BOJ 15681 트리와 쿼리

SCV 2022.01.27

Step1 접근

문제

간선에 가중치와 방향성이 없는 임의의 루트 있는 트리가 주어졌을 때, 아래의 쿼리에 답해보도록 하자.

- 정점 U를 루트로 하는 서브트리에 속한 정점의 수를 출력한다.

만약 이 문제를 해결하는 데에 어려움이 있다면, 하단의 힌트에 첨부한 문서를 참고하자.

입력

트리의 정점의 수 N과 루트의 번호 R, 쿼리의 수 Q가 주어진다. ($2 \leq N \leq 10^5$, $1 \leq R \leq N$, $1 \leq Q \leq 10^5$)

이어 N-1줄에 걸쳐, U V의 형태로 트리에 속한 간선의 정보가 주어진다. ($1 \leq U, V \leq N$, $U \neq V$)

이는 U와 V를 양 끝점으로 하는 간선이 트리에 속함을 의미한다.

이어 Q줄에 걸쳐, 문제에 설명한 U가 하나씩 주어진다. ($1 \leq U \leq N$)

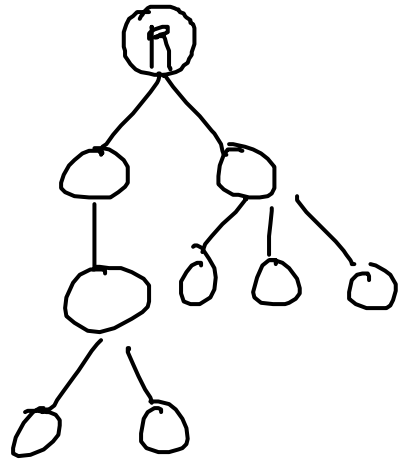
입력으로 주어지는 트리는 항상 올바른 트리임이 보장된다.

출력

Q줄에 걸쳐 각 쿼리의 답을 정수 하나로 출력한다.

트리가 주어지고 루트로 삼을 정점 R이 주어짐!

이때 정점 U를 루트로 하는 서브트리에 속한 노드 개수 출력



트리 DP를 이해하기 위한 기본적인 문제

주어진 트리가 정점 U를 루트로 할 때

정점 U의 서브트리를 구성하고 있는 노드의 개수를 구하는 문제

정점 U의 서브트리의 노드의 개수 + 1(정점 U)

매번 리프노드까지 접근해서 노드의 개수를 알려고 하면 시간이 오래 걸림

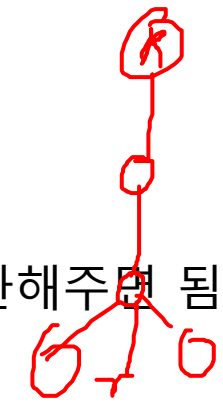
루트에서 시작하여 자신을 포함한 서브트리에서의 개수를 **저장하기**

초기값
size = [0] * (N+1) // dp배열

top-down 방식 사용

루트 R에서 시작하여 size값을 모두 구해두면

이후 쿼리가 들어왔을 때 해당 쿼리의 size[query]만 반환해주면 됨!



Step2 구현

```
int solve(int curr, vector<vector<int>> &tree, vector<int> &size)
{
    if (size[curr] != 0)
        return size[curr];

    size[curr] = 1;
    for (int &child : tree[curr]) {
        if (size[child] != 0)
            continue;

        size[curr] += solve(child, tree, size);
    }

    return size[curr];
}
```

```cpp
int main(void)
{
    cin.tie(NULL);
    cout.tie(NULL);

    int n, r, q;
    cin >> n >> r >> q;

    vector<vector<int>> tree((n+1), vector<int>());
    vector<int> size(n + 1, 0);

    for (int i = 0; i < n-1; i++) {
        int v1, v2;
        cin >> v1 >> v2;

        tree[v1].push_back(v2);
        tree[v2].push_back(v1);
    }
```

```cpp
    solve(r, tree, size);

    for (int i = 0; i < q; i++) {
        int subRoot;
        cin >> subRoot;

        cout << size[subRoot] << "\n";
    }

    return 0;
}
```

THX~