#Problem 1: Unweighted Shortest Path - Undirected

Description: In this problem you will be given an undirected unweighted (or each edge equally weighted) graph G, a source vertex S and destination vertex D. You need to calculate the shortest distance between S to D. In this problem, the shortest distance between two vertices means using the minimum number of edges to reach from one vertex to the other.

In the first line you will be given two integers V and E denoting the number of vertices and edges of this graph respectively. All the vertices will have ids between 1 to V. In the following E lines, you will get the information about all the edges. In the i^{th} line you will have two integers x (1<=x<=V) and y (1<=y<=V) which denotes there is an undirected edge between x and y. After E lines of input, you will get two integers S and D (1<=S<=V, 1<=D<=V) denoting the source and destination vertices of the problem.

As output, you will print an integer number d which will denote the minimum distance between S and D.

Limits
1<=V<=100000, 1<=|E|<=100000
Test Cases:

Input	Output

4 5
1 2
1 3
2 4
3 4
1 4
1 4
	1

6 8
1 2
1 3
1 4
1 5
2 5
3 5
4 5
5 6
1 6
	2

10 13
1 7
1 5
2 6
	2

2 10
2 7
3 8
3 4
3 6
4 6
5 10
6 8
7 9
8 9
2 3	

12 16
1 11
1 8
2 4
2 3
3 7
4 12
4 5
5 10
5 9
6 7
6 10
7 9
7 8
8 12
9 11
11 12
8 10
	3

#Problem 2: Bicoloring

Description: In this problem you will be given an undirected unweighted graph G. You have to identify if the given graph is bicolorable or not. The rule of coloring this graph is, for each edge connecting two vertices u and v, must have different colors.

In the first line you will be given two integers V and E denoting the number of vertices and edges of this graph respectively. All the vertices will have ids between 1 to V. In the following E lines, you will get the information about all the edges. In the i^{th} line you will have two integers x (1<=x<=V) and y (1<=y<=V) which denotes there is an undirected edge between x and y.

As output, you need to print a string “YES” without quotes if the graph is bi-colorable else print “NO”.

Limits
1<=V<=100000, 1<=|E|<=100000

Test Cases:

Input	Output

4 5
1 2
1 3
2 4
3 4
1 4
	NO

4 4
1 2
1 3
2 4
3 4
	YES

10 10
1 2
1 9
2 5
3 9
3 4
4 7
5 10
6 7
6 8
8 10
	YES

9 9
1 9
1 7
2 6
2 4
3 5
3 7
4 9
5 8
6 8
	NO

#Problem 3: Cycle Finding and Printing

Description: In this problem you will be given an undirected unweighted graph G. You have to determine if the graph contains any cycle or not.

In the first line you will be given two integers V and E denoting the number of vertices and edges of this graph respectively. All the vertices will have ids between 1 to V. In the following E lines, you will get the information about all the edges. In the ith line you will have two integers x (1<=x<=V) and y (1<=y<=V) which denotes there is an undirected edge between x and y.

You need to write a program that will detect if the graph contains any cycle or not. If not found print “NO” (without quote). But if the graph contains any cycle, you need to print “YES” (without quote). In the following line you need to print the vertices which make the cycle maintaining the ascending order of the vertices’ ids. No vertex will be repeated in the printing. In this problem, to have a cycle, you will need at least three nodes. In this problem you can safely assume that the given graph will contain a single cycle or not. Print the vertices separated by a single space.

Limits
1<=V<=100000, 1<=|E|<=100000

Test Cases:

Input	Output

4 3
1 2
1 3
1 4
	NO

4 4
1 2
1 3
1 4
2 4
	YES
	1 2 4

10 10
1 2
1 5
2 4
3 4
3 5
4 8
6 10
6 7
8 10
9 10
	YES
	1 2 3 4 5

10 9
1 2
2 3
2 4
3 5
4 9
6 8
6 7
	No

7 10 8 9	

#Problem 4: Maximal Connected Component in an Undirected Graph

Description: In this problem you will be given an undirected unweighted graph G. You have to discover the maximal connected component from this graph. A connected component C_1 is larger compared to the other connected component C_2 iff C1 contains more vertices than C2.

In the first line you will be given two integers V and E denoting the number of vertices and edges of this graph respectively. All the vertices will have ids between 1 to V. In the following E lines, you will get the information about all the edges. In the i^{th} line you will have two integers x (1<=x<=V) and y (1<=y<=V) which denotes there is an undirected edge between x and y.

You need to write a program that will detect the maximal connected component from the given graph as input. There will be a single line of output for each test case containing the number of vertices that make the maximal connected component.

Limits
1<=V<=1000, 1<=|E|<=1000000

Test Cases:

Input	Output

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

15 15
1 8
1 7
2 10
2 3
3 9
4 7
4 9
5 6
5 8
6 10
11 14
	10

11 15
12 13
12 14
13 15	

#Problem 5: Tree Diameter for an undirected tree graph

Description: In this problem you will be given an undirected unweighted Tree graph G. You have to discover the diameter of the given tree. Tree diameter mainly indicates the largest distance found between any two nodes in the given tree. Distance means the number of edges between the nodes.

In the first line you will be given two integers V and E denoting the number of vertices and edges of this graph respectively. All the vertices will have ids between 1 to V. In the following E lines, you will get the information about all the edges. In the ith line you will have two integers x (1<=x<=V) and y (1<=y<=V) which denotes there is an undirected edge between x and y.

You will write a program that will calculate the diameter of the given undirected tree. For each test case you will print a single line denoting the diameter.

Limits
1<=V<=1000, 1<=|E|<=1000000

Test Cases:

Input	Output

5 4
1 2
1 5
2 3
3 4
	4

7 6
1 2
1 4
2 3
4 5
5 6
6 7
	6

20 19
1 4
2 4
2 5
	7

3 4
4 8
4 6
4 9
4 10
5 17
7 8
11 20
12 17
13 19
14 17
15 17
16 17
17 20
17 19
17 18	