Problem A: Network of Friends

Problem Statement

In a social network of n users, each person is identified by a unique number from 0 to $n-1$. There are m friendship relations. Each friendship is bidirectional — if person u is a friend of person v, then v is also a friend of u.

Two people are considered to be in the same friend group if they are directly or indirectly connected through a chain of friendships.

Your task is to determine the total number of **distinct friend groups** in the network.

Input

- The first line contains two integers n and m ($1 \leq n \leq 10^5,\ 0 \leq m \leq 10^5$) — the number of people and the number of friendships.
- The next m lines each contain two integers u and v ($0 \leq u, v < n,\ u \neq v$) — indicating a bidirectional friendship between person u and person v.

It is guaranteed that there are no duplicate friendships and no self-loops.

Output

Print a single integer — the number of **distinct friend groups** in the network.

Example

Input

```
6 4
0 1
1 2
3 4
4 5
```

Output

```
2
```

Note

In the example above:

- The first friend group includes persons 0, 1, and 2.
- The second group includes persons 3, 4, and 5.

So, there are 2 distinct friend groups in total.

Problem B: Virus Spread Simulation

Problem Statement

A virus starts spreading from a set of infected computers in a network. The network consists of n computers and m direct connections. Each minute, the virus spreads from infected computers to all directly connected uninfected ones.

You are given a computer that is initially infected. Determine the number of minutes required to infect all computers in the network. If some computers can never be infected, output `-1`.

Input

- The first line contains two integers n and m ($1 \le n \le 10^5,\ 0 \le m \le 10^5$) — the number of computers and connections.
- The next m lines each contain two integers u and v ($0 \le u, v < n,\ u \ne v$) — a bidirectional connection between computer u and computer v.
- The following line contains a single integer k ($1 \le k \le n$) — ID of initially infected computer.

It is guaranteed that there are no duplicate connections and no self-loops.

Output

Print a single integer — the total number of minutes required to infect all computers, or `-1` if some computers can never be infected.

Example

Input

```
6 5
0 1
1 2
2 3
3 4
4 5
1
0
```

Output

```
5
```

Note

In the given network:

- Initially, only computer 0 is infected.
- After 1 minute: 1 gets infected.
- After 2 minutes: 2 gets infected.
- ...
- After 5 minutes: 5 gets infected.

So, all computers get infected in 5 minutes.

Problem C: Most Distant Nodes

Problem Statement

You are given an undirected and unweighted graph with n nodes and m edges. Your task is to find the two nodes in the graph that are the farthest apart — i.e., the pair of nodes whose shortest path is the longest among all pairs in the graph.
Print the two nodes in the graph that are the farthest apart.
If the graph is disconnected, print `-1, -1`.

Input

- The first line contains two integers n and m ($1 \leq n \leq 10^5,\ 0 \leq m \leq 10^5$) — the number of nodes and edges.
- The next m lines each contain two integers u and v ($0 \leq u, v < n,\ u \neq v$) — representing an undirected edge between nodes u and v.

It is guaranteed that there are no self-loops and no multiple edges.

Output

Print two integers — the two nodes in the graph that are the farthest apart or {-1, -1} if the graph is disconnected.

Example

Input

```
6 5
0 1
1 2
2 3
3 4
4 5
```

Output

```
5
```

Note

In the given example:

- The farthest two nodes are node 0 and node 5.
- The shortest path from 0 to 5 passes through 5 edges.

So, the output is 5.
If the graph had two or more disconnected components, the output would be `-1`.