WORKSHEET 6

Student Name: Hari Krishnan Nair
UID: 20BCS2000
Branch: CSE
Section: 903 (DWWC- 43)

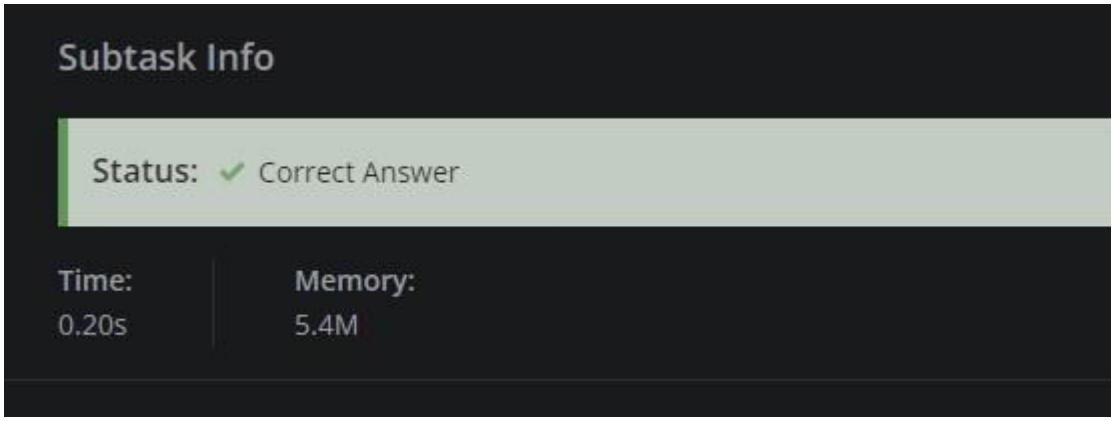
PROBLEM-1. Shortest Path in Binary Trees

CODE:

```
#include <iostream>

using namespace

std; int main() {  int

n;

 cin>>n;

while(n--){

long long a, b;

cin>>a>>b;

    int count  =0;

while(a!=b){

        if (a>b) {

a /= 2;

count++;
```

```
        }else if(b>a){
           b/=2;
count++;
        }
    }
    cout<<count<<endl;
 } return
0;
}
```

OUTPUT:



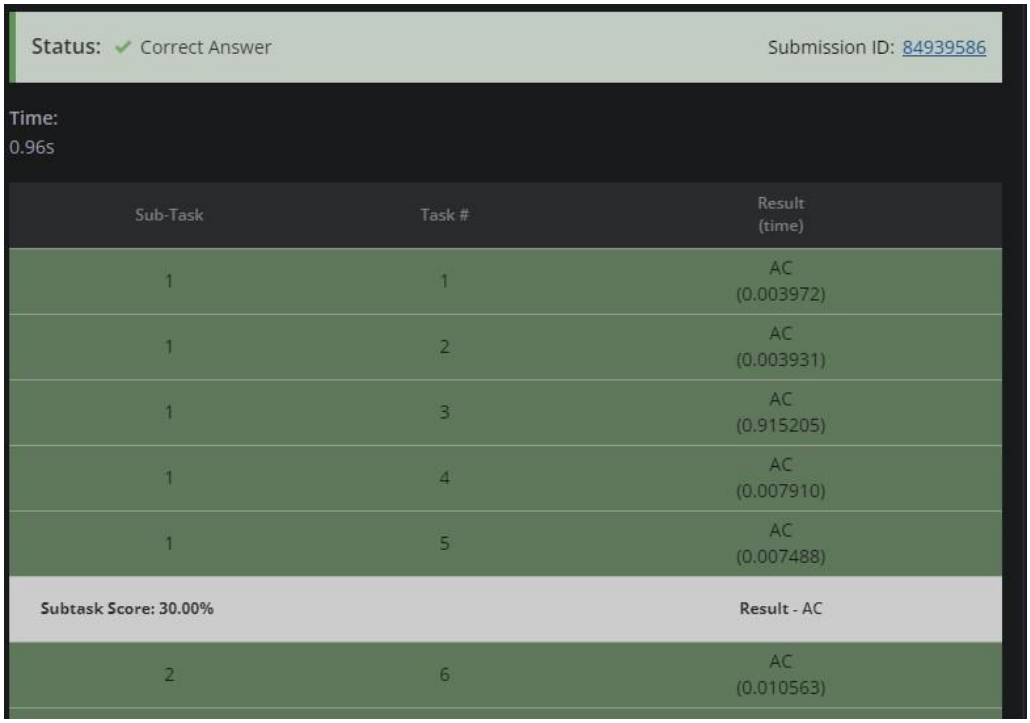
PROBLEM-2. Subtree Removal **Code:**

```
#include <iostream>
#include <bits/stdc++.h>
using namespace std;

vector <long> value;
long x;
long max(long a,long b)
```

```
{
if(a>=b)
{
return a;
}
else
{
return b;
}
}
long dfs(vector <long>v[],long a,long b)
{    long    i;    long
cal=value[a-1];
for(i=0; i<v[a].size(); i++)
{
if(v[a][i]!=b)
{
cal+=dfs(v,v[a][i],a);
}
}
return max(cal,-x);
}
int main() {
long t;
cin>>t;
while(t--)
{
value.clear(); long n;
cin>>n>>x; vector
<long> v[n+1]; long
i; long a,b; for(i=1;
i<=n; i++)
{
cin>>a;
value.push_back(a);
```

```
}
for(i=0; i<n-1; i++)
{
cin>>a>>b;
v[a].push_back(b);
v[b].push_back(a);
}
long cal=dfs(v,1,-1);
printf("%ld\n",cal);
}
}
```

