```
/*
    Time complexity: O(V + E)
    Space complexity: O(V^2)

    where V is the number of vertices in the input graph and
    E is the number of edges in the input graph
*/
import java.util.ArrayList;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;
public class Solution {
    static BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

        public static ArrayList<Integer> getPathDFSHelper(int[][] edges, int sv, int ev, boolean[] visited) {

                if(sv == ev) {
                        ArrayList<Integer> ans = new ArrayList<>();
                        ans.add(sv);
                        return ans;
                }

                visited[sv] = true;
                int n = edges.length;

                for(int i = 0; i < n; i++) {
                        if(edges[sv][i] == 1 && !visited[i]) {
                                ArrayList<Integer> smallAns = getPathDFSHelper(edges, i, ev, visited);

                                if(smallAns != null) {
                                        smallAns.add(sv);
                                        return smallAns;
                                }
                        }
                }

                return null;
        }

        public static ArrayList<Integer> getPathDFS(int[][] edges, int sv, int ev) {
                boolean visited[] = new boolean[edges.length];

                return getPathDFSHelper(edges, sv, ev, visited);

        }

        public static void main(String[] args) throws NumberFormatException, IOException {
                String[] strNums;
```

```
        strNums = br.readLine().split("\\s");
        int n = Integer.parseInt(strNums[0]);
        int e = Integer.parseInt(strNums[1]);


                int edges[][] = new int[n][n];

        for (int i = 0; i < e; i++) {
            String[] strNums1;
            strNums1 = br.readLine().split("\\s");
            int fv = Integer.parseInt(strNums1[0]);
            int sv = Integer.parseInt(strNums1[1]);
            edges[fv][sv] = 1;
            edges[sv][fv] = 1;
        }

                String[] strNums1;
        strNums1 = br.readLine().split("\\s");
        int sv = Integer.parseInt(strNums1[0]);
        int ev = Integer.parseInt(strNums1[1]);

                ArrayList<Integer> ans = getPathDFS(edges, sv, ev);
                if(ans != null) {
                        for(int elem: ans) {
                                System.out.print(elem + " ");
                        }
                }
        }
}
```