1. MooTube (Silver)

먼저, N개의 동영상 중, 임의의 두 동영상 쌍에 대한 USADO 값이 N-1개 가 주어진다. 이는 N개의 동영상이 USADO의 가중치를 가진 간선으로 이어진 트리 구조를 이룸을 알 수 있다. 따라서 임의의 두 정점에 대하여 둘 사이의 경로는 유일함이 보장된다. 그리고 임의의 두 동영상에 대하여 두 동영상의 USADO는 경로상에 존재한 USADO 값 중, 최솟값이다.

그렇다면 v와 k이상의 USADO를 가진 동영상을 어떻게 찾을 수 있을까? 문제에서 제시한 것처럼 동영상 v를 정점이라 하겠다. v에 직접 연결된, 즉 v를 root node로 뒀을 때, 자식 node와의 USADO 가 k 이상인 USADO를 가진 자식 노드를 먼저 찾아낸다. 왜냐하면 k미만의 USADO를 가진 node를 root로 하는 서브 트리에 대해서 해당 서브 트리의 임의의 정점 u에 대해서 v, u의 USADO는 k미만임이 보장된다. 따라서 찾아낸 node를 제외한 node론 더 이상 탐색을 진행하지 않아도 된다. 결과적으로 찾아낸 노드에 대해서 해당 노드들을 다시 root로 두고 해당 서브 트리에 대해서 위의 과정을 반복하여 찾아낼 노드가 없을 때까지 찾아낸 노드의 개수는 v와 k이상의 USADO를 가진 노드의 개수와 같다.

1. 전단지 돌리기

이 문제 또한 트리로 구성 되어있다. 따라서 임의의 두 정점에 대해서 정점을 있는 경로는 유일하다. 따라서 S를 root로 한 트리에 대해서 각 정점 v에서 가장 깊은 곳에 있는 정점까지의 거리를 max\_depth(v) 라 하자.

그렇다면 S에서 탐색을 시작해서 연결된 간선의 수를 센다. 이 때, 탐색할 정점 v에 대해서 max\_depth(v) 가 D 미만이라면 정점 v 를 탐색하지 않아도 된다. 따라서 이는 S를 root로 한 트리에 대해서 leaf node 로부터 d만큼 떨어진 정점들까지 연결된 간선의 수를 세는 것과 같다. 그러나, 문제의 경우 왕복한 거리이기에 구한 간선의 수에 2를 곱하면 된다.

1. 얼음깨기 펭귄

N개의 정점에 대해서 N-1개의 연결된 얼음의 정보가 주어지므로 해당 얼음판은 트리구조이다. 펭귄이 위치한 정점 P에 대해서 P와 연결된 지지대 얼음이 2개 이상 있으면서 최대한 많은 얼음을 깨야 하므로 2개의 지지대 얼음과만 연결하면 된다. 그렇다면 2개의 지지대 얼음은 어떻게 골라야 할까? 지지대 얼음들 {1,2,3,4,5, … , S} 에 대하여 P에서 각 지지대 얼음까지의 거리를 {p1,p2,p3,p4,…,pS} 라 하자. 연결할 두 지지대 얼음을 i,j 라 하자. 그렇다면 깰 얼음의 개수는 N-(pi+pj) 이다. 따라서 pi+pj가 최소여야 한다. 따라서 이는 P에서 가장 가까운 지지대 얼음 2개를 고르는 문제로 바뀐다. 이는 P 지점에서 BFS를 수행하면 해결할 수 있다.

1. 연결

기본적으로 빨간색과 파란색점을 각각 최단거리로 이어주면 된다. 이는 2차원 배열에 대하여 BFS로 해결할 수 있으나 빨간색을 먼저 연결할 때와 파란색을 먼저 연결할 때, 결과가 달라질 수 있다.

다음과 같은 상황을 보자.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

이 상태에서 파란색을 먼저 연결하면 다음과 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

이 필요한 전선의 길이는 파란 점을 연결하는데 필요한 길이 4, 빨간 점은 8이 필요하며 총 12의 길이가 필요하다.

빨간색을 먼저 하게 되면 다음과 같다

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

이때 빨간 점을 연결하는데 필요한 길이 3, 파란 점을 연결하는데 필요한 길이 7이므로 10의 길이가 필요하다. 각 색깔의 정점을 이을 때 빨간 점을 연결한 뒤 파란 점을 잇는 경우와 반대의 경우를 모두 수행하여 두 케이스 중 더 짧은 경로를 가진 케이스를 고르면 된다. 이 때 주의할 점은 2가지이다.

1. 두 점을 연결하는데 다른 점을 지나선 안된다.
2. 연결된 선을 질러 가선 안된다.

2차원 배열 connect에 대하여 connect[r][c]가 true 라면 r행 c열에는 전선이 깔려 있다는 의미라 하자.

1번 케이스의 경우엔 각 색깔별 두 점에 대하여 BFS를 진행할 때, 이미 전선이 깔려 있다고 표시를 하면 된다.

2번 케이스의 경우엔 BFS를 진행하며 다음 정점의 좌표를 큐에 넣을 때, 다음 정점에 이전 정점의 좌표 값을 넣어 탐색이 끝난 뒤, 종점에서부터 역으로 거슬러 올라가 connect배열에 true로 표시하고 다음 탐색을 진행하면 된다.

1. 색칠하기

1,2,3번과 다르게 주어진 입력이 트리라는 보장이 없다. 또한 하나의 그래프란 주어진단 보장이 없다. 그러나, 설명을 위해 하나의 그래프만을 가정하겠다. 해당 그래프의 임의의 정점 v에 대하여 색깔 1로 칠했다 가정하자. 그렇다면 v와 연결된 정점들에 대하여 해당 정점들은 색깔 2 여야 한다. 그리고 색깔 2 로 칠한 정점들에 대하여 자신과 연결된 정점을 1로 칠해야 한다. 이 때, 1로 칠해야 할 자리에 2가 칠해져 있거나, 2로 칠해야 할 자리에 1이 칠해져 있다면 2가지 색깔로 칠할 수 없는 경우를 말한다. 따라서 이 문제는 1~N까지 방문하지 않은 정점에 대하여 해당 정점을 아무 색으로 칠한 뒤, 해당 점을 시작으로 그래프를 탐색하며 각 정점 별로 자신과 연결된 정점 중에 같은 색이 있는 지 확인하면 해결할 수 있다.