Submissions

Testrun

0pt	Not attempted **0/142 users** correct (0%)

sandwich

1pt	Not attempted **187/205 users** correct (91%)
15pt	Not attempted **141/178 users** correct (79%)

majority

1pt	Not attempted **170/176 users** correct (97%)
20pt	Not attempted **80/167 users** correct (48%)

shhhh

1pt	Not attempted **110/115 users** correct (96%)
30pt	Not attempted **69/102 users** correct (68%)

load_balance

2pt	Not attempted **94/101 users** correct (93%)
35pt	Not attempted **33/88 users** correct (38%)

Top Scores

iwi	105
simonlindholm	105
Murphy	105
stgatilov	105
Alexander86	105
microtony	105
eatmore	105
uwi	105
Marcin.Smulewicz	105
tczajka	105

Problem A. Testrun

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
0 points
2 minute timeout

The contest is finished.

Problem

This is a way to test your solutions, not a real problem!

When you submit a solution to this problem, it will run one testcase on a 100 nodes. This will allow you to estimate how fast your solution will run on our system.

Remember to change your solution appropriately before submitting it for real, so you don't fail because of a compilation error! The best way to check is to run your solution on the small input before submitting to the large input.

Input

There is no input for this problem. This means you should not include / import an input library.

Output

Doesn't really matter what you output. If your solution runs successfully to completion, it will be judged as "Wrong Answer".

Limits

Each node will have access to 1 GB of RAM, and a time limit of 26 seconds. The maximum number of messages a single node can send is 5000, and the maximum sum of the sizes of those messages is 8MB.
This problem only has one small test case. It will run on 100 nodes.

Problem B. sandwich

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 15 points *10 minute timeout*	The contest is finished.

Problem

Your friends made a brave attempt to beat the world record for making the longest sub sandwich ever. They failed or succeeded (it doesn't really matter), and proposed to you to eat as much of the sandwich as you want to.
The sandwich is composed of N parts. Each part has a taste value for you, which might be positive (meaning you want to eat it) or negative (meaning you would prefer not to). Ideally, you would just eat the tasty parts, but it's rude to break out the middle of the sandwich. So, instead you want to eat some part from the beginning, and some part from the end; and to make the total taste of what you eat as large as possible. The total taste is the sum of tastes of all the parts you have eaten.
Note: it's OK to eat the whole sandwich, or to eat nothing at all. Output the largest total taste of what you can eat.

Input

The input library will be called "sandwich", see the sample inputs below for examples in your language. It will define two methods: GetN(), which will return the number of parts of the sandwich, and GetTaste(i), which will return the taste of the *i*th part of the sandwich, for $0 \le i < N$.
A single call to GetTaste(i) will take approximately 0.01 microseconds.

Output

Output one number: the maximum possible total taste of the parts you will eat.

Limits

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

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() $\le 5 \times 10^8$.

Sample

Input	Output

`See the input files below.`	`For sample input 1:`
	`14`
	`For sample input 2:`
	`0`
	`For sample input 3:`
	`5`

Note: this problem idea was used by us in the practice round of the Algorithmic Engagements contest in 2014, and also displayed as an example in our FAQ.

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

Submissions

Testrun

0pt	Not attempted **0/142 users** correct (0%)

sandwich

1pt	Not attempted **187/205 users** correct (91%)
15pt	Not attempted **141/178 users** correct (79%)

majority

1pt	Not attempted **170/176 users** correct (97%)
20pt	Not attempted **80/167 users** correct (48%)

shhhh

1pt	Not attempted **110/115 users** correct (96%)
30pt	Not attempted **69/102 users** correct (68%)

load_balance

2pt	Not attempted **94/101 users** correct (93%)
35pt	Not attempted **33/88 users** correct (38%)

Top Scores

iwi	105
simonlindholm	105
Murphy	105
stgatilov	105
Alexander86	105
microtony	105
eatmore	105
uwi	105
Marcin.Smulewicz	105
tczajka	105

Submissions

Testrun

0pt	Not attempted **0/142 users** correct (0%)

sandwich

1pt	Not attempted **187/205 users** correct (91%)
15pt	Not attempted **141/178 users** correct (79%)

majority

1pt	Not attempted **170/176 users** correct (97%)
20pt	Not attempted **80/167 users** correct (48%)

shhhh

1pt	Not attempted **110/115 users** correct (96%)
30pt	Not attempted **69/102 users** correct (68%)

load_balance

2pt	Not attempted **94/101 users** correct (93%)
35pt	Not attempted **33/88 users** correct (38%)

Top Scores

iwi	105
simonlindholm	105
Murphy	105
stgatilov	105
Alexander86	105
microtony	105
eatmore	105
uwi	105
Marcin.Smulewicz	105
tczajka	105

Problem C. 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 ≤ 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.
$0 \le \text{GetVote(i)} \le 10^9$ for all *i* with $0 \le i < N$.

