve this problem, we can use dynamic programming to calculate the minimum health required at each cell in the dungeon starting from the bottom-right corner and working our way back to the top-left corner.

We need to keep track of the minimum health required to enter each cell and ensure that the knight's health never drops to zero or below. Here's how you can do it:

1. Create a 2D array dp with the same dimensions as the dungeon.
2. Initialize dp[i][j] to be the minimum health needed to reach the princess from cell (i, j).
3. Start from the bottom-right corner and work your way backward to the top-left corner.
4. For each cell, calculate the minimum health needed to move right or down, ensuring that the health is always at least 1.

Here's the code to implement the solution:

python

def calculateMinimumHP(dungeon):

n, m = len(dungeon), len(dungeon[0])

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

dp[n][m-1] = dp[n-1][m] = 1

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

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

min\_health\_on\_exit = min(dp[i+1][j], dp[i][j+1])

dp[i][j] = max(min\_health\_on\_exit - dungeon[i][j], 1)

return dp[0][0]

# Reading input

n, m = map(int, input().split())

dungeon = [list(map(int, input().split())) for \_ in range(n)]

# Printing output

print(calculateMinimumHP(dungeon))

This code computes the minimum initial health required for the knight to rescue the princess while ensuring that his health never drops to zero or below.