Competitive Programming Contest: Linked List

A. The Headset Challenge

1 second, 256 megabytes

At long last, the Apple Vision Pro headset has launched in Malaysia after a long wait! However, Apple cannot supply all of their stock of headsets in the country right now due to the extremely high demand in other countries worldwide.

To combat the supply issue, in addition to the steep price for the novel tech headset and the headset being purchasable in-store only, they have devised an extra step or challenge for those really keen on buying their latest product: there is an odd-looking, obfuscated string that is given out to customers at each store that, at first, appears meaningless to the naked eye. However, dedicated Apple customers will realize that they must first unobfuscate the string to obtain a keycode within, which can be entered onto a keypad available at every Apple store.

Of course, the company would not like this challenge to horribly cannibalize their revenues, so they have provided a few tips on how to solve the code: customers are allowed to try their luck by attempting to enter the keycode on a store's keypad directly. Some characters within the string indicate that the customer must press the backward cursor button, the forward cursor button, or the Backspace button on the keypad to move the keypad cursor along the characters or erase characters in the keycode.

Are you able to step up to the challenge to solve and obtain the keycode to obtain exclusive access to purchase the brand new Vision Pro headset?

Input

The input only consists of one line, containing a string of length n ($1 \le n \le 10^6$). The true keycode of an Apple store only consists of lowercase letters and digits. The $<$, $>$, and X characters in the obfuscated string represent the backward cursor button, forward cursor button, and Backspace button on the keypad respectively.

If the keypad cursor is already at the start of the string and the backward cursor button ($<$) is pressed, or the keypad cursor is already at the end of the string and the forward cursor button ($>$) is pressed, there are no effects to the cursor's position, meaning that the keypad cursor will not move.

The $<$ and X characters do not appear as the first character in the obfuscated string.

Additionally, $>$ also does not appear as the last character in the string.

Output

Print one line that contains the keycode obtainable from the obfuscated string.

Scoring

Group	Points	Constraints
1	8	$n \le 5 \cdot 10^5$, X is not in the string
2	12	No additional constraints

input

```
m<t<k<s<f
```

output

```
fsktm
```

input

```
pple<<XXapp
```

output

```
apple
```

input

```
keycode1984
```

output

```
keycode1984
```

In the first example, the repeated $<$ characters meant that the keypad cursor is moved backwards (towards the start of the string) each time a letter was typed, resulting in a reversed order compared to the original string.

In the second example, the $<$ and X characters moved the cursor backwards (towards the start of the string) by two characters and erased two characters after, leading the characters "pp" to be erased, continued with the insertion of the characters "app" in the last segment of the obfuscated string.

In the third example, neither $<$, $>$, nor X appear in the string. Thus, the output string remains the same as the input string.

B. Magician's Shuffle

5 seconds, 256 megabytes

A magician has a deck of n cards. However, before performing his magic, he needs to shuffle the cards according to the numbers on the cards in non-descending order from top to bottom.

Now, he picks m audience members from his crowd to "cut" the deck from the ath card to the bth and move the entire segment below the cth card.

Then, after finishing all the cuts, m audience members are asked to place a card with a value o of their choice into the deck below the pth card of their choice.

Through his careful manipulation of the cards, he performs his magic flawlessly and is able to guess the arrangement of the entire deck. You, as his curious audience, are wondering how this could be solved using coding skills even if you do not have a good memory.

Input

The first line of input contains an integer n ($0 \le n \le 2 \cdot 10^7$) — the number of cards in the deck.

The next line contains n integers, k_i ($-10^6 \le k_i \le 10^6$) — the number on each card, k_i.

The third line contains a single integer, m ($0 \le m \le 2 \cdot 10^3$) — the number of audience members picked to cut the deck.

The following m lines contain 3 integers a, b, and c ($0 \le a \le b < n$), ($0 \le c < a$ or $b < c < n$), where a is the starting index of the segment, b is the ending index of the segment, and c is the target index where the segment should be placed under.

The last m lines contain 2 integers o ($-10^6 \le o \le 10^6$) — the number on the to-be-inserted card and p ($0 \le p \le 10^5$) — the pth card to insert below.

Output

A single line of integers representing the final shuffled cards.

If there is no card to shuffle, print "there is no card".

Scoring

Group	Points	Constraints
1	4	$1 \le n \le 20$, $m = 0$, $-25 \le k_i \le 25$

2	9	$n \le 1000$, $1 \le m \le 50$, $-1000 \le k_i \le 1000$
3	17	No additional constraints

input

```
0
0
```

output

```
there is no card
```

input

```
8
1 4 2 3 2 2 1 1
3
1 2 3
0 2 4
2 2 6
0 0
5 8
1 1
```

output

```
0 1 1 2 2 1 2 3 1 5 4
```

input

```
6
2 1 3 4 5 6
2
0 2 4
1 1 5
100 0
99 6
```

output

```
100 4 1 2 3 6 99 5
```

input

```
5
-1 5 12 3 -2
0
```

output

```
-2 -1 3 5 12
```

In the second example test case, first we sort the deck and we will get $[1, 1, 1, 2, 2, 2, 3, 4]$.

