Bank robbery

The city is plagued with bank robbers, and last year there was a robbery each week in some bank in the city. Police wants to fight this plague, and has to come up with a strategy to solve the problem. One obvious solution would be to place a police patrol at every bank, but this would cost too much money, and because of the crisis it is not possible to choose this solution.

An alternative solution is as follows. Police is alerted about the robbery right after it happened, and can react pretty fast. Robbers use only either motors or cars. Police is able to block any crossroad in the city, and once a crossroad is blocked it is sure that the robbers will be caught if they try to pass the blockage. Moreover, it is possible to forecast some possible destinations of escape from a given bank. Thus the strategy in this case would be to, immediately after a robbery, block some crossroads to make it impossible to reach an escape destination. It would be also great if the number of blockages would be as small as possible, since then the action is cheaper, more effective, and less troublesome for the citizens.

Obviously, if we would compute the smallest number of crossroads that make it impossible to pass from a given bank to given escape points, then this information would be actually useless for the police. Because of the geometry of the city, most likely the smallest set of blockages would consist of couple of crossroads right next to the bank, and by the time police would arrive there, the robbers would be already somewhere else. Thus during the chase, police wants to make real-time decisions that depend on the progress of the situation. To make smart decisions police needs an efficient computer program to compute blockages, and it is your task to write this program. Your program has to read the description of the city map and two locations s and t. It has to compute and return one number, the minimum number of blocked crossroads that will make it impossible to go from location s to location t.

Input

Your program reads from the standard input. In the first line of input you are given two natural numbers n and m (1 < n <= 1000, 1<= m <= 5000), where n is the number of crossroads in the city, and m is the number of streets, where each street connects two crossroads; any two crossroads can be connected by at most one street. Crossroads are numbered from 1 to n. In the line number 2 of the input you are given two numbers s and t, indicating the crossroads that have to be disconnected. In the next m lines of the input you will be given the description of the connections; each such line consists of two numbers c1, c2 and it means that there is a two-way street connecting crossroads c1 and c2.

Output

On the standard output your program has to write one number, the smallest number of crossroads that, when blocked, make it impossible to reach t from s. You may assume that there is no street directly connecting s and t.

Example 1

Input:

```
10 14
1 10
1 2
1 3
1 4
2 5
3 5
3 6
4 6
5 7
5 8
6 8
6 9
7 10
8 10
9 10
```

Output:

```
2
```

Example 2

Input:

```
10 14
5 10
1 2
1 3
1 4
2 5
3 5
3 6
4 6
5 7
5 8
6 8
6 9
7 10
8 10
9 10
```

Output:

```
3
```