# How far on Königsberg



Suppose you are given an undirected graph  $G=(V_1\cup V_2,E)$  in which there are two components connected by a single edge. Let's refer the two components as  $G_1=(V_1,E_1)$  and  $G_2=(V_2,E_2)$ . For each pair of vertices (u,v) in any of the components of G, they lie on at least one simple cycle.

Given such a graph G and a pair of vertices  $s\in G_1$  and  $t\in G_2$ , write a program which find the shortest distance between them. Since graph G is unweighted, the distance is simply the number of edges between s and t.

# Input Format

The first line of each test will be three space separated positive integers |V|, |E| and q. |V| and |E| denotes number of vertices and number of edges in input graph G respectively, while q denotes number of queries. Let the vertices of graph G be labelled with  $\{0,\ldots,|V|-1\}$ 

Each of the following |E| lines will contain two space separated positive integers u and v denoting an edge between vertix u and v.

Each of the next q lines will contain a query as two space separated positive integers  $s \in G_1$  and  $t \in G_2$ .

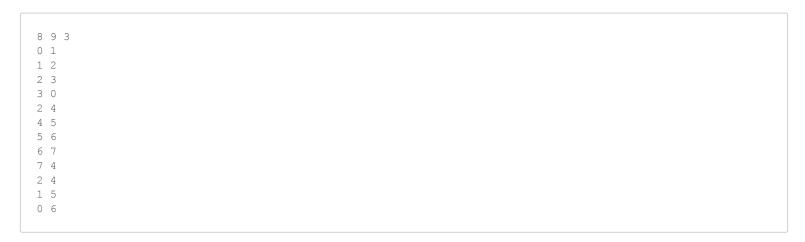
#### Constraints

- $1 \le |V| \le 10^4$
- $1 \le |E| \le 10^8$
- $1 \le q \le 10^4$

### **Output Format**

For each of the q queries print the shortest distance (minimum number of edges) between s and t.

# Sample Input 0



# Sample Output 0

```
1
3
5
```