**Coverage of all Zeros in a Binary Matrix**

Show Topic Tags   

Given a binary matrix (M[][]) having n rows and m columns, your task is to find the sum of coverage of all zeros in the matrix where coverage for a particular 0 is defined as total number of ones around a zero in left, right, up and bottom directions.  
  
Examples:

Input : M[][] = 0 0 0 0

1 0 0 1

0 1 1 0

0 1 0 0

Output : 13

First and last zeros are surrounded by only one 1's each.

Zeros in second row are surrounded by two 1's each.

Similarly counting for others, we get total count as

1 + 1 + 2 + 2 + 2 + 2 + 1 + 2 = 13

Input : M[][] = 1 1 1 0

1 0 0 1

Output : 6

Coverage of first zero is 2

Coverages of other two zeros is 2

Total coverage = 2 + 2 + 2 = 6

**Input:**  
The first line of input contains an integer T denoting the number of test cases. Then T test cases follow. Each test case contains two space separated integer n,m denoting no of rows, no of columns of the matrix  M respectively. Then in the next line are n\*m space separated values of the matrix M.  
  
**Output:**  
For each test case in a new line print the required output.  
  
**Constraints:**  
1<=T<=100  
1 <= n,m <= 20  
  
**Example:  
Input:**  
2  
4 4  
0 0 0 0 1 0 0 1 0 1 1 0 0 1 0 0  
2 4  
1 1 1 0 1 0 0 1  
**Output:**  
13  
6

\*\*For More Examples Use Expected Output\*\*

<http://practice.geeksforgeeks.org/problems/coverage-of-all-zeros-in-a-binary-matrix/0>

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication250;

import java.io.\*;

import java.math.\*;

import java.util.\*;

/\*\*

\*

\* @author Administrador

\*/

public class JavaApplication250 {

public static void main(String[] args) throws IOException {

// TODO code application logic here

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int t = Integer.parseInt(br.readLine());

while(t-- > 0) {

String[] nm = br.readLine().trim().split( " ");

int n = Integer.parseInt(nm[0]);

int m = Integer.parseInt(nm[1]);

String[] input = br.readLine().trim().split(" ");

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

int index =0;

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

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

matrix[i][j] = Integer.parseInt(input[index++]);

}

}

/\*

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

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

System.out.print(matrix[i][j] + " ");

}

System.out.println();

}\*/

int ans =0;

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

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

if(matrix[i][j] == 0) {

if(i-1 >=0) {

if(matrix[i-1][j] == 1){

ans++;

}

}

if(i+1 < n) {

if(matrix[i+1][j] == 1){

ans++;

}

}

if(j-1>=0){

if(matrix[i][j-1] == 1){

ans++;

}

}

if(j+1 <m) {

if(matrix[i][j+1] == 1){

ans++;

}

}

}

}

}

System.out.println(ans);

}

}

}