Harbour - DFS

A. Number of Edges

2 seconds, 256 megabytes

You are given a simple undirected graph. It is represented by its adjacency matrix.

Your task is to find the number of edges in the graph.

Input
The first line of the input contains one integer n ($1 \le n \le 1000$) — the number of vertices in the graph.

The next n lines contain n integer each. The j-th integer in the i-th line is $g_{i,j}$ is either 0 (if there is no edge between vertices i and j) or 1 otherwise.

It is guaranteed that the given graph is simple and undirected.

Output
Print one integer — the number of edges in the given graph.

input
4 0 0 1 1 0 0 1 0 1 1 0 1 1 0 1 0
output
4

input
2 0 0 0 0
output
0

B. Print Adjacency Lists

2 seconds, 256 megabytes

You are given a simple directed graph. It is represented by its edges list.

Your task is to print an adjacency list for each vertex from 1 to n.

Recall that an adjacency list for the vertex v is the list of vertices connected directly with the vertex v by an edge.

Input
The first line contains two integers n and m ($2 \le n \le 10^5; 1 \le m \le min(\frac{n(n-1)}{2}, 10^5)$) — the number of vertices and the number of edges in the graph.

The next m lines describe edges. The i-th edge is given as two integers x_i and y_i ($1 \le x_i, y_i \le n; x_i \ne y_i$). The edge is directed from the vertex x_i to the vertex y_i.

It is guaranteed that the given graph is simple and directed.

Output
Print n lines. In the i-th line, print the adjacency list of the vertex i in **sorted** order. If some vertex is not connected to any other vertex, leave the corresponding line empty.

input
5 7 1 3 1 2 3 5 2 4 5 1 3 4 2 5
output
2 3 4 5 4 5 1

C. Transitive Closure

2 seconds, 256 megabytes

You are given a simple undirected graph. It is represented by its edges list.

Imagine the following infinite (actually, finite) process: if there is an edge between vertices a and b, and also an edge between vertices b and c, the edge between vertices a and c is added to the graph.

Your task is to calculate the number of edges that will be added to the graph.

Input
The first line of the input contains two integers n and m ($1 \le n \le 2 \cdot 10^5, 0 \le m \le 2 \cdot 10^5$) — the number of vertices and the number of edges in the graph, respectively.

The next m lines describe edges. The edge i is given as a pair of vertices x_i, y_i ($1 \le x_i, y_i \le n, x_i \ne y_i$).

It is guaranteed that the given graph is simple and undirected.

Output
Print one integer — the number of edges that will be added to the graph during the process described in the problem statement.

input
7 5 1 2 2 3 4 5 5 6 6 7
output
4

input
6 5 1 2 2 3 3 4 4 5 5 6
output
10

input
3 0
output
0

In the first example, edges $(1, 3)$, $(7, 4)$, $(4, 6)$ and $(5, 7)$ will be added.

D. Connected Components

2 seconds, 256 megabytes

You are given a simple undirected graph. It is represented by its edges list.

Your task is to calculate the number of connected components of the given graph.

Input

The first line of the input contains two integers n and m ($1 \le n \le 2 \cdot 10^5$, $0 \le m \le 2 \cdot 10^5$) — the number of vertices and the number of edges in the graph, respectively.

The next m lines describe edges. The edge i is given as a pair of vertices x_i, y_i ($1 \le x_i, y_i \le n, x_i \ne y_i$).

It is guaranteed that the given graph is simple and undirected.

Output

Print one integer — the number of connected components of the given graph.

input
5 2 1 2 4 3
output
3

input
3 2 1 2 2 3
output
1

input
3 0
output
3

input
5 3 1 2 2 3 4 5
output
2

input
6 3 1 2 2 3 5 6
output
3

E. Connected Components Sizes

2 seconds, 256 megabytes

You are given a simple undirected graph. It is represented by its edges list.

Your task is to calculate for each vertex v from 1 to n the size of its connected component.

Input

The first line of the input contains two integers n and m ($1 \le n \le 2 \cdot 10^5$, $0 \le m \le 2 \cdot 10^5$) — the number of vertices and the number of edges in the graph, respectively.

The next m lines describe edges. The edge i is given as a pair of vertices x_i, y_i ($1 \le x_i, y_i \le n, x_i \ne y_i$).

It is guaranteed that the given graph is simple and undirected.

Output

Print n integers. The i-th integer should be equal to the size of the connected component of the i-th vertex.

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

input
3 2 1 2 2 3
output
3 3 3

input
3 0
output
1 1 1

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

input
6 3 1 2 2 3 5 6
output
3 3 3 1 2 2

F. Is it a Tree?

1 second, 256 megabytes

You are given a simple undirected graph. It is represented by its edges list.

Your task is to determine if the given graph is a tree or not.

Input

The first line contains two integers n and m ($2 \le n \le 10^5; 1 \le m \le min(\frac{n(n-1)}{2}, 10^5)$) — the number of vertices and the number of edges in the graph.

The next m lines describe edges. The i-th edge is given as two integers x_i and y_i ($1 \le x_i, y_i \le n; x_i \ne y_i$). The edge is directed from the vertex x_i to the vertex y_i.

It is guaranteed that the given graph is simple and undirected.

Output

Print YES if the given graph is a tree and NO otherwise.

input
3 2 2 3 1 3
output
YES

input
4 3 1 2 2 3 3 1
output
NO

input
4 4 1 2 2 3 3 4 1 4
output
NO

G. Is it Reachable?

2 seconds, 256 megabytes

You are given a simple undirected graph. It is represented by its edges list.

Your task is to determine if the vertex b is reachable from the vertex a or not.

Input

The first line of the input contains four integers n, m, a and b ($1 \le n \le 2 \cdot 10^5$; $0 \le m \le 2 \cdot 10^5$; $1 \le a, b \le n$) — the number of vertices, the number of edges and two vertices, respectively.

The next m lines describe edges. The edge i is given as a pair of vertices x_i, y_i ($1 \le x_i, y_i \le n$, $x_i \ne y_i$).

It is guaranteed that the given graph is simple and undirected.

Output

Print YES if the vertex b is reachable from the vertex a and NO otherwise.

input
5 4 1 5 1 2 2 3 4 5 1 3
output
NO

input
5 4 1 5 1 2 2 3 4 5 5 3
output
YES