**SOLUTION**

class Solution {

public:

bool possibleBipartition(int N, vector<vector<int>>& dislikes) {

vector<int> groups(N, -1);

vector<vector<int>> adj(N,vector<int>());

for(vector<int>& p: dislikes){

adj[p[0]-1].push\_back(p[1]-1);

adj[p[1]-1].push\_back(p[0]-1);

}

for(int i=0;i<N;i++){

if(groups[i]==-1 && !dfs(adj,groups,i,0))

return false;

}

return true;

}

bool dfs(vector<vector<int>>& adj, vector<int>& groups, int v, int grp){

if(groups[v] == -1)

groups[v] = grp;

else

return groups[v] == grp;

for(int n: adj[v]){

if(!dfs(adj,groups,n,1-grp))

return false;

}

return true;

}

};

**TIME COMPLEXITY= O(N)**

**SPACE COMPLEXITY= O(N)**