#include<bits/stdc++.h>

#define llINF 9223372036854775807

#define pi 3.141592653589793//23846264338327950254

#define endl "\n"

#define mp make\_pair

#define pb push\_back

#define ll long long

#define int long long

#define IOS ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);

**using** **namespace** std;

**const** ll maxn=1e5+7;

**const** **double** eps=1e-10;

**const** ll mod=1e9+7;

//首先需要推一下无法染色的情况，其实就是某一个点存在三条或更多的边的时候，必然会出现染色冲突的情况（自己画图证明）

//再由任意三个相邻的点颜色必不同，可以推出整棵树其实把所有点分成三个部分，每个部分的所有点颜色必然是相同的

//之后我们可以从某一个叶子结点出发dfs，把整个树的点分割成三个部分，再枚举这三个部分对应颜色计算即可

vector<ll>field[maxn];

ll val[3][maxn];

ll cas[maxn];

ll color[6][3]=

{

    1,2,3,

    1,3,2,

    2,1,3,

    2,3,1,

    3,1,2,

    3,2,1

};

**void** dfs(ll now,ll ope)

{

**if**(cas[now]==-1)

    {

        cas[now]=ope;

        ope=(ope+1)%3;

**for**(ll i=0;i<field[now].size();i++)

            dfs(field[now][i],ope);

    }

}

int32\_t main()

{

    IOS;

    ll n;

    cin>>n;

**for**(ll i=1;i<=n;i++) cin>>val[0][i];

**for**(ll i=1;i<=n;i++) cin>>val[1][i];

**for**(ll i=1;i<=n;i++) cin>>val[2][i];

**bool** flag=1;

**for**(ll i=1;i<n;i++)

    {

        ll u,v;

        cin>>u>>v;

        field[v].pb(u);

        field[u].pb(v);

**if**(field[v].size()>2) flag=0;

**if**(field[u].size()>2) flag=0;

    }

**if**(flag)

    {

        ll ans=llINF;

        ll tar=-1;

**for**(ll i=1;i<=n;i++)

        {

**if**(field[i].size()==1) tar=i;

            cas[i]=-1;

        }

        dfs(tar,0);

        ll col=-1;

**for**(ll i=0;i<6;i++)

        {

            ll temp=0;

**for**(ll j=1;j<=n;j++)

                temp+=val[color[i][cas[j]]-1][j];

**if**(temp<ans)

            {

                ans=temp;

                col=i;

            }

        }

        cout<<ans<<endl;

**for**(ll i=1;i<=n;i++)

        {

**if**(i>1) cout<<' ';

            cout<<color[col][cas[i]];

        }

        cout<<endl;

    }

**else** cout<<-1<<endl;

}