Problem A
Compare Suffixes
Time limit: 2 seconds

The *judge program* possesses a hidden string S of length n consisting only of lowercase Latin letters (`a–z`). You cannot access the string. At the start, only the length n is given to you.

Your task is to determine the order of all n suffixes using a limited number of *queries*.

For an integer k ($1 \le k \le n$), let $S(k)$ denote the suffix of S starting at the k-th character. In particular, $S(1) = S$.

In a single query, you specify two distinct integers i and j. The judge program compares $S(i)$ and $S(j)$ lexicographically and returns whether $S(i) < S(j)$ or $S(i) > S(j)$ to you. Note that ties never occur for $i \neq j$ because all suffixes are distinct.

Find a permutation $(p_1, p_2, \ldots, p_n)$ of $(1, 2, \ldots, n)$ such that $S(p_1) < S(p_2) < \cdots < S(p_n)$.

Interaction

The first line of input contains the integer n ($2 \le n \le 1000$).

To issue a query, your program should write a line of the form "`query` i j" ($1 \le i \le n; 1 \le j \le n; i \neq j$). After that, an input line containing `first` or `second` becomes available. A line containing `first` means $S(i) < S(j)$, while a line containing `second` means $S(i) > S(j)$.

Once you determine the order of the suffixes, your program should write a line of the form "`answer` p_1 p_2 $\ldots$ p_n". After that, the interaction stops and your program should terminate without extra output.

Your program may issue at most 6260 queries. If your program issues more than 6260 queries, it will be judged as "Wrong Answer."

It is guaranteed that the hidden string S consists only of lowercase letters (`a–z`).

Notes on interactive judging:

- The evaluation is non-adversarial, meaning that the string S is chosen in advance rather than in response to your queries.
- Do not forget to flush output buffers after writing. See the "Judging Details" document for details.
- You are provided with a command-line tool for local testing, together with input files corresponding to the sample interactions. You can download these files from DOMjudge. The tool has comments at the top to explain its use.

Read	Sample Interaction #1	Write
4		
	query 2 1	
first		
	query 2 4	
second		
	query 1 3	
first		
	answer 4 2 1 3	

Explanation for the sample interaction #1

In this sample, $S =$ `icpc` is assumed. The order of four suffixes is $S(4) < S(2) < S(1) < S(3)$ because `c` $<$ `cpc` $<$ `icpc` $<$ `pc`.

In the first and third query, `first` is returned because $S(2) < S(1)$ and $S(1) < S(3)$. In the second query, `second` is returned because $S(2) > S(4)$. With these responses, you can determine the ordering.