Problem C. Corrupted Sort

Time limit:	2 seconds
Memory limit:	512 megabytes

This is an interactive problem.

Chloe wants to test her sorting skills. She has n cards with distinct integers from 1 to n written on them. She asks her little brother Connor to first blindfold her and then arrange all cards in a row in some order. Positions of cards are numbered from 1 to n from left to right.

Chloe doesn't know the order of the cards, but she wants to sort them, so that the leftmost card has number 1 and the rightmost card has number n on it. Formally, for each i she wants the card on position i to have number i on it. To achieve the goal, Chloe can ask Connor to do one or more operations.

Each operation can be denoted by two integers pos_i and pos_j ($1 \le pos_i < pos_j \le n$). Connor looks at the cards on positions pos_i and pos_j, and if the card on position pos_i has a bigger number than the card on position pos_j, he swaps them. Otherwise, he does nothing. Connor also tells Chloe if he swapped the cards or not.

To make the game more interesting, after every $2n$ operations Connor chooses two distinct cards uniformly at random and swaps them without telling anything to Chloe.

If after some of Chloe's operations all cards become sorted, Chloe wins. Help Chloe to sort all cards using at most 10 000 operations.

Interaction Protocol

Your program should start by reading a single integer n ($2 \le n \le 50$) — the number of cards.

Your program can ask to do the specified operation one or more times. To do the operation, your program should print its description formatted as "pos_i pos_j" — two distinct numbers denoting the positions of cards ($1 \le pos_i < pos_j \le n$). Remember to flush the output after printing each operation description. Then your program should read the outcome of the operation — it will be a single word `SWAPPED` if the cards were swapped, or a single word `STAYED` if nothing changed.

If instead of an operation outcome your program receives a single word `WIN`, it means that the cards have become sorted. Your program should terminate silently after that, it is not allowed to output any extra operations.

After every $2n$ operations two distinct cards are chosen randomly and swapped without telling anything to your program. Each pair of cards has equal probability to be chosen. This swap is not counted as an operation.

Example

standard input	standard output
3 SWAPPED STAYED WIN	 1 2 1 3 2 3

Note

Initial card ordering in the example was 3 1 2 (that is, the card on position 1 had number 3 on it, the card on position 2 had number 1, and the card on position 3 had number 2).