Tree difference

You are given a tree with NN nodes (numbered 11 through NN). For each valid ii, node ii has a value AiAi.

You should answer QQ queries. In each query:

* You are given two nodes aa and bb.
* Let SS denote the set of all nodes on the simple path between the nodes aa and bb (including these nodes).
* Find the minimum value of |Ax−Ay||Ax−Ay| over all pairs x,y∈Sx,y∈S such that x≠yx≠y.

**Input**

* The first line of the input contains a single integer TT denoting the number of test cases. The description of TT test cases follows.
* The first line of each test case contains two space-separated integers NN and QQ.
* The second line contains NN space-separated integers A1,A2,…,ANA1,A2,…,AN.
* Each of the next N−1N−1 lines contains two space-separated integers uu and vv denoting that nodes uu and vv are connected by an edge.
* Each of the last QQ lines contains two space-separated integers aa and bb describing a query.

**Output**

For each query, print a single line containing one integer ― the answer to the query.

**Constraints**

* 1≤T≤201≤T≤20
* 2≤N≤2⋅1052≤N≤2⋅105
* 1≤Q≤2⋅1051≤Q≤2⋅105
* 1≤Ai≤1001≤Ai≤100 for each valid ii
* 1≤u,v,a,b≤N1≤u,v,a,b≤N
* a≠ba≠b
* the sum of NN over all test cases does not exceed 5⋅1055⋅105
* the sum of QQ over all test cases does not exceed 5⋅1055⋅105

**Subtasks**

**Subtask #1 (30 points):** N,Q≤1,000N,Q≤1,000

**Subtask #2 (70 points):** original constraints

**Example Input**

1

6 3

10 2 5 6 5 6

1 2

2 3

2 4

1 5

5 6

5 6

3 5

1 4

**Example Output**

1

0

4

**Explanation**

**Example case 1:**

* For the first query, S={5,6}S={5,6}, so the answer is |A5−A6|=1|A5−A6|=1.
* For the second query, S={3,2,1,5}S={3,2,1,5} and we can see that |A3−A5|=0|A3−A5|=0.
* For the third query, S={1,2,4}S={1,2,4}.
* Solution:

#include<bits/stdc++.h>

#define pb push\_back

#define int long long int

using namespace std;

vector<int> adj[200001];

int h[200001],p[200001][18];

void dfs(int node,int par)

{

h[node]=h[par]+1;

p[node][0]=par;

for(int child: adj[node])

{

if(child==par)

continue;

dfs(child,node);

}

}

int ar[200001];

int solve(int a,int b)

{

int freq[105];

memset(freq,0,sizeof(freq));

while(a != b)

{

if(h[a] > h[b])

{

freq[ar[a]]++;

if(freq[ar[a]]>1)

return 0;

a = p[a][0];

}

else if(h[a] < h[b])

{

freq[ar[b]]++;

if(freq[ar[b]]>1)

return 0;

b = p[b][0];

}

else

{

freq[ar[a]]++;

if(freq[ar[a]]>1)

return 0;

freq[ar[b]]++;

if(freq[ar[b]]>1)

return 0;

a = p[a][0];

b = p[b][0];

}

}

freq[ar[a]]++;

if(freq[ar[a]]>1)

return 0;

int prev1 = -200;

int res = 105;

for (int i = 1; i <= 100; i++) {

if (freq[i]) {

res = min(res, i - prev1);

prev1 = i;

}

}

return res;

}

int lca(int u,int v)

{

if(h[u]<h[v])

swap(u , v);

int diff = h[u] - h[v];

for(int i = 0;i < 18;i++)

{

if(diff&(1<<i))

u = p[u][i];

}

if(u == v) return u;

for(int i = 17;i >= 0;i--)

{

if(p[u][i] != p[v][i])

u = p[u][i],

v = p[v][i];

}

return p[u][0];

}

void buildp(int n)

{

for(int i=1;i<18;i++)

{

for(int u=1;u<=n;u++)

{

p[u][i]=p[p[u][i-1]][i-1];

}

}

}

main()

{

ios\_base::sync\_with\_stdio(false);

cin.tie(NULL);

cout.tie(NULL);

int t;

cin>>t;

while(t--)

{

for(int i=0;i<200001;i++)

adj[i].clear();

memset(h,0,sizeof(h));

int n,q;

cin>>n>>q;

int i;

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

cin>>ar[i];

memset(p,0,sizeof(p));

int root=-1;

for(i=0;i<n-1;i++)

{

int u,v;

cin>>u>>v;

if(root==-1)

root=u;

adj[u].pb(v);

adj[v].pb(u);

}

dfs(root,0);

buildp(n);

while(q--)

{

int a,b;

cin>>a>>b;

cout<<solve(a , b)<<"\n";

}

}

}