Small input

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

Large input

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

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]

Submissions

Testrun

0pt	Not attempted **0/142 users** correct (0%)

sandwich

1pt	Not attempted **187/205 users** correct (91%)
15pt	Not attempted **141/178 users** correct (79%)

majority

1pt	Not attempted **170/176 users** correct (97%)
20pt	Not attempted **80/167 users** correct (48%)

shhhh

1pt	Not attempted **110/115 users** correct (96%)
30pt	Not attempted **69/102 users** correct (68%)

load_balance

2pt	Not attempted **94/101 users** correct (93%)
35pt	Not attempted **33/88 users** correct (38%)

Top Scores

iwi	105
simonlindholm	105
Murphy	105
stgatilov	105
Alexander86	105
microtony	105
eatmore	105
uwi	105
Marcin.Smulewicz	105
tczajka	105

Problem D. shhhh

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 30 points *10 minute timeout*	The contest is finished.

Problem

You are at a reception. Along with a (huge) number of other people, you are seated at a round table, and listen to the speaker speak... and speak... and speak. You would like the speech to end, but that’s not likely to happen, so you’d at least like to tell your friend — who sits somewhere else at the same table — how badly bored you are.
However, it’s rude to speak loudly. So, you’ll whisper to one of your two neighbours, asking them to pass the message along. They’ll then whisper to the other neighbour, and so on, until the message reaches your friend. You now need to decide which neighbour to choose so the distance traveled by the message is as short as possible, and how long is it going to go.

Input

Each person at the table has a unique integer assigned, from 0 to N-1, where N is the number of people at the table (including you and your friend). You are assigned the identifier 0, and your friend is assigned the identifier 1.
The input library will be called "shhhh", see the sample inputs below for examples in your language. It will define three methods: GetN(), which will return the number N of people at the table, GetLeftNeighbour(i) for $0 \le i < N$, which will return the identifier of the left neighbour of the person with identifier i, and GetRightNeighbour(i) for $0 \le i < N$, which will return the identifier of the right neighbour of the person with identifier i.
A single call to GetLeftNeighbour(i) or GetRightNeighbour(i) will take approximately 0.015 microseconds.

Output

Output one line, containing one word and one number, separated by a single space. The word should be "LEFT" if it is faster to pass the message to your left neighbour, "RIGHT" if it is faster to pass the message to your right neighbour, or "WHATEVER" if the distance is the same in both directions (quotes around all words are for clarity only). The number should be the distance the message will have to travel (that is, the number of people who will hear the message, including your friend, but not including you).

Limits

Each node will have access to 128MB of RAM, and a time limit of 4 seconds.

Small input

Your solution will run on 10 nodes.
$2 \le \text{GetN}() \le 10^7$.

Large input

Your solution will run on 100 nodes.
$2 \le \text{GetN}() \le 10^9$.

Sample

Input	Output

`See input files below.`	`For sample input 1:`
	`WHATEVER 1`
	`For sample input 2:`
	`RIGHT 1`
	`For sample input 3:`
	`LEFT 2`

Note: the same problem idea (authored by Onufry Wojtaszczyk, Robert Obryk and Adam Polak) was used by us in the Algorithmic Engagements contest in 2014.

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

Submissions

Testrun

0pt	Not attempted **0/142 users** correct (0%)

sandwich

1pt	Not attempted **187/205 users** correct (91%)
15pt	Not attempted **141/178 users** correct (79%)

majority

1pt	Not attempted **170/176 users** correct (97%)
20pt	Not attempted **80/167 users** correct (48%)

shhhh

1pt	Not attempted **110/115 users** correct (96%)
30pt	Not attempted **69/102 users** correct (68%)

load_balance

2pt	Not attempted **94/101 users** correct (93%)
35pt	Not attempted **33/88 users** correct (38%)

Top Scores

iwi	105
simonlindholm	105
Murphy	105
stgatilov	105
Alexander86	105
microtony	105
eatmore	105
uwi	105
Marcin.Smulewicz	105
tczajka	105

Problem E. **load_balance**

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 2 points *2 minute timeout*	The contest is finished.
large 35 points *10 minute timeout*	The contest is finished.

Problem

At your flat, you take turns in bringing the groceries in. Each day, one person goes out and does the shopping for everybody who lives in the whole building. Today it's your turn, you did the shopping, and all these bags are heavy!
You can't do much about the fact the bags are heavy — people depend on you to bring the bags home. But they will be easier to carry if you distribute the load equally between the left and the right hand. So, you look at all the bags you have, and wonder whether it's possible to split them so that the weight of the ones you'll carry in your left hand will be equal to the weight of those you will carry in your right hand.

Input

The input library will be called "load_balance", see the sample inputs below for examples in your language. It will define two methods: GetN(), which will return the number of bags you have to carry, and GetWeight(i), which will return the weight of the *i*th bag, for 0 ≤ i < N.

Output

Output one string: "IMPOSSIBLE" if it is impossible to split the load equally, or "POSSIBLE" if it is possible (quotes are for clarity only).

Limits

Each node will have access to 512MB of RAM, and a time limit of 4 seconds.
$1 \le \text{GetWeight}(i) \le 10^{15}$ for all i with $0 \le i < \text{GetN}()$.

Small input

Your solution will run on 10 nodes.
$1 \le \text{GetN}() \le 30$.

Large input

Your solution will run on 100 nodes.
$1 \le \text{GetN}() \le 52$.

Sample

Input	Output

`See input files below.`	`For sample input 1:`
	`POSSIBLE`
	`For sample input 2:`
	`IMPOSSIBLE`
	`For sample input 3:`
	`POSSIBLE`

Note: this problem might be known to a few people, since we communicated it externally when speaking about Distributed Code Jam.

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