F004A. Layers of onion — P88060_en

Given a matrix, all the elements which are in the first row, in the first column, in the last row and the last column, form the *first layer* of the matrix. Equally, the elements which are in the second row, in the second column, in the penultimate row and the penultimate column (but that are not in the first layer), form the *second layer*. The concept of layer becomes generalized in the same way for all the elements of the matrix.

For instance, for a matrix with 8 rows and 9 columns, the next diagram shows in which layer is each element of the matrix:

1	1	1	1	1	1	1	1	1
1	2	2	2	2	2	2	2	1
1	2	3	3	3	3	3	2	1
1	2	3	4	4	4	3	2	1
1	2	3	4	4	4	3	2	1
1	2	3	3	3	3	3	2	1
1	2	2	2	2	2	2	2	1
1	1	1	1	1	1	1	1	1

Your task is to write a program that, given different integer matrices, calculates, for each one, the values of the minimum and the maximum of all their layers. For instance, for the matrix

15	1	92	92	82	15	1	92	92
46	31	13	3	32	46	32	13	13
15	3	32	2	22	16	12	34	14
23	17	33	1	2	23	17	32	21
13	63	56	2	21	13	62	66	4
43	9	8	52	11	43	9	8	64
53	96	6	42	17	63	99	6	14
99	94	5	51	21	99	94	5	51

The minimum and the maximum of the first layer are 1 and 99, the minimum and the maximum of the second layer are 3 and 99, the minimum and the maximum of the third layer are 2 and 62, and the minimum and the maximum of the fourth layer are 1 and 23. We represent this information with a vector of pairs:

1,99	3,99	2,62	1,23

The main program is already done; do not modify it. Using the types

```
struct Info {
    int min, max;
};

typedef vector<vector<int> > Matrix;
```

You must implement the function

vector<Info> info_layers (**const** *Matrix*& *mat*);

that, given a rectangular matrix `mat` (with, at least, a row and a column), returns a vector with as many positions as layers has the matrix, where the position *i* of the vector contains the minimum and the maximum values of the layer *i*+1 of `mat`.

Sample input

```
8 9
15  1 92 92 82 15  1 92 92
46 31 13  3 32 46 32 13 13
15  3 32  2 22 16 12 34 14
23 17 33  1  2 23 17 32 21
13 63 56  2 21 13 62 66  4
43  9  8 52 11 43  9  8 64
53 96  6 42 17 63 99  6 14
99 94  5 51 21 99 94  5 51

10 10
1 1 1 1 1 1 1 1 1 1
1 2 2 2 2 2 2 2 2 1
1 2 3 3 3 3 3 3 2 1
1 2 3 4 4 4 4 3 2 1
1 2 3 4 5 5 4 3 2 1
1 2 3 4 5 5 4 3 2 1
1 2 3 4 4 4 4 3 2 1
1 2 3 3 3 3 3 3 2 1
1 2 2 2 2 2 2 2 2 1
1 1 1 1 1 1 1 1 1 1

5 4
 1  2  3  4
 5  6  7  8
 9  1 11 12
13 14 15 16
17 28 29 20

3 5
11 21 31 74 85
17 18 69 10 11
13 14 15 16 17

1 1
-666

3 1
10000001
-10000001
0

1 9
6 6 6 8 8 8 9 9 9
```

Sample output

```
matriu 1: 1,99 3,99 2,62 1,23
matriu 2: 1,1 2,2 3,3 4,4 5,5
matriu 3: 1,29 1,15
matriu 4: 11,85 10,69
matriu 5: -666,-666
matriu 6: -10000001,10000001
matriu 7: 6,9
```

Problem information

Author : Professorat de P1
Translator : Carlos Molina
Generation : 2013-09-02 15:09:10