Friends Two

time limit per test: 1 second (3 seconds for Python)
memory limit per test: 500 megabytes
input: standard input
output: standard output
points: 100 · problem code: friendstwo

There are n people in a class, numbered 1 through n. Friendship is mutual: if a is friends with b, then b is friends with a. Two people are in the same friend group if they are the same person, or if they are connected by a chain of friendships (friends, friends of friends, and so on).

At the start of the year, some friendships exist. Over the year, those friendships break one by one until none remain. You are given a chronological log of q events. Every friendship that ever existed appears in this log as a breakup, so the initial friendships are exactly the pairs that later break.

Process the events in order and answer queries about who is still in the same friend group.

Input

The first line contains two integers n and q, the number of people and the number of events.

The next q lines each describe one event in one of the following forms:

- `1 a b` — the friendship between a and b breaks (and is removed). If that friendship is already gone, nothing happens.
- `2 a b` — ask whether a and b are currently in the same friend group.

Output

For each event of type 2, output one line:

- `Yes` if a and b are in the same friend group,
- `No` otherwise.

Constraints

- $1 \le n \le 10^5$
- $1 \le q \le 5 \times 10^5$
- $1 \le a, b \le n$
- $a \neq b$

Example 1

Input

```
5 10
1 1 2
1 4 5
2 3 4
1 3 4
1 2 5
2 2 5
2 2 5
1 3 5
2 2 4
1 1 5
```

Output

```
Yes
No
No
No
```

Explanation

The breakups in the log are the pairs $(1, 2)$, $(4, 5)$, $(3, 4)$, $(2, 5)$, $(3, 5)$, and $(1, 5)$, so those six friendships exist at the start.

- After $(1, 2)$ and $(4, 5)$ break, 3 and 4 are still friends directly, so the answer is `Yes`.
- After $(3, 4)$ and $(2, 5)$ also break, 2 and 5 are no longer connected, so both queries answer `No`.
- After $(3, 5)$ breaks, 2 and 4 are still not connected, so the answer is `No`.

Example 2

Input

```
10 20
2 3 9
2 6 10
1 3 10
2 4 8
1 1 9
1 2 8
2 6 10
2 4 8
1 2 9
1 6 10
1 5 8
1 2 8
1 5 6
2 7 9
1 8 10
2 7 9
2 2 4
1 3 5
1 5 8
2 1 8
```

Output

```
Yes
Yes
No
Yes
No
No
No
No
No
```