Approach :

STORAGE & MEANING

It's fairly simple. We make an array for DP of the dimensions same as that of the matrix i.e. n*m.

The meaning assigned to a single cell in this grid is that "if we would have started from this cell, what is the maximum amount of points we could have collected".

DIRECTION

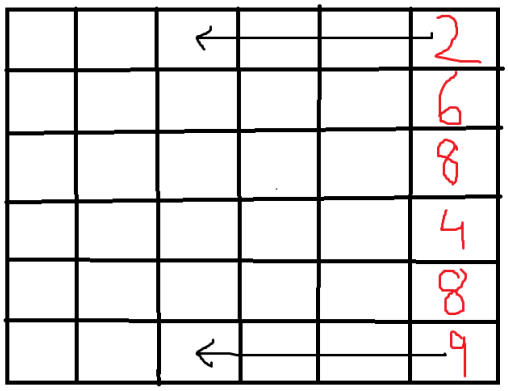
We always try to solve the question for a simpler and smaller problem via which we move on to the bigger problem.

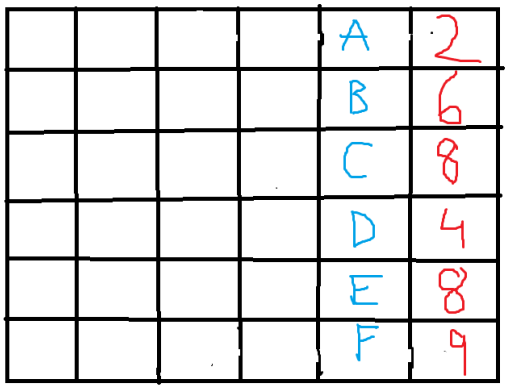
We want you to meditate on the fact that if we had started from any cell of the "end" column ,then the maximum amount of points we could have collected is the same as the amount of points corresponding to the cell value in the input array (according to the "Meaning" we assigned to a single cell).

Why do you think it is so?

It's because we can only take a step in the right direction. In the end column, taking a step anywhere to the right isn't possible because the grid comes to an end after that.

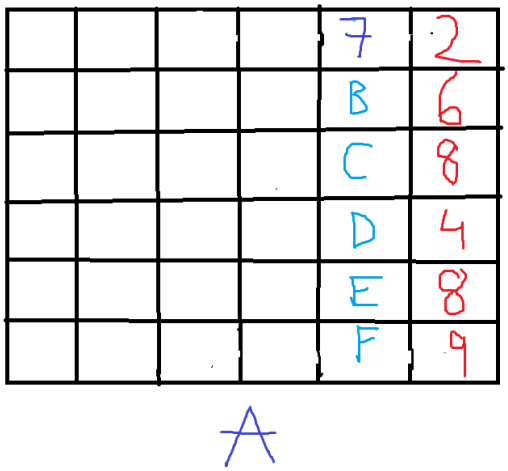
Hence our smaller problem lies at the "end" column. So we move from right to the left direction while calculating the maximum amount.



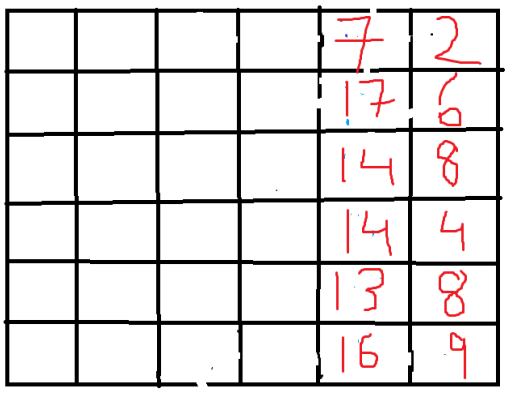


Once we have all the maximum amounts for the "end" column, we try to calculate the maximum amount for the column before that. We first calculate it for 'A'.

A' has 2 options whether to go to the right i.e. '2' or right down i.e. '6'.Since, 6>2, then for maximum amount of points, 'A' goes to 6. Hence we store the sum of the original value at the cell A i.e. 1 and the value 6. Hence we store 6+1=7 at A.



