Pointers

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

You are given a connected undirected graph with n nodes and m edges. Each node u has an ordered list l_u of its neighbors, and an arrow pointing to one of its neighbors p_u. Initially, p_u is the first neighbor in l_u.

You start at node s, and repeat the following process infinitely many times:

1. Let v be the node at which you are currently located. Move from v to p_v.
2. Increment p_v to the next neighbor in l_v cyclically.

See the sample notes for an example of this process.

Consider the list $p_1, p_2, \cdots p_n$ over the course of this process, as well as the current node c. We call this a "state".

Print any state that appears an infinite amount of times.

Input

The first line of the input contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains three integers n, m, and k ($1 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 4 \cdot 10^5$, $1 \le s \le n$) — the number of vertices and edges in the graph, and the starting node, respectively.

The u-th of the next n lines describes the ordered list l_u of neighbors of u. It begins with an integer k_u ($1 \le k_u < n$) — the number of neighbors of u. This is followed by k_u distinct integers $v_1, v_2, \cdots v_{k_u}$ ($1 \le v_i \le n$, $v_i \ne u$) — the neighbors of u.

It is guaranteed that if v is a neighbor of u, then u is a neighbor of v. It is also guaranteed that there are m undirected edges in total.

Across all test cases, it is guaranteed that the sum of n is at most $2 \cdot 10^5$, and the sum of m is at most $4 \cdot 10^5$.

Output

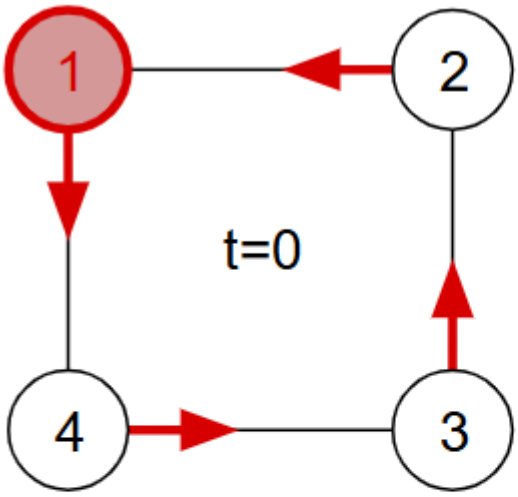
For each test case print any state that repeats infinitely in the format c p_1 p_2 ... p_n.

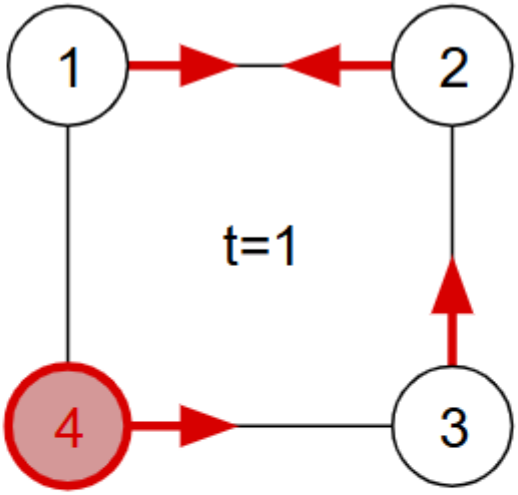
Example

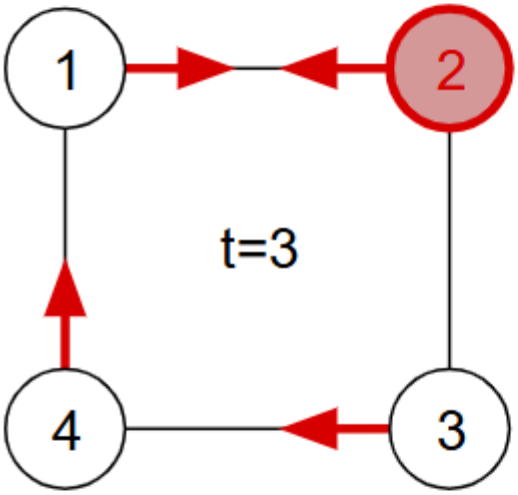
standard input	standard output
3 4 3 2 1 4 1 3 2 4 2 2 3 1 4 3 3 1 4 1 3 2 4 2 2 3 1 4 4 1 2 4 2 2 1 3 2 2 4 2 3 1	3 4 3 2 1 3 4 3 2 1 1 4 1 2 3

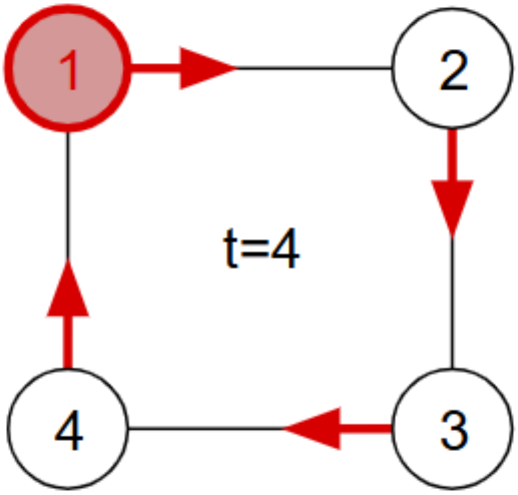
Note

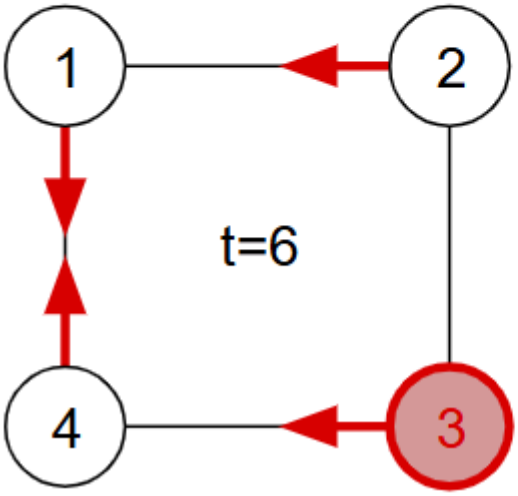
Let's visualize the third sample case. The red node represents your current position, and the arrow pointing out from each node u points at node p_u. Here is how the graph looks at the start of the process, and after each of the next 8 steps:

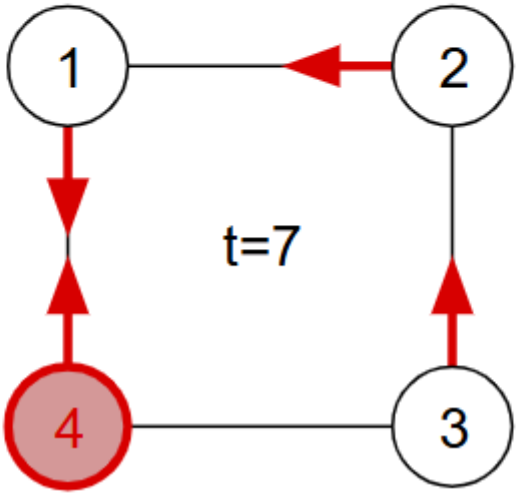












We can see that after 8 operations have been performed, we have reached the initial state

$p_1 = 4, p_2 = 1, p_3 = 2, p_4 = 3$ once again, and our current location (node 1) is the same as it was at the beginning. Therefore, a valid answer is to simply print the initial state.