Appleby Contest '19 P5 - Matrix Operation

Time limit: 2.0s **Memory limit:** 1G
Python: 4.0s

Plasmatic is writing a problem for a programming contest. However, there is one issue: he can't solve it and needs your help.

The problem goes as follows:

> Given an $N \times N$ matrix of numbers, what is the longest **strictly** increasing path through the matrix, assuming that you can only move horizontally or vertically?

Being the good friend that you are, can you help him do it?

Note: If you are using Python 2/3, submit using PyPy 2/3 instead because it is significantly faster

Input Specification

The first line contains N.

The next N lines each contain N space separated integers x that denote the matrix.

Output Specification

On a line, output the longest **strictly** increasing path in the matrix.

Constraints

For all subtasks:

$1 \leq x \leq 10^9$

Subtask 1 [5%]

$1 \leq N \leq 2$

Subtask 2 [20%]

$1 \leq N \leq 50$

Subtask 3 [75%]

$1 \leq N \leq 1500$

Sample Input 1

```
3
9 8 4
7 2 3
6 1 5
```

Sample Output 1

```
5
```

Explanation for Sample Output 1

To get the longest **strictly** increasing path you can traverse the values of the matrix in the following order: $1, 2, 3, 4, 8, 9$. Since path length is calculated by how many movements you have to make and not by the amount of values that you touch, the answer is 5 instead of 6.

Note that the values aren't guaranteed to be unique in the actual test data.

Sample Input 2

```
3
1 2 3
8 9 4
7 6 5
```

Sample Output 2

```
8
```

Explanation for Sample Output 2

The longest path is a spiral starting from 1 and ending at 9.