백준 19538번 - 루머

백준 19538번 - 루머

시간제한	메모리 제한	제출	정답	맞은 사람	정답 비율
10초	1024MB	2327	928	683	40.271%

문제

당신은 루머를 믿는가?

한 유명 심리학 실험에서는 사람들에게 두 개의 줄을 보여주고, 어떤 줄이 더 긴지 말하라 했다. 사실 한 사람을 제외하고 나머지는 실험자에 의해 사전에 조작된 사람들이었다. 조작된 사람들은 사실상 더 짧은 줄을 더 길다고 말했다. 주변 모두가 같은 답변을 하자, 진짜 피실험자 또한 짧은 줄이 더 길다고 말했다. 이 실험은 사람들이 주변인의 영향을 강하게 받는다는 것을 보여주었는데, 루머도 이와 같다.

루머는 최초 유포자로부터 시작한다. 최초 유포자는 여러 명일 수 있고, 최초 유포자를 제외하고 스스로 루머를 만들어 믿는 사람은 없다.

매분 루머를 믿는 사람은 모든 주변인에게 루머를 동시에 퍼트리며, 군중 속 사람은 주변인의 절반 이상이 루머를 믿을 때 본인도 루머를 믿는다.

루머를 믿는 순간부터 다른 말은 듣지 않기 때문에, 한번 믿은 루머는 계속 믿는다.

이때, 사람들이 루머를 처음 믿기 시작하는 시간을 알아내 보자.

입력

첫째 줄에 사람의 수 N (1≤N≤200 000)이 주어진다. 이는 1번 사람부터 N번 사람까지 있음을 의미한다.

둘째 줄부터 N개의 줄이 주어진다. 이 중 i(1≤i≤N)번째 줄에는 i번 사람의 주변인들의 번호와 입력의 마지막을 나타내는 `0`이 공백으로 구분되어 주어진다. 번호는 1이상 N 이하의 자연수이고, 같은 줄에 중복된 번호는 없다. 자기 자신이 주변인이거나 일방적으로 주변인인 경우는 없으며, 전체 양방향 주변인 관계는 1 000 000개를 넘지 않는다.

다음 줄에는 루머를 퍼뜨리는 최초 유포자의 수 M이 주어진다. (1≤M≤N)

마지막 줄에는 최초 유포자의 번호가 공백으로 구분되어 주어진다. 최초 유포자의 번호는 중복되지 않는다.

출력

N개의 정수 t1,t2,⋯,tN을 공백 단위로 출력한다. ti는 i번 사람이 루머를 처음 믿기 시작한 시간(분)이며, 충분히 많은 시간이 지나도 루머를 믿지 않을 경우 −1이다. 최초 유포자는 루머를 0분부터 믿기 시작했다고 생각한다.

예제 입력1

```
7
2 3 0
1 3 0
1 2 4 0
3 5 0
4 0
0
0
2
1 6
```

예제 출력1

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

예제 입력2

```
7
2 4 0
1 3 0
2 5 0
1 5 6 0
3 4 6 7 0
4 5 7 0
5 6 0
1
6
```

예제 출력2

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

출처

출처

알고리즘 분류

- 그래프 이론
- 그래프 탐색
- 너비 우선 탐색

접근 방법

최초 유포자가 루머를 알게 된 시점이 `0` 이므로, `1`이라는 시간에 루머를 믿게 되는 사람들은 최초 유포자와 인접한 노드들이다. 만약, 인접한 노드들이 루머를 받아들이게 될 경우, 또 인접한 노드에 영향을 줄 수 있으므로, 이 경우에 해당 정점을 기준으로 루머 유포한다. 이 과정은 BFS와 1대1로 매칭된다.

인접한 노드가 루머를 받아들이는 경우는 `인접한 노드 중 절반 이상이 루머를 믿을 때` 이므로, 해당 조건을 고려하기 위해서는 인접 노드들의 개수를 counting하는 로직 또한 필요하다. 이 때 인접 노드들의 개수(k)가 홀수일 경우 절반은 k/2 + 1이 된다는 것을 잊지말자. 가장 편한 방법은 올림 처리를 하는 것이다.

필자는 {현재 시간, 현재 노드}의 pair를 큐에 넣어, 이미 루머를 받아 들인 노드에 대해서는 추가적인 탐색 없이 skip하고 소문을 받아 들인 시간이 현재 시간 이전일 경우에만 counting하는 것으로 timing 문제 또한 해결하였다.

소스코드

```cpp
#define FASTIO cin.tie(0)->sync_with_stdio(false), cout.tie(0)
////////////////////////////////////////////////////////////////////
#include <bits/stdc++.h>
using namespace std;
int N, M, t[200001];
vector<int> adj[200001];
int main(void){
    FASTIO;
////////////////////////////////////////////////////////////////////

    cin >> N;
    for (int i = 1; i <= N; i++) {
        int in;
        while (cin >> in, in != 0) {
            adj[i].push_back(in);
            adj[in].push_back(i);
        }
    }

    memset(t, -1, sizeof(t));
    cin >> M;

    priority_queue<pair<int,int>, vector<pair<int,int>>, greater<pair<int,int>>> PQ;
    for (int i = 0; i < M; i++) {
        int in;
```

```
        cin >> in;
        t[in] = 0;
        PQ.push({0, in});
    }

    while (!PQ.empty()) {
        // 특정 정점이 루머를 알게 될 경우에만, update의 가능성이 생긴다.
        auto[curTime, cur] = PQ.top();
        PQ.pop();

        for (auto next: adj[cur]) {
            if (t[next] != -1)
                continue;
            // next의 모든 인접 노드들 중 1/2이상이 소문을 알고 있다면? Queue에 추가
            int count = 0;
            for (auto nearBy: adj[next]) {
                if (t[nearBy] != -1 && t[nearBy] <= curTime)
                    count++;
            }
            if (count >= ceil(adj[next].size()/2.0)) {
                t[next] = curTime + 1;
                PQ.push({curTime + 1, next});
            }
        }
    }

    for (int i = 1; i <= N; i++) {
        cout << t[i] << ' ';
    }
    return 0;
}
```