E. TOF

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Today Pari gave Arya a cool graph problem. Arya wrote a non-optimal solution for it, because he believes in his ability to optimize non-optimal solutions. In addition to being non-optimal, his code was buggy and he tried a lot to optimize it, so the code also became dirty! He keeps getting Time Limit Exceeds and he is disappointed. Suddenly a bright idea came to his mind!

Here is how his dirty code looks like:

```
dfs(v)
{
     set count[v] = count[v] + 1
     if(count[v] < 1000)
     {
          foreach u in neighbors[v]
          {
               if(visited[u] is equal to false)
               {
                    dfs(u)
               }
               break
          }
     }
     set visited[v] = true
}

main()
{
     input the digraph()
     TOF()
     foreach 1<=i<=n
     {
          set count[i] = 0 , visited[i] = false
     }
     foreach 1 <= v <= n
     {
          if(visited[v] is equal to false)
          {
               dfs(v)
          }
     }
     ... // And do something cool and magical but we can't tell you what!
}
```

He asks you to write the TOF function in order to optimize the running time of the code with minimizing the number of calls of the dfs function. The input is a directed graph and in the TOF function you have to rearrange the edges of the graph in the list neighbors for each vertex. The number of calls of dfs function depends on the arrangement of neighbors of each vertex.

Input

The first line of the input contains two integers n and m ($1 \le n, m \le 5000$) — the number of vertices and then number of directed edges in the input graph.

Each of the next m lines contains a pair of integers u_i and v_i ($1 \le u_i, v_i \le n$), meaning there is a directed edge in the input graph.

You may assume that the graph won't contain any self-loops and there is at most one edge between any unordered pair of vertices.

Output

Print a single integer — the minimum possible number of dfs calls that can be achieved with permuting the edges.

Examples

input
3 3 1 2 2 3 3 1
output
2998

input
6 7 1 2 2 3 3 1 3 4 4 5 5 6 6 4
output
3001