Distributed Round 1 2016

A. Testrun

B. oops

C. rps

D. crates

E. winning_move

Contest Analysis

Questions asked 8

Submissions

Testrun	
0pt	Not attempted **0/422 users** correct (0%)
oops	
2pt	Not attempted **893/925 users** correct (97%)
12pt	Not attempted **756/882 users** correct (86%)
rps	
1pt	Not attempted **789/857 users** correct (92%)
15pt	Not attempted **585/783 users** correct (75%)
crates	
8pt	Not attempted **557/673 users** correct (83%)
25pt	Not attempted **258/433 users** correct (60%)
winning_move	
3pt	Not attempted **635/700 users** correct (91%)
34pt	Not attempted **49/309 users** correct (16%)

Top Scores

simonlindholm	100
tomconerly	100
eatmore	100
cgy4ever	100
bmerry	100
Simon.M	100
Klockan	100
tczajka	100
tkociumaka	100
Zlobober	100

Problem E. winning_move

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 3 points *2 minute timeout*	The contest is finished.
large 34 points *10 minute timeout*	The contest is finished.

Problem

The Only Winning Move

Perhaps you have played this game before: Every player submits a positive integer. The winner, if any, is the player who submitted the smallest positive integer that was submitted by no other player.

Your friends invited you to play this game, but you decided that game theory is complex and the only winning move is not to play. Instead, you volunteered to judge the game. Given the players' choices, can you determine what the winning number was?

Input

The input library will be called "winning_move"; see the sample inputs below for examples in your language. It defines two methods:

- **GetNumPlayers()**:
  - Takes no argument.
  - Returns a 64-bit number: the number of players in the game.
  - Expect each call to take 0.12 microseconds.
- **GetSubmission(playernum)**:
  - Takes a 64-bit number in the range 0 ≤ playernum < GetNumPlayers().
  - Returns a 64-bit number: the number chosen by the player with ID number playernum.
  - Expect each call to take 0.12 microseconds.

Output

Output one value: the winning number, or 0 if there was no winner.

Limits

Time limit: 4 seconds.
Memory limit per node: 128 MB.
Maximum number of messages a single node can send: 1000.
Maximum total size of messages a single node can send: 1 GB.
1 ≤ GetSubmission(playernum) ≤ 10^{18}, for all playernum.

Small dataset

Number of nodes: 10.
1 ≤ GetNumPlayers() ≤ 1000.

Large dataset

Number of nodes: 100.
1 ≤ GetNumPlayers() ≤ 35000000.

Sample

```
Input                      Output

See input files below.     For sample input 1:
                           4
                           For sample input 2:
                           3
                           For sample input 3:
                           0
```

Sample input libraries:
Sample input for test 1: winning_move.h [CPP] winning_move.java [Java]

Sample input for test 2: winning_move.h [CPP] winning_move.java [Java]
Sample input for test 3: winning_move.h [CPP] winning_move.java [Java]