BFS Traversal 4

Given an undirected and disconnected graph G(V, E), print its BFS traversal.
Note:
Here you need to consider that you need to print BFS path starting from vertex 0 only.
V is the number of vertices present in graph G and vertices are numbered from 0 to V-1.
E is the number of edges present in graph G.
Take graph input in the adjacency matrix.
Handle for Disconnected Graphs as well

Input Format

The first line of input contains two integers, that denote the value of V and E.
Each of the following E lines contains space separated two integers, that denote that there exists an edge between vertex a and b.

Constraints

0 <= V <= 1000
0 <= E <= (V * (V - 1)) / 2
0 <= a <= V - 1

0 <= b <= V - 1

Time Limit: 1 second

Output Format

Print the BFS Traversal, as described in the task.

Sample Input 0

```
4 4
0 1
0 3
1 2
2 3
```

Sample Output 0

```
0 1 3 2
```

```java
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be named Solution. */
        Scanner sc=new Scanner(System.in);
        int v=sc.nextInt();
        int e=sc.nextInt();
        int arr[][]=new int[v][v];
        for(int i=0;i<e;i++){
            int st=sc.nextInt();
            int ed=sc.nextInt();
            arr[st][ed]=1;
            arr[ed][st]=1;
        }
        Queue<Integer> q=new LinkedList<>();
        boolean vis[]=new boolean[v];
        q.add(0);
        vis[0]=true;
        while(!q.isEmpty()){
            int ele=q.poll();
            System.out.print(ele+" ");
            for(int i=0;i<v;i++){
                if(arr[ele][i]==1 && !vis[i]){
                    q.add(i);
                    vis[i]=true;
                }
            }
        }
    }
}
```

Line: 34 Col: 10

Upload Code as File Test against custom input Run Code Submit Code

Testcase 0 ✔

Congratulations, you passed the sample test case.

Click the **Submit Code** button to run your code against all the test cases.

Input (stdin)

```
4 4
0 1
0 3
1 2
2 3
```

Your Output (stdout)

```
0 1 3 2
```

Expected Output

```
0 1 3 2
```