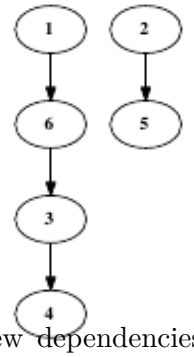
# 4626 Hypervisor MacrOS

Bob is the leader of a team developing a hypervisor system *MacrOS*. The project is huge and comprises lots of packages. Since some of them may depend on others, the installation process of the whole system is rather complicated. MacrOS is in alpha stage now, so there are many customers testing the new product. The installation process is overwhelming for many of them, and unfortunately the technical support is Bob's responsibility as well.




Bob had been given a list of dependencies by his team of programmers before the alpha tests started. Each dependency reads "package $B$ depends on $A$", i.e., one has to install package $A$ before $B$. This is denoted "$A$ $B$" in short. Of course, if package $C$ depends on $B$, and $B$ in turn depends on $A$, then $C$ depends on $A$ as well, i.e., the dependencies are transitive.

For example, the initial list of dependencies can be as follows:

```
1 6, 6 3, 3 4, 2 5
```

As the product develops, the programmers sometimes call Bob to inform of new dependencies. Further, being in charge of technical support, Bob often receives phone calls from customers with questions which packages should be installed first. To no surprise, after several calls from programmers and customers, Bob realized how difficult it is to keep track of dependencies and answer queries on-line at the same time, and. For automatizing this process, he wrote a program which generated two log files. The first one contains the history of all phone calls. An entry '`1` $A$ $B$' denotes a new dependency introduced by the programmers, and '`0` $A$ $B$' denotes a query from a customer meaning "should I install $A$ before $B$?". The second log is a history of all answers given to customers. An example is given in Table 1.

After a long period of testing, MacrOS is finally ready to enter beta stage. But Bob wants to be sure that there were no mistakes during the alpha testing. He wants to check if the answers given in the second log file are correct, and you are to help him. However, Bob does not give you the second log file. Moreover, he modified the first log in a tricky way: after each line corresponding to customer phone call that should have been answered with `NO`, he started/stopped reversing all lines corresponding to programmers calls; see the example below. Taking into account the Bob's modification of the first log file, you should give all the answers to the customers' questions.

| First log file | Second log file (answers) |
|---|---|
| *1* 1 3 | |
| *0* 1 3 | `YES` |
| *1* 1 4 | |
| *0* 5 2 | `NO` |
| *1* 3 2 | |
| *1* 6 5 | |
| *0* 1 5 | `YES` |
| *1* 4 5 | |
| *0* 5 3 | `NO` |
| *1* 1 2 | |

Table 1. The first answer is `YES` because package 1 should be installed before 3.

| Modified first log file |
|---|
| *1* 1 3 |
| *0* 1 3 |
| *1* 1 4 |
| *0* 5 2 — start reversing |
| *1* **2 3** |
| *1* **5 6** |
| *0* 1 5 |
| *1* **5 4** |
| *0* 5 3 — stop reversing |
| *1* 1 2 |

Table 2. Reversed numbers of packages are written in bold.

## Input

The input contains several test cases. The first line of the input contains a positive integer $Z \leq 15$, denoting the number of test cases. Then $Z$ test cases follow, each conforming to the format described below.

Two integers $n$ and $m$ ($1 \leq n \leq 10^5$, $0 \leq m \leq 10^5$) separated by a single space are given in the first line of an input instance. These denote the number of MacrOS' packages and the number of dependencies before alpha tests took place, respectively. The following $m$ lines describe the initial list of dependencies. Each contains two numbers $A$ and $B$ ($1 \leq A, B \leq n$) separated by a single space, denoting that package $B$ depends on $A$. Some dependencies may appear multiple times, but there will be no pair of mutually dependent packages. The next part of the input contains the Bob's modified version of the first log file, with the format described above. The total number of all phone calls is at most $10^5$. The input instance ends with a line containing three zeros, '`0 0 0`'.

## Output

For each test case, your program has to write an output conforming to the format described below.

Your program should print the content of the second log file. This means that for each input line '`0` $A$ $B$', you should print '`YES`' if package $A$ should be installed first and '`NO`' if $B$ should be installed first.

**You can assume that all the queries have a unique answer, i.e., at the moment of such query there was already a dependency between packages $A$ and $B$.**

## Sample Input

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

## Sample Output

```
YES
NO
YES
NO
YES
NO
```