Problem 3 - Bitwise 1024

Stamat loves playing the 1024 game (http://1024game.org/). So he decided to make his own version of the game. He recently started learning programming from the C Programming course at Software University and therefore he believes he has what it needs to make his very own game. Well, why would you care, you might ask. Unluckily for you, you have been selected as one of Stamat's teammates for your first teamwork assignment at the university and since he is a cheeky guy, he has convinced you to create his game of choice. What's more, you have less than 6 hours until the teamwork defense is due to take place and you have to at least come up with something working.

Well, Stamat has given you instructions on how the game works (You got to give him 100 points in the teamwork feedback afterwards for him being so useful, right?). What's more, he decided you made a bitwise version of the game called Bitwise 1024.

You are given as input **N** numbers.

Afterwards, you are given a series of commands on the standard input which could be either **left or right** and a position to start from(since the game doesn't implement the full version with commands for up and down, you decided to name it bitwise 1024 instead of 2048)**.** Given the command, you should move all the bits from the given position (inclusively) from all the numbers in the direction of the command (The zero bits count as empty spaces).

The game should stop when the command **end** is received. Afterwards, you should print the resulting numbers on the standard output.

Input

The input will be read from the standard input.

- On the **first line** you will be given an integer **N** – **the number of input numbers.**
- On the next **N** lines, you will be given numbers to be processed as a game field.
- On the next lines, you will receive commands which determine how the state of the board should change. They will be either **left** or **right**.
- When you receive the command **end**, you have to stop the game and process the output.

The input will always be valid and in the format described, there is no need to check it explicitly.

Output

The output should be printed on the standard output.

- You should print the state of the input numbers after all the commands have been processed

Constraints

- The number N will be in the range [1 … 20].
- The input numbers will be in the range [0 … 2^{63}]
- The number of **commands** will be in the range [0 … 30].
- Allowed working time: 0.1 seconds. Allowed memory: 16 MB.

Examples

Input	Output	Comments
1 7 left 1 right 63 end	7	0000000000000000000000000000000000000000000000000000000000000111 ↓ 1100000000000000000000000000000000000000000000000000000000000001 ↓ 0000000000000000000000000000000000000000000000000000000000000111

Input	Output	Comments
3 23 54 17 left 13 left 4 right 60 left 9 end	9223372036854775815 13835058055282163715 9223372036854775809	