



통계 수정 삭제

백준

9/12

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초	512 MB	5112	2483	1638	47.205%

문제

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
2 초	512 MB	5112	2483	1638	47.205%

문제

[illegible]

입력

출력

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

public class Main {
    static List<Integer> adjList[];
    static boolean[] visited, isCycle;
    static int[] dist;
    static int N;
```

```

static Queue<Integer> q = new LinkedList<>();
public static void main(String[] args) throws Exception {
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    N = Integer.parseInt(br.readLine());

    adjList = new ArrayList[N + 1];

    for (int i = 1; i <= N; i++) {
        adjList[i] = new ArrayList<>();
    }

    for (int i = 0; i < N; i++) {
        StringTokenizer st = new StringTokenizer(br.readLine());
        int curr = Integer.parseInt(st.nextToken());
        int next = Integer.parseInt(st.nextToken());

        adjList[curr].add(next);
        adjList[next].add(curr);
    }

    dist = new int[N + 1];
    isCycle = new boolean[N+1];
    // 순환선 찾기
    for(int i=1; i<=N; i++) {
        visited = new boolean[N + 1];
        findCycle(i, i,1);
    }

    for(int i=1; i<=N; i++) {
        // 사이클 아닌 애들 대표값에서 미리 분류
        if(!isCycle[i]) dist[i]= -1;
        else q.add(i);
    }

    // 찾은 순환선에서 떨어진 최소 루트 구하기
    // 다익스트라
    dijkstra();
    StringBuilder sb = new StringBuilder();
    for(int i=1; i<=N; i++) {
        sb.append(dist[i]).append(" ");
    }
    System.out.println(sb.toString());
}

private static void findCycle(int idx, int start ,int cnt) {
    visited[idx] = true;
    for (int next : adjList[idx]) {
        // 사이클인 경우
        // 1. 시작한 곳이 다시 나올것
        // 2. 최소 3개 이상의 노드로 구성될 것
        // 2개의 경우에는 그냥 양방향으로 연결되어서 가는 경우도 있으므로 3개 이상이어야 함
        if(next == start && cnt >= 3) isCycle[next] = true;
        if(!visited[next]) findCycle(next, start,cnt+1);
    }
}

private static void dijkstra() {
    while(!q.isEmpty()) {
        int curr = q.poll();
        for(int next: adjList[curr]) {
            // 사이클이 아닌 노드라면
            if(dist[next] == -1) {
                // BFS는 와이파이라고 했으니, 가장 먼저 도달하는 곳에서 +1 한 값이 최소 루트이겠지
                dist[next]= dist[curr]+1;
                q.add(next);
            }
        }
    }
}
}

```