Optimal Marks

You are given an undirected graph G(V, E). Each vertex has a mark which is an integer from the range $[0..2^{31} – 1]$. Different vertexes may have the same mark.

For an edge (u, v), we define Cost(u, v) = mark[u] xor mark[v].

Now we know the marks of some certain nodes. You have to determine the marks of other nodes so that the total cost of edges is as small as possible.

Input

The first line of the input data contains integer **T** (1 ≤ **T** ≤ 10) - the number of testcases. Then the descriptions of T testcases follow.

First line of each testcase contains 2 integers **N** and **M** (0 < **N** <= 500, 0 <= **M** <= 3000). **N** is the number of vertexes and **M** is the number of edges. Then **M** lines describing edges follow, each of them contains two integers u, v representing an edge connecting u and v.

Then an integer **K**, representing the number of nodes whose mark is known. The next **K** lines contain 2 integers u and p each, meaning that node u has a mark p. It's guaranteed that nodes won't duplicate in this part.

Output

For each testcase you should print **N** lines integer the output. The **K**th line contains an integer number representing the mark of node **K**. If there are several solutions, you have to output the one which minimize the sum of marks. If there are several solutions, just output any of them.

Example

Input:
1
3 2
1 2
2 3
2
1 5
3 100

Output:
5
4
100