Problem G. Graph

Input file:	`graph.in`
Output file:	`graph.out`
Time limit:	2 seconds
Memory limit:	256 megabytes

The sequence $a_1, a_2, \ldots, a_n$ is called a *permutation*, if it contains every integer from 1 to n.

The permutation of vertices $a_1, a_2, \ldots, a_n$ is a *topological sort* of a directed graph, if for every directed edge from u to v, vertex u comes before v in this permutation.

The permutation $a_1, a_2, \ldots, a_n$ is *lexicographically smaller* than the permutation $b_1, b_2, \ldots, b_n$, if there exists m such that $a_i = b_i$ for every $1 \le i < m$ and $a_m < b_m$.

Given a directed acyclic graph, add at most k directed edges to it in such a way, that the resulting graph still has no cycles and the lexicographically minimal topological sort of the graph is *maximum possible*.

Input

The first line of the input file contains three integers n, m and k — the number of vertices and directed edges in the original graph, and the number of directed edges, that you are allowed to add ($1 \le n \le 100\,000$; $0 \le m, k \le 100\,000$).

Each of the following m lines contains two integers u_i, v_i, describing directed edge from u_i to v_i ($1 \le u_i, v_i \le n$).

The graph has no cycles.

Output

The first line of the output file should contain n integers — the lexicographically minimal topological sort of the modified graph. The second line should contain a single integer x ($0 \le x \le k$) — the number of directed edges to add. The following x lines of the output should contain description of added directed edges in the same format as in the input file.

Examples

graph.in	graph.out
5 3 2 1 4 4 2 1 3	5 1 4 2 3 2 4 3 5 1
2 2 20 1 2 1 2	1 2 1 1 2