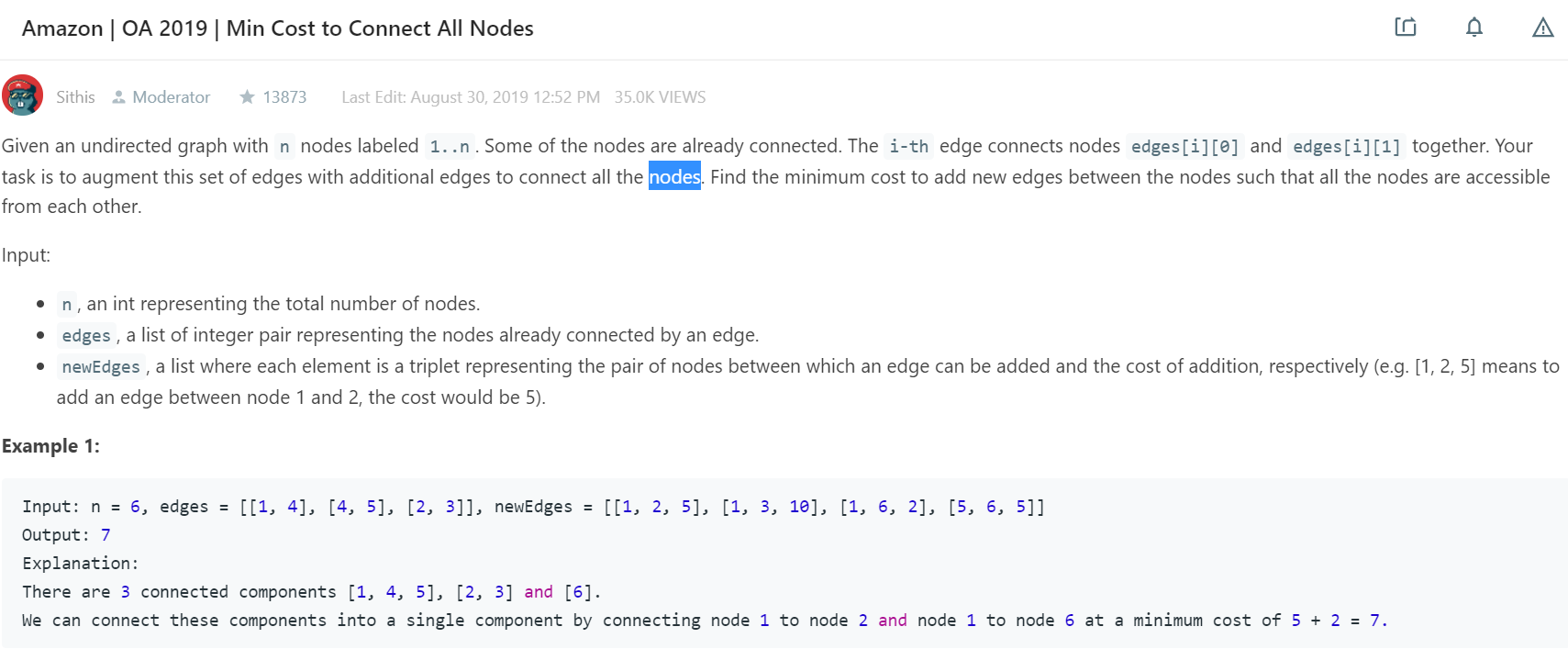
<https://leetcode.com/discuss/interview-question/356981>



// Amazon | OA 2019 | Min Cost to Connect All Nodes

// https://leetcode.com/discuss/interview-question/356960/Amazon-or-OA-2019-or-Find-Pair-With-Given-Sum

public class Main {

int[] parent;

int component;

private int find(int v){

if(parent[v] == v) return v;

return parent[v] = find(parent[v]);

}

private void connect(int v1, int v2){

if(find(v1) == find(v2)) return;

int root = find(v1);

while(v2 != parent[v2]){

int temp = parent[v2];

parent[v2] = root;

v2 = temp;

}

--component;

parent[v2] = root;

}

private boolean isConnected(int v1, int v2){

return find(v1) == find(v2);

}

public int minCosttoConnectAllNodes(int n, int edges[][], int newEdges[][]){

parent = new int[n + 1]; component = n;

for(int i = 0; i <= n; ++i) parent[i] = i;

for(int[] edge: edges) connect(edge[0], edge[1]);

Arrays.sort(newEdges, (a, b) -> (a[2] - b[2]));

int cost = 0;

for(int i = 0; i < newEdges.length; ++i){

if(!isConnected(newEdges[i][0], newEdges[i][1])){

connect(newEdges[i][0], newEdges[i][1]);

cost += newEdges[i][2];

if(component == 1) return cost;

}

}

return -1;

}

public static void main(String[] args) {

Main main = new Main();

int[] n\_tests = {6};

int[][][] edges\_tests = { {{1, 4}, {4, 5}, {2, 3}}};

int[][][] newEdges\_tests = { {{1, 2, 5}, {1, 3, 10}, {1, 6, 2}, {5, 6, 5}}};

for(int i = 0; i < n\_tests.length; ++i){

System.out.println(main.minCosttoConnectAllNodes(n\_tests[i], edges\_tests[i],

newEdges\_tests[i]));

}

}

}