American Computer Science League

PROBLEM: Given 3 two-dimensional arrays of the same shape, move through the arrays, starting in the upper left corner. At each step, look at the neighboring cell to the right and the cell below. Find the largest of these values in all 3 arrays and move to that cell location. However, if there are two or more largest values, move to the cell location that is diagonally down and to the right. Stop when you reach a cell location that is in the last row or the last column. Output the sum of the smallest value in each cell location that was visited along the way.

Array A			
1	2	3	4
7	7	8	9
5	6	7	8

Array B			
6	8	6	4
4	5	3	2
8	3	1	9

Array C			
3	6	7	3
4	6	2	1
3	2	5	5

In the example above, the traversal proceeds as follows:

> Step 1: Start at the cell in the upper left corner (0,0). The neighbors are 2 and 7 in array A, 8 and 4 in array B, and 6 and 4 in array C. The largest is 8 so move to the right to cell location (0,1).

> Step 2: The neighbors of cell (0,1) are 3 and 7 in array A; 6 and 5 in array B; 7 and 6 in array C. The largest is 7, which happens from array A and array C. Move diagonally down and to the right, to cell location (1,2).

> Step 3: At location (1,2), the largest value from {9,7,2,1,1,5} is 9 so move to location (1,3). The process stops because location (1,3) is in the last column.

Thus, the visited cells in this example are (0,0), (0,1), (1,2), and (1,3). The smallest values in each of those cell locations among all 3 arrays are 1, 2, 2, and 1, which sums to 6.

INPUT: There will be 5 sets of data, each containing 4 lines of integer values. The first line will contain 2 positive integers representing the number of rows and the number of columns in each array. Each of the next three lines will contain the values of the 3 arrays in row-major order starting with location (0,0).

OUTPUT: Print the sum of the smallest value of each visited cell location.

SAMPLE INPUT:

```
3 4
1 2 3 4 7 7 8 9 5 6 7 8
6 8 6 4 4 5 3 2 8 3 1 9
3 6 7 3 4 6 2 1 3 2 5 5
4 2
31 17 24 19 15 29 22 26
25 13 25 18 19 27 19 13
12 15 17 18 29 16 25 20
4 5
3 1 4 1 5 9 2 6 5 3 5 8 9 7 9 3 2 3 8 4
6 2 6 4 3 3 8 3 2 7 7 2 3 8 3 3 4 6 2 6
5 8 3 2 3 9 7 9 8 5 3 5 6 2 9 5 1 4 1 3
```

SAMPLE OUTPUT:

```
1. 6

2. 60

3. 16
```

TEST INPUT (http://xxxx):

```
4 3
3 1 4 1 5 2 2 6 5 3 5 8
9 7 1 3 2 6 8 4 6 2 6 4
3 2 1 3 2 1 1 2 3 1 2 3
4 6
4 8 3 7 1 6 7 6 2 4 3 3 7 5 1 0 5 8 2 0 9 5 3 2
4 4 5 9 2 3 0 2 1 9 6 4 0 6 2 8 4 2 0 8 9 3 5 2
6 9 5 0 3 4 8 7 5 3 4 2 1 1 7 10 6 7 9 3 1 2 3 2
5 3
31 41 59 26 53 58 97 93 23 84 62 64 33 83 27
95 28 84 19 71 69 39 93 75 10 85 20 97 49 44
59 23 78 61 40 62 97 20 49 98 62 80 34 83 53
4 5
-3 -1 -4 -1 -5 -9 -2 -6 -5 -3 -5 -8 -9 -7 -4 -3 -5 -3 -8 -4
-6 -2 -6 -4 -3 -3 -8 -3 -2 -7 -9 -5 -2 -8 -8 -4 -4 -9 -7 -1
-6 -9 -3 -9 -9 -3 -7 -5 -1 -5 -8 -2 -9 -7 -4 -9 -4 -4 -5 -9
5 5
1 2 3 4 5 6 7 8 9 10 11 12 13 12 11 10 9 8 7 8 5 4 3 2 1
2 4 6 8 10 12 14 16 18 20 22 20 18 16 14 12 10 8 26 4 2 4 6 8 10
1 3 5 7 9 11 13 15 17 19 21 23 25 23 21 19 17 15 13 1 9 7 5 3 1
```

TEST OUTPUT (http://xxx):

1. 9
2. 11
3. 159
4. -49
5. 63