IOI 2010: Team Selection Test

19th June, 2010

Question	Marks	Time Limit
Home	100	1 second
Crates	100	1 second
Roads	100	1 second

Rules

1. The test will run to a total of 5 hours
2. Your submissions should be written in C++.
3. While all questions carry the same marks, they might be of the varying difficulties. It is to your advantage to identify the easier questions and complete them first.
4. You may ask questions only during the first hour of the test. Make sure that you read and understand the problems in this time.
5. You may not bring in electronic items, books or stationery into the examination room. However, you may bring in your own pen or pencil. Paper for draft work will be supplied.

Home

I sleepily open my eyes and wonder where I am. And what are these bright lights heading towards me?

Honk! Honk!

[Scream] I manage to roll onto the pavement before the vehicle overruns me.

I vaguely remember going to a party last night. Now how many drinks did I have?

Oh god... if the cops catch me now, I'll be locked up for the night. Better find my way home right now.

Good thing I brought a map....

Given a map with your current location and the location of your home, find the shortest path to your house.

Note that you can only move in the directions North, South, East and West (you are too drunk to do anything else)

Input Data
Read your input from the standard input. The first line will contain two values: the width and height of the map, as integers, separated by a single space.

You will be given a map of the town you live in, in the following format:

```
XXXXXXXXXX
XsX      X
X X XX XXX    X = Private property (impassable)
X X XX XXX    s = Starting position
X X XX XXX    h = Position of their home
X X XX XXX    (space) = Road
X X XX XXX
X X XX  hX
X   XXXXXX
XXXXXXXXXX
```

Output Data
Print your output to the standard output.

Your output should consist of one integer: the least number of steps needed to move from your starting position to your house.

For the example above, the output should be **27**

Further information

The shortest path is illustrated in the map below

```
XXXXXXXXXX
XsX@@@@  X
X@X@XX@XXX
X@X@XX@XXX    s  =  Starting position
X@X@XX@XXX    @  =  Your path
X@X@XX@XXX
X@X@XX@XXX
X@X@XX@@@X
X@@@XXXXXX
XXXXXXXXXX
```

Note that moving to the position of your home is also counted as a step.

Constraints
The width and height will be between 4 and 10.

Crates

Oh... my head....

Even though I got home safely, I still have to go to work, damn it. And my hangover still hasn't gone down....

Now, where are the controls of this crane? And what am I supposed to do? My head still feels like it is stuffed with cotton wool....

Ah, yes. I have to stack crates one on top of the other, and create a pile as tall as possible.

Oh yes, I also remember - I can only stack one crate on top of another if the base of the crate on top is smaller (in at least one dimension and the other dimension can be the same but not larger) than that of the one below it.

And thanks to my hangover, I cannot remember how to rotate the crates at all....

For simplicity, assume that all crates are of the same height. You only need to consider the width and length of their bases.

Input Data
Read your input from the standard input.

The first line will contain the number of crates.

The width and length of the bases of each crate will be listed, one crate on each line. The width and length will be integers, both separated by one space.

For example:

3
4 6
2 7
10 12

Output Data
Print your output to the standard output.

Your output should consist of one integer: the height (in crates) of the tallest pile created.

For example, the output for the above example should be **2**

Further information

In the above example, there are 3 crates:

a. size 4 x 6
b. size 2 x 7
c. size 10 x 12

Note that the bases of crates **a** and **b** are smaller than that of crate **c**, which means that we can stack either of them on top of crate **c**.

However, the width of crate **a** is greater than that of crate **b**, so crate **a** cannot be stacked on top of crate **b**.

On the other hand, the length of crate **b** is greater than that of crate **a**, which means that crate **b** cannot be stacked on top of crate **a** either.

Therefore, the height of the highest possible pile would be 2 crates (which you could obtain by either stacking crate **a** on top of crate **c**, or by stacking crate **b** on top of crate **c**).

Constraints

The number of crates will be at most 100.

The width and length of the crates will be integers between 1 and 1000.

Roads

Finally, I've finished stacking the crates! Now, time to go back home for a nice nap.

Ugh... it's past office time. The roads will be packed now.

Hmmm... let me first calculate the fastest route home - *then* I'll get into the car.

Given a graph containing the roads and intersections in your city, find the fastest route back home.

Input Data
Read your input from the standard input.

The first line will contain the number of intersections, N.

This will be followed by N lines, progressively describing the intersections from 1 to N.

Each line will consist of N integers. If there is a road from intersection I to intersection J, the J^{th} value in the I^{th} row will contain the time taken to travel from I to J. If there is no road, that value will be 0.

Note that there is only one road between each pair of intersections, and the time taken to travel either way down the road is the same.

Assume that you start off at intersection 1, and your home is located at intersection N.

For example:

```
4
0 10 100 0
10 0 20 0
100 20 0 5
0 0 5 0
```

Output Data
Print your output to the standard output.

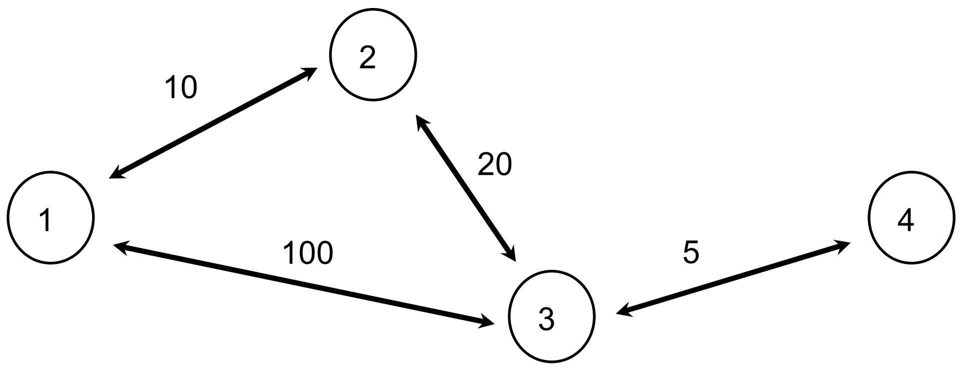
Your output should consist of one integer: the minimum time needed to travel between intersections 1 to N.

For example, the output for the above example should be **35**

Further information

In the above example, there are 4 intersections.

Going by the test data, the following graph of roads can be constructed:



Note that the fastest route is 1 → 2 → 3 → 4, giving a total time of 10 + 20 + 5 = 35.

Constraints

The number of intersections will be at most 1000.

The times required to travel between intersections will be integers between 1 and 1000.