A. Maze

time limit per test: 2 seconds

memory limit per test: 256 megabytes

Pavel loves grid mazes. A grid maze is an $n \times m$ rectangle maze where each cell is either empty, or is a wall. You can go from one cell to another only if both cells are empty and have a common side.

Pavel drew a grid maze with all empty cells forming a connected area. That is, you can go from any empty cell to any other one. Pavel doesn't like it when his maze has too little walls. He wants to turn exactly k empty cells into walls so that all the remaining cells still formed a connected area. Help him.

Input

The first line contains three integers n, m, k ($1 \le n, m \le 500$, $0 \le k < s$), where n and m are the maze's height and width, correspondingly, k is the number of walls Pavel wants to add and letter s represents the number of empty cells in the original maze.

Each of the next n lines contains m characters. They describe the original maze. If a character on a line equals ".", then the corresponding cell is empty and if the character equals "#", then the cell is a wall.

Output

Print n lines containing m characters each: the new maze that fits Pavel's requirements. Mark the empty cells that you transformed into walls as "X", the other cells must be left without changes (that is, "." and "#").

It is guaranteed that a solution exists. If there are multiple solutions you can output any of them.

Examples

input
```
3 4 2
#..#
..#.
#...
```

output
```
#.X#
X.#.
#...
```

input
```
5 4 5
#...
#.#.
.#..
...#
.#.#
```

output
```
#XXX
#X#.
X#..
...#
.#.#
```

B. Planar Graph

Time limit:	5 seconds
Memory limit:	256 megabytes

Given an undirected graph $G = (V, E)$ with n vertices and m edges which contains a Hamilton cycle $(c_1, c_2, \ldots, c_n)$, test whether G is a planar graph or not.

Recall that a Hamilton cycle $(c_1, c_2, \ldots, c_n)$ in an undirected graph $G = (V, E)$ satisfies

1. Each vertex $v \in V$ appears exactly once in $(c_1, c_2, \ldots, c_n)$;
2. $(c_1, c_2), (c_2, c_3), \ldots, (c_{n-1}, c_n), (c_n, c_1) \in E$.

Moreover, an undirected graph G is planar if it can be drawn on the plane in such a way that its edges intersect only at their endpoints. (By way of a hint, it might be useful to review Euler's Formula for planar graphs.)

Input

The first line contains an integer T, which is the number of test cases.

For each test case, the first line contains two integers n and m. Each of the next m lines contains two integers u and v such that $1 \leq u, v \leq n$ which correspond to an edge in the graph. The last line contains n integers $1 \leq c_1, c_2, \ldots, c_n \leq n$ which describes the Hamilton cycle.

It is guaranteed that $T \leq 100$, $3 \leq n \leq 200$, $m \leq 10000$, and there are no multiple edges or self-loops in the input graph G.

Output

For each test test case, output "YES" if the given graph is planar, and "NO" otherwise.

Example

standard input	standard output

```
2
6 9
1 4
1 5
1 6
2 4
2 5
2 6
3 4
3 5
3 6
1 4 2 5 3 6
5 5
1 2
2 3
3 4
4 5
5 1
1 2 3 4 5
```
	```
	NO
	YES
	```

C. Edge Assignment

Time limit:	2 seconds
Memory limit:	256 megabytes

Given an undirected graph $G = (V, E)$ with n vertices and m edges, assign each edge to one of its two endpoints, such that for any vertex $v \in V$, there is at most one edge assigned to V.

Count the number of assignments modulo 1000000007.

Input

The first line contains two integers n and m. It is guaranteed that $n, m <= 200000$.

Each of the next m lines contains two integers u and v such that $1 \leq u \neq v \leq n$ which correspond to an edge in the graph.

Output

The number of assignments modulo 1000000007.