Then, $a = 1$, $b = 2$, $c = 3$ (shift cards from index 1 — 2 to below 3rd card)

$$[1, 2, 1, 1, 2, 2, 3, 4]$$

Next, $a = 0$, $b = 2$, $c = 4$

$$[1, 2, 1, 2, 1, 2, 3, 4]$$

Finally, $a = 2$, $b = 2$, $c = 6$ (shift card at index 2 to below 6th card)

$$[1, 2, 2, 1, 2, 3, 1, 4]$$

The segment of the cards always lands after the target index. For example, it is possible to move a segment to the very end of the deck, but not to the very beginning of the deck (since the target index of 0 will land the segment directly after the first number).

C. Pam's Party Game

2 seconds, 256 megabytes

Hey fellow adventurers! You won't believe what's happening right now — Pam's throwing this epic party in a top-secret location. It's like something out of a treasure hunt, full of mystery and excitement! And get this, there are a bunch of people in costumes hitting the dance floor, ready to spice up the night.

But hold on tight because Pam, being the fearless leader she is, has decided to kick off her own party game. It's time to shake things up with Pam's Party Game!

So, she gathers all the folk, gets them to pair up into N pairs, and lines them up, setting the stage for some serious fun.

Imagine this: each pair standing side by side in the queue, just like old pals. Think Emma and Hank, then Duke and Oliver, all queued up and ready to roll. Thus the queue from back to front will be Emma, Hank, Duke, and then Oliver.

Then, Pam passes around a mic to the first person at the front of the queue and starts calling out X commands for everyone to follow. And let me tell you, these commands are the key to unlocking a whole new level of adventure:

A: Yell your counterpart's name into the mic. Spread that love loud and clear!

B: Pass the mic to the person in front of you. Keep the energy flowing forward! It is guaranteed such a person exists.

C: Pass the mic to the person behind you. Let's keep this party moving! It is guaranteed such a person exists.

D: Pass the mic behind you, then hustle to the back of the queue. If you're already at the back, just pass it to the one at the front of the queue. It's all about keeping the rotation going!

E: Pass the mic behind you, then move behind your counterpart. It's like a dance move and a mic pass, double the excitement! If there's no one behind you, pass the mic to the person at the front of the queue instead.

And guess what? You're not just a guest — you're the official event logger! Your mission, should you choose to accept it, is to jot down every name that gets shouted during Pam's Party Game and keep track of how the queue ends up after all X commands have been played out. Let's dive into this adventure and make some unforgettable memories!

Input

The first line of input contains two integers N $(1 \le N \le 250,000)$ and X $(1 \le X \le 1,000,000)$, denoting the number of pairs participating and the number of commands respectively.

The next N lines each contain two names, the names of pairs participating in the game. The list of names will be given following the order they will be standing in the queue. Each name will consist of at most 7 lower-case Latin alphabet characters ('a' — 'z'). It is guaranteed that the given names are unique.

The final line will be a string of length X, representing the commands given by Pam. It is guaranteed that the string will only contain the characters A, B, C, D, and E.

Output

For each name yelled, print the name on a separate line in order. Then, print an empty line. Finally, print $2N$ lines, each containing the names of the person in the queue from front to back.

Scoring

Group	Points	Constraints
1	7	There will be no A-type, D-type, or E-type commands.
2	11	There will be no D-type, or E-type commands.
3	6	$N \le 1000$, $X \le 3000$ There will be no E-type commands.
4	6	$N \le 7500$, $X \le 20000$ There will be no E-type commands.
5	10	There will be no E-type commands.
6	5	$N \le 7500$, $X \le 20000$

7	5	No additional constraints

input

```
3 6
amelia bubba
kiryu coco
ollie udin
ACCABA
```

output

```
bubba
coco
amelia

amelia
bubba
kiryu
coco
ollie
udin
```

input

```
3 16
amelia bubba
kiryu coco
ollie udin
CDCADBBADCCCCCDA
```

output

```
kiryu
bubba
coco

kiryu
ollie
udin
bubba
coco
amelia
```

input

```
3 22
amelia bubba
kiryu coco
ollie udin
CDCADBBADCCCCCDAECCEBA
```

output

```
kiryu
bubba
coco
ollie

ollie
udin
coco
kiryu
amelia
bubba
```