Problem E
Enigmatic Enumeration

Your friend Cajsa had a graph with n vertices, and she needed to find its shortest cycle. To do this, she just took a random sequence of vertices and luckily this happened to be a shortest cycle. "What are the odds?", she asked herself and wrote another program to calculate this probability.

To do this, Cajsa needed an algorithm to count the number of shortest cycles in a graph. She uses a homemade randomized algorithm that runs in $\mathcal{O}(1)$. But you suspect that this algorithm is incorrect, because surely it would have to consider every vertex of the graph (right?). You think that Cajsa's algorithm just prints random numbers that happen to be correct on some small graphs.

In response to these doubts, Cajsa generated a bunch of graphs, and challenges you to check that her answers are correct.

You are given an undirected graph with n vertices and m edges, and your task is to count the number of shortest cycles in it.

A *cycle* in a graph is a path of **distinct** vertices where, additionally, there is an edge between the first and last vertices of the path. Two cycles are considered distinct if they don't consist of the same set of edges. Thus the cycles $3, 1, 2$ and $3, 2, 1$ are the same, and the cycles $1, 2, 3$ and $2, 3, 1$ are considered the same.

Input

The first line of input contains two integers n and m ($3 \leq n \leq 3000$, $3 \leq m \leq 6000$), the number of vertices and the number of edges.

The following m lines each contain two integers u_i and v_i ($1 \leq u_i \neq v_i \leq n$), indicating that an undirected edge goes between u_i and v_i. The graph will not contain duplicate edges or self-loops.

It is guaranteed that the graph contains at least one cycle. However, note that the graph does *not* have to be connected.

Output

Print one integer, the number of shortest cycles in the graph.

Sample Input 1	Sample Output 1
4 4 1 2 2 3 3 4 4 1	1

Sample Input 2

```
5 10
1 2
1 3
1 4
1 5
2 3
2 4
2 5
3 4
3 5
4 5
```

Sample Output 2

```
10
```

Sample Input 3

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

Sample Output 3

```
2
```