**Paths leading to Treasure**

You and your friends went camping in a jungle. There you saw a mysterious looking piece of paper fallen near a tree. That was the map of a Treasure. Now you guys are really excited about it and wanna give it a try. Let's go for the treasure hunt and make the trip more adventurous.

But there comes a plot twist. The map that you found is quite unique. The paths leading to treasure are given in the form of a matrix with some rows and columns.

The top-left position (or the first element) of that matrix denotes the initial point of the path, which happens to be the tree where you found the map. Similarly, the bottom-right position (or the last element) of the matrix denotes the final destination which is a little cave where you’re gonna find the treasure.

Well the map shows many paths. But the forest department has blocked some of them which nobody exactly knows. So now you guys need to go through all the possible paths by breaking into teams, equal to the number of unique paths.

**Problem Description**

Write a code to find the number of unique paths leading to the treasure. A matrix of *M* rows and *N* columns are given as input. Your code should return the number of unique paths from first to the last element of the matrix.

**Input Format**

Input is the number of rows and columns, *M* and *N*.

**Output Format**

Number of unique paths, as explained above.

**Constraints**

1<= *M*,*N* <=15

**Sample Input**

2 3

**Sample Output**

3

**Explanation**

The given matrix with 2 rows and 3 columns will have 6 elements. Let the elements are like:

A B C

D E F

Here A is the starting point and F is the destination point. Clearly there are only 3 unique paths, that are ABCF, ADEF and ABEF. Note that paths like AEF or ADEBCF are not to be considered.