Coins [II]

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

points: 100 · problem code: coins2 · author: glipR

In this problem you are presenting with n evenly weighted coins, except you aren't! ***Two*** of the coins are actually heavier than the others (and are equally heavy), and you want to find them efficiently.

You are equipped with a seesaw, meaning you can put two piles of coins on either side and observe whether the seesaw dips to the left, right, or stays where it is.

In this interactive problem, determine the weighted coins using the seesaw, making few queries.

Interaction

This is an interactive problem. So rather than reading input once and printing output once, your program should communicate to the judge using input and output.

Interaction will start with a single integer n, the number of coins. These coins are labelled 1 through to n inclusive.

Queries may then start. Your program can use the seesaw by writing a query of the form

```
TEST {left coins} | {right coins}
```

Where left/right coins are space separated integers. For example, to weigh coins 1 and 3 against coins 2 and 4, you can do:

```
TEST 1 3 | 2 4
```

The judge will then print `LEFT` if 1+3 is larger than 2+4, `RIGHT` if 2+4 is larger than 1+3, and `EQUAL` if 1+3 weighs the same as 2+4.

Once you've determined the weighted coins, you end interaction by printing `ANS {coin1} {coin2}`. For example `ANS 3 6`.

Constraints

- $1 \leq n \leq 10^5$
- The total number of queries should not exceed 22.

Example interaction

```
[JUDGE]:    8
[PROCESS]:  TEST 1 | 2
[JUDGE]:    EQUAL
[PROCESS]:  TEST 3 4 5 | 6 7 8
[JUDGE]:    EQUAL
[PROCESS]:  TEST 3 | 4
[JUDGE]:    RIGHT
[PROCESS]:  TEST 5 6 | 7 8
[JUDGE]:    LEFT
[PROCESS]:  ANS 4 6
```