2 Adding Exits to a Maze

Problem Introduction

Now you decide to make sure that there are no dead zones in a maze, that is, that at least one exit is reachable from each cell. For this, you find connected components of the corresponding undirected graph and ensure that each component contains an exit cell.

Problem Description

Task. Given an undirected graph with n vertices and m edges, compute the number of connected components in it.

Input Format. A graph is given in the standard format.

Constraints. $1 \le n \le 10^3$, $0 \le m \le 10^3$.

Output Format. Output the number of connected components.

Time Limits.

language	`C`	`C++`	`Java`	`Python`	`C#`	`Haskell`	`JavaScript`	`Ruby`	`Scala`
time (sec)	1	1	1.5	5	1.5	2	5	5	3

Memory Limit. 512MB.

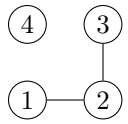
Sample 1.

Input:

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

Output:

```
2
```



There are two connected components here: $\{1, 2, 3\}$ and $\{4\}$.

Starter Files

The starter solutions for this problem read the input data from the standard input, pass it to a blank procedure, and then write the result to the standard output. You are supposed to implement your algorithm in this blank procedure if you are using `C++`, `Java`, or `Python3`. For other programming languages, you need to implement a solution from scratch. Filename: `connected_components`