Similarly doing for all the cells we got



B. This cell has 3 options: 2, 6 and 8. Since 8 is the largest of them, therefore the value of cell B is 9+8=17.

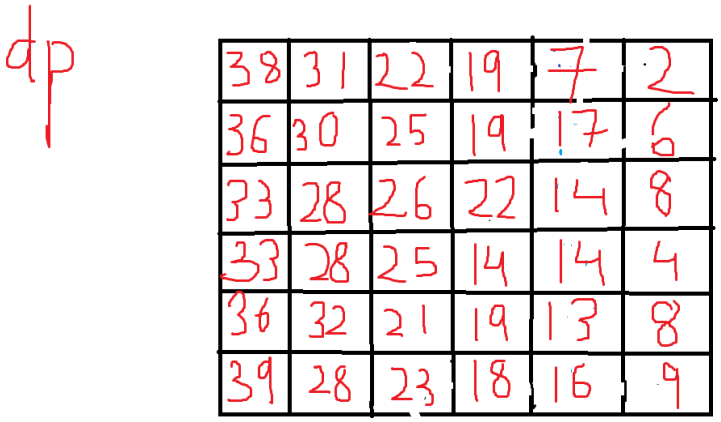
C. This cell has 3 options: 6, 8 and 4. Since 8 is the largest of them, therefore the value of cell C is 6+8=14.

D. This cell has 3 options: 8, 4 and 8. Since 8 is the largest of them, therefore the value of cell D is 6+8=14.

E. This cell has 3 options: 4, 8 and 9. Since 9 is the largest of them, therefore the value of cell E is 4+9 =13.

F. This cell has 2 options: 8 and 9. Since 9 is the largest of them, therefore the value of cell F is 7+9=16.

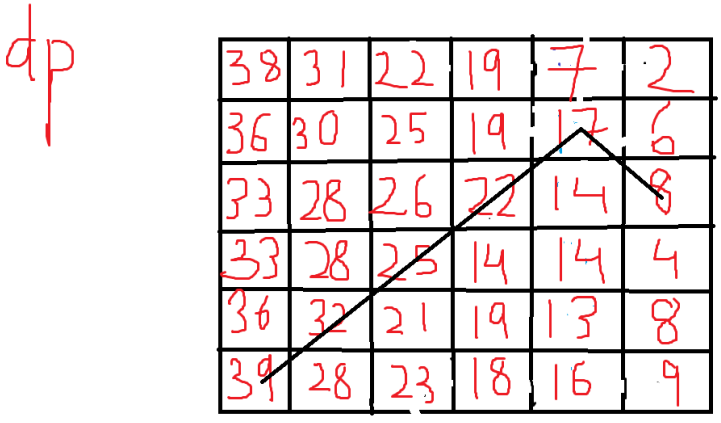
This way you can solve the maximum amount for all the columns.



Now that we have filled the entire grid, we look at the "start" column and think which cell of the "start" column would be wise to start our journey of collecting points from? We obviously chose the cell with maximum value i.e. 39. Don't understand why?

Go and check the "meaning" of the grid again. "Each cell is assigned with the maximum amount of points that would be collected if we start from that cell". Hence, 39 is the maximum value that could be traced out of all the paths if we started from that cell.

The required path for amount 39 would like this:



Code:

```
def maxpoints(Maze, dp):
    for j in range(len(dp[0])-1, -1, -1):
        for i in range(len(dp)-1, -1, -1):
            if j == len(dp[0])-1:
                dp[i][j] = Maze[i][j]
            elif i == 0:
                dp[i][j] = max(dp[i][j+1], dp[i+1][j+1])+Maze[i][j]
            elif i == len(dp)-1:
                dp[i][j] = max(dp[i][j+1], dp[i-1][j+1])+Maze[i][j]
            else:
                dp[i][j] = max(dp[i][j+1], dp[i+1][j+1],
                               dp[i-1][j+1]) + Maze[i][j]

    col = 0
    maximum = 0

    for i in range(len(dp)):
        maximum = max(maximum, dp[i][col])

    return maximum


n = int(input())
m = int(input())


Maze = []
for i in range(n):
    Maze.append(list(map(int, input().split())))

dp = [[0 for x in range(m)]for y in range(n)]

print(maxpoints(Maze, dp))
```

Time Complexity:

O(n^2)

Space Complexity:

O(n^2)