IOI Training Camp 2017
Team Selection Tests, Day 2

Check SCC

There is a directed graph with N vertices (1-indexed) such that for all i, j ($i! = j$), there is either an edge from i to j or from j to i (or maybe both). But the edges of the graph are hidden from you. You only know the indegree and outdegree of each vertex. You can ask the grader questions of the form "Is there an edge from u to v" and the grader will reply with Yes or No. Your job is to find whether the entire graph is strongly connected or not. You can ask as many questions as you want but you have to stop as soon as you get N Yes replies.

You have to write a function

`void solve (int n, int& res)`

which calls a function

`vector<pair<int, int> > init ()`

once, and a function

`bool query (int i, int j)`

multiple times.

After doing

`vector < pair<int,int> > v = init();`

in your code, v[i].first will contain the in-degree of the $i + 1$-th vertex, and v[i].second will contain the out-degree of the $i + 1$-th vertex.

`query(u,v)`

returns True (which is referred to above as Yes) if (u, v) is an edge, and False otherwise. You will get an error if the parameters are not valid, or $i == j$.

Your solve function should end with the variable res being 0 if you think the graph is not strongly connected, and 1 if you think it is.

You will be provided two files:
grader.cpp: You should not edit this file. This files reads a description of the graph from the input, calls

`void solve (int n, int& res)`

once and compares the int res given by solve() with the actual answer.
grader.cpp also implements the functions init() and query().
The exact implementation of grader.cpp in the online judge may vary from the code given to you, but the two will be functionally equivalent.

`dummy_solution.cpp`

: This is the file in which you write your code for solve(). There are some function headers, etc., that you should not modify or delete. The place where you have to insert your code is clearly marked.

Compiling and testing your code

To compile your code use the following command:

```
g++ dummy_grader.cpp dummy_solution.cpp -o grader
```

To provide test inputs to the code, see the section Input format below.

Submissions

You should submit your modified version of

```
dummy_solution.cpp
```

to the online judge, which will compile it with grader.cpp and evaluate your implementation of solve(). You get full marks if your code is able to exactly identify whether the graph is strong connected or not in less than or equal to N Yes-query replies.

Input

All input is done by grader.cpp so you do not need to write any code for input. However, to test your program, you can supply grader.cpp with the definition of a graph in the following format.
First line contains N. The next N lines contain N bits. The ith bit in the jth line should be a 1 if there is an edge from i to j. 0 otherwise.

Output

All output is done by grader.cpp so you do not need to write any code for output. Your implementation of solve() has to compute and store the answer in res, as described above.

Constraints

- $1 \leq N \leq 1000$

Limits

Time: 2 seconds

Memory: 256 MB