Ashish and the Grand Flight

✓ **Points:** 100 (partial)
Time limit: 1.0s
Memory limit: 512M

Authors:
admin, 2021102016

Problem types

Allowed languages
C, C++

Once upon a time in the heart of India, a young explorer named **Ashish** from the vibrant city of **Hyderabad** won a national-level aviation-themed quiz contest. As a reward, he was offered a chance to go on a spectacular flight journey across Indian cities, from his hometown Hyderabad to the cultural capital **Kolkata**.

However, Ashish wasnt interested in the fastest route. With a wanderer soul, he wanted to visit **as many cities as possible** before reaching his final destination. The catch? The flights were all **one-way only**, and the airline network was carefully designed so that there were **no circular flight paths** — once you leave a city, you cannot return to it.

Now, Ashish needs your help to plan his most adventurous route!

You are given a list of flights (directed edges between cities). Your task is to figure out the **maximum number of cities** Ashish can visit in a single journey from **Hyderabad (city 1)** to **Kolkata (city n)**. If there is no possible path between these two cities, print `"IMPOSSIBLE"`.

The skies await your code — will you help Ashish soar across the subcontinent?

Input

The first line contains two integers `n` and `m` — the number of cities and the number of flights.

Each of the next `m` lines contains two integers `a` and `b`, meaning there is a **one-way flight** from city `a` to city `b`.

- Print a single integer — the **maximum number of cities** Ashish can visit in order.

If no such route exists, print: `"IMPOSSIBLE"`

Constraints

- `2 ≤ n ≤ 10^5`
- `1 ≤ m ≤ 2 · 10^5`
- `1 ≤ a, b ≤ n`
- The flight network forms a **Directed Acyclic Graph (DAG)**

Example

Input

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

Copy

Output

```
4
```

Copy

Note

- Ashish starts from **Hyderabad (city 1)** and wants to reach **Kolkata (city n)**.
- He wants to **maximize the number of cities** he visits, not minimize the flight time.
- The cities are numbered from 1 to `n` and are connected with **one-way** flights as given.
- Since the flight map is **acyclic**, there is no need to worry about Ashish getting stuck in a loop.

Tip for Full Marks!

If your solution is **getting around 90 points but not a full score**, chances are you might be using a general-purpose algorithm, which is great — but not the best fit here. Since the flight network is a **Directed Acyclic Graph (DAG)**, there is a more efficient way to solve this problem

So, ready your maps and algorithms?

Hello, **2024101067**.

Clarifications

Request clarification

No clarifications have been made at this time.