Iterative DFS

```
import java.io.*;
import java.util.*;

public class Main {
   static class Edge {
      int src;
      int nbr;

      Edge(int src, int nbr) {
         this.src = src;
         this.nbr = nbr;
      }
   }

   public static void main(String[] args) throws Exception {
      BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));

      int vtces = Integer.parseInt(br.readLine());
      ArrayList<Edge>[] graph = new ArrayList[vtces];
      for (int i = 0; i < vtces; i++) {
         graph[i] = new ArrayList<>();
      }

      int edges = Integer.parseInt(br.readLine());
      for (int i = 0; i < edges; i++) {
         String[] parts = br.readLine().split(" ");
         int v1 = Integer.parseInt(parts[0]);
         int v2 = Integer.parseInt(parts[1]);
         graph[v1].add(new Edge(v1, v2));
         graph[v2].add(new Edge(v2, v1));
      }

      int src = Integer.parseInt(br.readLine());

      Stack<Pair> stack = new Stack<>();
      stack.push(new Pair(src, src + ""));
      boolean[] visited = new boolean[vtces];
      while(stack.size() > 0){
```

```
            Pair rem = stack.pop();

            if(visited[rem.v] == true){
                continue;
            }
            visited[rem.v] = true;
            System.out.println(rem.v + "@" + rem.psf);

            for (Edge e : graph[rem.v]) {
                if (visited[e.nbr] == false) {
                    stack.push(new Pair(e.nbr, rem.psf + e.nbr));
                }
            }
        }
    }

    static class Pair {
        int v;
        String psf;

        Pair(int v, String psf){
            this.v = v;
            this.psf = psf;
        }
    }
}
```

Nothing much. Just pop from end in the queue instead of beginning.