Problem D: Precedence graph

Advanced Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

A friend of yours is currently in the process of creating his own operating system. He recently told you in confidence how he can't seem to find a good way to handle the scheduling of codependent processes. So far, he is already able to let the processes in question create a precedence graph (i.e. a graph in which the vertices are the processes and an edge from vertex a to vertex b denotes that process a has to be done before process b can be started) but could not yet figure out an efficient way to find an order of computation that is consistent with this graph.

Back when he told you this you had no idea on how to solve it either, but in the latest *Advanced Algorithms for Programming Contests* lecture you learned about an algorithm that seems to exist for this exact purpose, so you decided to surprise your friend by using it to write a program that solves his problem.

Input

The input consists of

- one line containing N ($2 \leq N \leq 10^4$) and M ($1 \leq M \leq 10^5$) – the number of processes and the number of dependencies between them, respectively
- M lines each containing PIDs a_i and b_i ($1 \leq a_i, b_i \leq n$), denoting that process a_i needs to be done computing before b_i may start.

Output

Output the PIDs $(1, ..., N)$ separated by spaces in an order that is consistent with the given precedence graph. If there are multiple solutions, assume that processes with smaller PID have higher priority and should therefore be first in the list (give back the list that is smallest lexicographically).

Sample input and output

Input	Output
4 3 1 3 1 2 4 2	1 3 4 2