Monopoly Battle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Alice and Bob are engaged in a fierce game of Monopoly Battle, a strategic board game featuring N cities, numbered from 1 to N. At the start of the game (turn 0), there are no paths connecting any of the cities, so Bob, who starts in City 1, cannot move to any other city.

Alice building roads while Bob strategizes with his magical dice. (Image generated by ChatGPT-4o)

However, Alice, being the mastermind of the game, has the ability to build roads between cities. Each turn, Alice decides which cities to connect. On turn t, she constructs a road between City u_t and City v_t, enabling movement from u_t to v_t. (Sometimes Alice gets bored and builds a road that connects a city to itself, i.e., $u_t = v_t$.)

Bob has a special power — he can control his dice roll to show exactly L. But beware! This isn't your typical six-sided die — Bob's magical dice can roll numbers as large as 10^9. That's right, Bob can roll impossibly large numbers, making any normal Monopoly player green with envy. Despite this, Bob must follow one strict rule: he can only move between cities by traveling along exactly L roads, no more and no less.

Bob needs to plan his moves carefully. After each roll of his magical dice, he wants to know which cities he can reach from City 1 by traveling along exactly L roads. He can pick any available route, but he must use precisely L roads.

Your task is to help Bob figure out, for each city $i = 1, 2, \ldots, N$, the earliest turn when Alice's roads allow him to reach that city by traveling exactly L roads. If Bob can't reach a city with exactly L roads, print -1 for that city.

Input

The first line contains three integers: N (the number of cities), T (the number of turns), and L (the number of roads Bob needs to travel).

The next T lines each contain two integers, u_t and v_t, indicating that on the t-th turn, Alice builds a road between City u_t and City v_t.

Constraints

- $2 \leq N \leq 100$
- $1 \leq T \leq N^2$
- $1 \leq L \leq 10^9$
- $1 \leq u_t, v_t \leq N$
- For $i \neq j$, $(u_i, v_i) \neq (u_j, v_j)$

Output

Output N integers. For each city i, print the earliest turn when Bob can reach the city by traveling exactly L roads. If city i cannot be reached with exactly L moves, print -1 instead.

Scoring

There are 3 subtasks for this problem.

Subtask	**Additional constraints**		**Points**	**Required subtasks**
	N, T	L		
1	$T \leq 100$, $N \leq 100$	$L \leq 10^3$	30%	—
2	$N \leq 100$, $T \leq N^2$	$L \leq 10^3$	40%	1
3	$N \leq 100$, $T \leq N^2$	—	30%	1, 2

Examples

standard input	standard output
4 5 3 2 3 3 4 1 2 3 2 2 2	-1 4 5 3
2 1 1000000000 1 2	-1 -1

Monopoly Battle 2

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

Alice and Bob are still at it in their never-ending Monopoly Battle. This time, things have gotten even more interesting. After Bob spent countless hours reading through some ancient magical books, he has unlocked a new power: he can control his dice rolls to be **any** number he wants — as large as he likes, without any upper limit!

But that's not all. Now, Bob can start his journey from **any** city, not just City 1! Despite his newfound magical powers, Bob can still only travel between cities if there are roads connecting them — and Alice is responsible for building those roads.

In this version of the game, the roads Alice builds are **bidirectional**, meaning Bob can travel back and forth between any two connected cities. Each turn, Alice builds a new road connecting two cities, and Bob wants to figure out when he can travel from a starting city s to a target city t for the first time, using any dice roll he likes.

Your task is to help Bob determine, for each pair of cities s and t, the earliest turn they become connected by the bidirectional roads that Alice builds.

Input

The first line contains three integers N, T, and Q: the number of cities, the number of roads Alice will build, and the number of city pairs Bob needs to figure out.

The next T lines describe the bidirectional roads that Alice builds, one per turn, in the order they are constructed. Each line contains two integers u_t and v_t, meaning that on turn t, Alice builds a road between City u_t and City v_t, allowing travel in both directions.

The final Q lines each contain two integers s and t, representing the starting city and the target city. For each pair, Bob needs to figure out the number of turns after which the two cities become connected, or determine if they never get connected.

Constraints

- $1 \leq N, T, Q \leq 2 \cdot 10^5$
- $1 \leq u_t, v_t, s, t \leq N$

Output

For each pair of cities, print the number of turns after which they become connected. If the cities are never connected, print -1.

Scoring

There are 3 subtasks for this problem.

Subtask	Additional constraints		Points	Required subtasks
	Q	T		
1	$Q = 1$	—	50	—
2	$Q \leq 1000$	$T \leq 1000$	20	—
3	—	—	30	1, 2

Example

standard input	standard output
5 4 3 1 2 3 5 2 4 1 5 1 3 3 4 3 5	4 4 2