

Problem 5 - Purple Cow (100 pts)

Problem Description

There is a mystery in Alley 118.

Never order a medium rare steak in Purple Cow (紫牛).

After celebrating the birthday in *Purple Cow*, USA has a stomachache. He starts to look for a bathroom. However, it is the peak time for bathrooms, and thus there are lots of people waiting in multiple lines. After waiting in a line for a while, USA notices that the number of the people in front of him increases. Afterwards, he discover that they are all impolite USBs who would cut in line and stand after their friends if they have friends from the *same group* that are already in the line.

There are M bathrooms with indices $0, \dots, M - 1$, and K groups of students with indices $0, \dots, K - 1$. Now you, a DSA STUDENT (Super Toilet & Unbelievably Diligent & Extra Noted Technician), would need to help USA simulate how these lines change. The following 4 situations might happen and cause some changes to the lines.

- **enter** i j m

USB with id j from group i comes to stand in line m (the line for bathroom m). Note that the USB is going to cut into the position after the *last* person in group i . If no one in group i is in that line, the USB is going to stand at the end of the line.

- **leave** m

The last USB in line m walks away.

- **go** m

The first USB in line m goes to use bathroom m .

- **close** m

Bathroom m is out of toilet paper and closes. So USBs in that line would move to the *nearest* open bathroom with a smaller index. Since the bathroom is placed circularly for some unknown reasons (probably because the earth is round :-), USBs would search for the first open bathroom in $M - 1, M - 2, \dots$ if all the smaller-indexed bathrooms have been closed.

When moving, USBs from the end of line m would move first because they are closer to the end of the other line. So effectively the order in line m is reversed in the new line. Besides, all those USBs still keep the habit of cutting in like **entering** when moving to another line.

Input Format

The first line contains three integers M , N , K , representing M bathrooms, N situations and K groups. The next N lines are formatted as follows.

- `enter`, followed by three integers i j m separated by spaces
- `leave`, followed by one integer m
- `go`, followed by one integer m
- `close`, followed by one integer m

Output Format

You should output M lines, with line m containing the ids of the USBs waiting for bathroom m by their order in the line.

Constraints

1. $1 \leq M, N, K \leq 10^6$
2. $1 \leq M \cdot K \leq 10^6$
3. every USB's id is *distinct* and ranges in $[1, 10^8]$
4. There will be no invalid situations, such as entering/closing a closed bathroom or leaving/going from an empty line.

Subtasks

Subtask 1 (20 pts)

- $M = 1$
- $N, K \leq 10^3$
- no `close` situation

Subtask 2 (20 pts)

- $1 \leq M, N, K \leq 10^3$
- $1 \leq M \cdot K \leq 10^4$
- no `close` situation

Subtask 3 (20 pts)

- $1 \leq M, N, K \leq 10^3$
- $1 \leq M \cdot K \leq 10^4$

Subtask 4 (40 pts)

- no other constraints

Sample Cases

Sample Input 1

```
1 11 4
enter 0 1 0
enter 1 2 0
enter 0 3 0
enter 1 4 0
enter 2 5 0
enter 1 6 0
leave 0
leave 0
go 0
go 0
enter 0 7 0
```

Sample Output 1

```
2 4 7
```

Sample Input 2

```
3 14 4
enter 0 1 0
enter 1 2 0
enter 0 3 0
enter 2 4 1
enter 1 5 1
enter 0 6 1
enter 1 7 1
enter 3 8 1
enter 1 9 2
enter 0 10 2
close 1
go 0
close 0
go 2
```

Sample Output 2

```
5 7 2 10 6 3 4 8
```

Note that there are two empty lines above 5 7 2 10 6 3 4 8. For more specific explanations, please refer to the demo below.

Sample 2 Demo

enter 0 1 0 & enter 1 2 0 & enter 0 3 0 & enter 2 4 1 & enter 1 5 1

0 | 1, 3, 2,
1 | 4, 5,
2 |

enter 0 6 1 & enter 1 7 1 & enter 3 8 1 & enter 1 9 2 & enter 0 10 2

0 | 1, 3, 2,
1 | 4, 5, 7, 6, 8,
2 | 9, 10,

close 1 & go 0

0 | 1, 3, 2,
1 | 4, 5, 7, 6, 8,
2 | 9, 10,

0 | 3, 6, 2, 7, 5, 8, 4,
1 |
2 | 9, 10,

close 0 & go 2

0 | 3, 6, 2, 7, 5, 8, 4,
1 |
2 | 9, 10,

0 | 3, 6, 2, 7, 5, 8, 4,
1 |
2 | 9, 5, 7, 2, 10, 6, 3, 4, 8,

Hints & Notes

- Closing an empty bathroom is valid situation.
- Moving USBs one by one is possibly too time-consuming.
- All you need are the data structures taught in class. But if you are interested in using even more advanced data structures to solve this problem, consider searching with the following keywords: *reversing doubly-linked list in $O(1)$* .
- Go⊕d Luck & Have Fun!