Ab Initio

"A journey of a thousand miles begins with a single step." -Lao Tzu.

When someone is studying competitive programming from the beginning, he or she is said to be a student of competitive programming ab initio. Students should be given relatively easy problems to begin their journeys, before they are gradually given more difficult problems.

Here is an example of an easy problem that does not even need a description.

Input

The first line of input contains three integers $V(2 \leq V \leq 2000)$, $E(1 \leq E \leq 200000)$, and $Q(1 \leq Q \leq 2000)$, the number of vertices, the number of edges and the number of queries that must be performed on a directed, unweighted graph G, respectively. For simplicity, we label the vertices $0, 1, 2, ..., V-1$.

The next E lines describe the edges of G, given in the format of an edge list. In particular, the i^{th} of these lines contains two integers A_i and $B_i(0 \leq Ai, Bi < V; Ai \neq Bi)$, denoting that the ith edge connects vertex A_i to vertex B_i. It is guaranteed that for any pair of vertices (a, b), there is at most one edge from a to b. The next Q lines contain the queries. In particular, the i^{th} of these lines begins with a single integer:

- If this integer is 1, no integers follow.

  You should add a new vertex labeled V to G. This vertex should not have edges to or from any other vertex. V - the current size of G - now increases by 1.

- If this integer is 2, two integers X_i and $Y_i(0 \leq X_i, Y_i < V; X_i \neq Y_i)$ follow.

  You should add a new directed edge connecting vertex X_i to vertex Y_i in G. It is guaranteed that this edge does not currently exist.

- If this integer is 3, a single integer $X_i(0 \leq X_i < V)$ follows.

  You should delete all the incoming and outgoing edges of X_i from the graph G.

- If this integer is 4, two integers X_i and $Y_i(0 \leq X_i, Y_i < V; X_i \neq Y_i)$ follow.

  You should remove the directed edge connecting vertex X_i to vertex Y_i from G. It is guaranteed that this edge currently exists.

- If this integer is 5, no integers follow.

  You should replace G with its transpose G', defined as follows: For every pair of vertices (a, b), the edhe from a to b exists in G' if and only if $a \neq b$ and the edge from a to b does not exist in G.

- If this integer is 6, no integers follow.

  You should replace G with its complement $\bar{G}$, defined as follows: For every pair of vertices (a, b), the edge from a to b exists in $\bar{G}$ if and only if $a \neq b$ and the edge from a to b does not exist in G.

Output

After performing all the queries, you will have a final graph G. Since this graph can be very large, we will not ask you to output the entire graph. Instead, by doing the below, you can convince us that you indeed have the required graph.

1. On the first line, output a single integer V, the number of vertices in the graph G.
2. For each of the next V lines, output two integers. In particular, in the i^{th} of these lines, output:
    1. d_i, the outdegree of vertex i, and
    2. h_i, the *hash* of the *adjacency* list of vertex i, defined as follows. Suppose the vertices in the out-neighborhood of vertex i are $n_1 < n_2 < ... < n_d$. Then

    $$h_i = 7^0 \cdot n_1 + 7^1 \cdot n_2 + 7^2 \cdot n_3 + ... + 7^{d_i - 1} \cdot n_{d_i}$$

    Since h_i can be quite large, you should output only the remainder after dividing this number by $10^9 + 7$.

Sample input	**Sample output**
`3 2 8`	`4`
`0 1`	`3 162`
`0 2`	`3 161`
`2 1 2`	`2 21`
`1`	`2 15`
`2 3 0`	
`4 0 2`	
`3 0`	
`2 0 3`	
`6`	
`5`	