Problem B. majority

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

small 1 points *2 minute timeout*	The contest is finished.
large 20 points *10 minute timeout*	The contest is finished.

Problem

Your country is electing its president, and you are in charge of the new electronic voting system. The citizens have voted, and now you have to check if any of the candidates obtained a *majority* — that is, if there is a candidate for whom more than half of the citizens voted.

Input

The input library will be called "majority", see the sample inputs below for examples in your language. It will define two methods: GetN(), which will return the number of voting citizens *N*, and GetVote(i), which will (for $0 \le i < N$) return the identifier of the candidate for whom citizen *i* voted.

Output

If any candidate obtained a majority of the votes, output the identifier of that candidate. Otherwise, output the string "NO WINNER" (quotes for clarity only). A single call to GetVote(i) will take approximately 0.025 microseconds.

Limits

Each node will have access to 128MB of RAM, and a time limit of 3 seconds.
$0 \le$ GetVote(i) $\le 10^9$ for all *i* with $0 \le i < N$.

Small input

Your solution will run on 10 nodes.
$1 \le$ GetN() ≤ 1000.

Large input

Your solution will run on 100 nodes.
$1 \le$ GetN() $\le 10^9$.

Sample

Input	Output
See the input files below.	For sample input 1: 7 For sample input 2: NO WINNER For sample input 3: NO WINNER

Note: the same problem idea was used by us in a tutorial in the Algorithmic Engagements contest in 2014.

Sample input libraries:
Sample input for test 1: majority.h [CPP] majority.java [Java]
Sample input for test 2: majority.h [CPP] majority.java [Java]
Sample input for test 3: majority.h [CPP] majority.java [Java]


Powered by
Google Cloud Platform