Example

standard input	standard output

5 4
1 2
3 2
4 5
4 5
	6

Note

For the example input, the six assignments are $\{2, 3, 4, 5\}$, $\{2, 3, 5, 4\}$, $\{1, 3, 4, 5\}$, $\{1, 3, 5, 4\}$, $\{1, 2, 4, 5\}$, $\{1, 2, 5, 4\}$. Here the i-th number is the vertex that the the i-th edge is assigned to.

D. Edge Removal

Time limit:	2 seconds
Memory limit:	256 megabytes

Given an undirected graph $G = (V, E)$, where $V = \{1, 2, \ldots, n\}$ and $|E| = m$. For each vertex $v \in V$, its importance is defined to be the length of the (unweighted) shortest path from 1 to v. If there exists no path from 1 to v, then the importance of v is defined to be $+\infty$.

For each edge $e \in E$, we say e is redundant if after removing e, the importance of all vertices in $\{2, 3, \ldots, n\}$ is unchanged or increased by at most 1.

Output all redundant edges.

Input

The first line contains two integers n and m. It is guaranteed that $n, m <= 200000$.

Each of the next m lines contains two integers u and v such that $1 \leq u \neq v \leq n$ which correspond to an edge in the graph. It is guaranteed that there are no multiple edges in the graph.

It is also guaranteed that the importance of all vertices in the given graph are not $+\infty$.

Output

Multiple lines with increasing integers where each line contains an integer in $\{1, 2, \ldots, m\}$, which are the number of all reductant edges in the given graph.

If all edges are not redundant, output "NO".

Examples

standard input	standard output

5 6
1 2
1 3
1 4
3 4
2 5
3 5
	2
	3
	4
	5
	6

2 1 1 2	NO

E. Underground Lab

time limit per test: 1 second

memory limit per test: 256 megabytes

The evil Bumbershoot corporation produces clones for gruesome experiments in a vast underground lab. On one occasion, the corp cloned a boy Andryusha who was smarter than his comrades. Immediately Andryusha understood that something fishy was going on there. He rallied fellow clones to go on a feud against the evil corp, and they set out to find an exit from the lab. The corp had to reduce to destroy the lab complex.

The lab can be pictured as a connected graph with n vertices and m edges. k clones of Andryusha start looking for an exit in some of the vertices. Each clone can traverse any edge once per second. Any number of clones are allowed to be at any vertex simultaneously. Each clone is allowed to stop looking at any time moment, but he must look at his starting vertex at least. The exit can be located at any vertex of the lab, hence each vertex must be visited by at least one clone.

Each clone can visit at most $\lceil \frac{2n}{k} \rceil$ vertices before the lab explodes.

Your task is to choose starting vertices and searching routes for the clones. Each route can have at most $\lceil \frac{2n}{k} \rceil$ vertices.

Input

The first line contains three integers n, m, and k ($1 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 2 \cdot 10^5$, $1 \le k \le n$) — the number of vertices and edges in the lab, and the number of clones.

Each of the next m lines contains two integers x_i and y_i ($1 \le x_i, y_i \le n$) — indices of vertices connected by the respective edge. The graph is allowed to have self-loops and multiple edges.

The graph is guaranteed to be connected.

Output

You should print k lines. i-th of these lines must start with an integer c_i ($1 \le c_i \le \lceil \frac{2n}{k} \rceil$) — the number of vertices visited by i-th clone, followed by c_i integers — indices of vertices visited by this clone in the order of visiting. You have to print each vertex every time it is visited, regardless if it was visited earlier or not.

It is guaranteed that a valid answer exists.

Examples

input
```
3 2 1
2 1
3 1
```

output
```
3 2 1 3
```

input
```
5 4 2
1 2
1 3
1 4
1 5
```

output
```
3 2 1 3
3 4 1 5
```

Note

In the first sample case there is only one clone who may visit vertices in order (2, 1, 3), which fits the constraint of 6 vertices per clone.

In the second sample case the two clones can visited vertices in order (2, 1, 3) and (4, 1, 5), which fits the constraint of 5 vertices per clone.