OUTPUT:



Status: ✔ Correct Answer — Submission ID: 84939586

Time: 0.96s

Sub-Task	Task #	Result (time)
1	1	AC (0.003972)
1	2	AC (0.003931)
1	3	AC (0.915205)
1	4	AC (0.007910)
1	5	AC (0.007488)
Subtask Score: 30.00%		Result - AC
2	6	AC (0.010563)

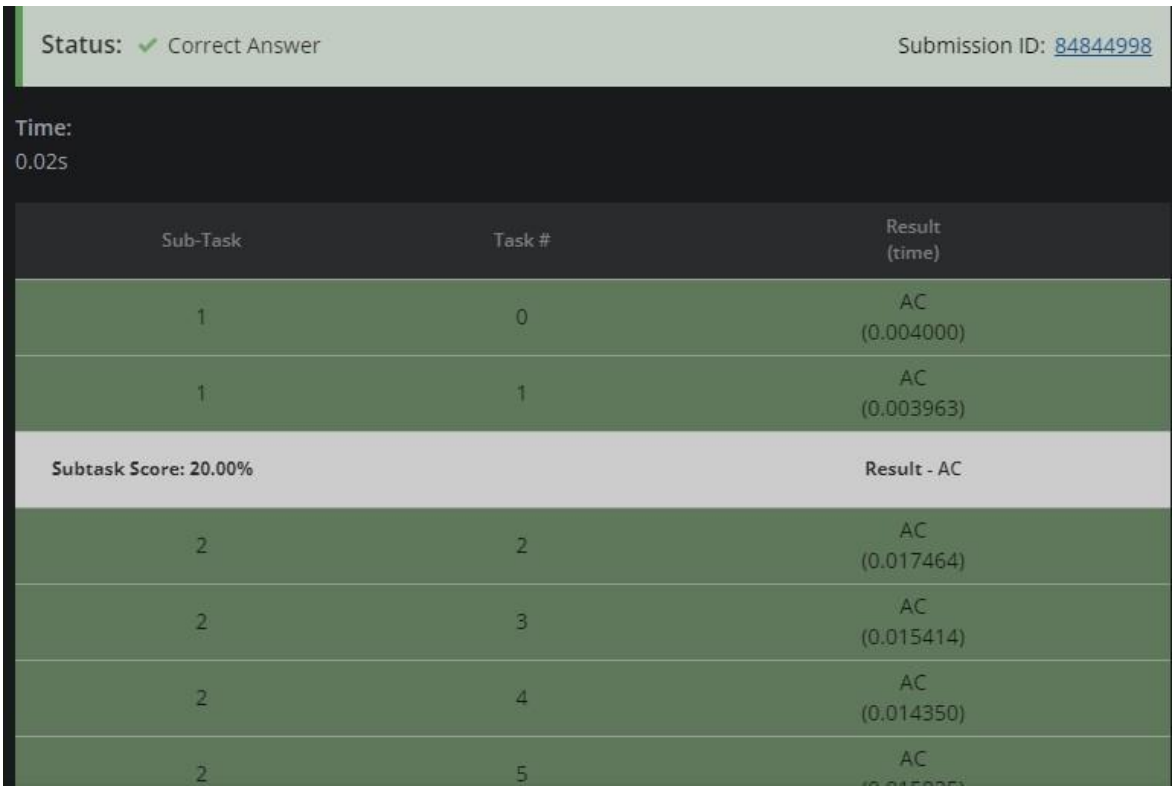
PROBLEM-3. Black and White Tree

Code:

```
#include "bits/stdc++.h" using
namespace std;
#define fast ios_base::sync_with_stdio(false);cin.tie(0);cout.tie(0);
#define tt int ct;cin>>ct;while(ct--)
#define MAX 100005 const
int mod = 998244353;
typedef long long ll; int
n,a[MAX]; vector<int>
ad[MAX]; int dp[MAX][2][2];
void dfs(int u,int par){
for(auto it : ad[u]){ if(it !=
par){ dfs(it,u);
}
}
for(int i=0;i<2;i++){ for(int
j=0;j<2;j++){ int odd = a[u]^i^j;
int cc = j; int dp1[2],dp2[2];
memset(dp2,0x3f,sizeof(dp2))
;
dp2[0] = 0; for(auto
v : ad[u]){
if(v != par){ swap(dp1[0],dp2[0]);
swap(dp1[1],dp2[1]);
memset(dp2,0x3f,sizeof(dp2));

dp2[0] = min(dp2[0],dp1[0]+dp[v][cc][0]);
dp2[1] = min(dp2[1],dp1[1]+dp[v][cc][0]);

dp2[0] = min(dp2[0],dp1[1]+dp[v][cc][1]);
dp2[1] = min(dp2[1],dp1[0]+dp[v][cc][1]);
} }
dp[u][i][j] = dp2[odd]+j;
} }
return;
}
```

```
void solve(){
cin>>n; for(int
i=0;i<n;i++){
cin>>a[i];
}

for(int i=0;i<n;i++){ ad[i].clear();
}
for(int i=0;i<n-1;i++){ int
u,v; cin>>u>>v; u--;v--;
ad[u].emplace_back(v);
ad[v].emplace_back(u);
}

dfs(0,-1);

int ans = min(dp[0][0][0],dp[0][0][1]);
if(ans > n){ printf("-1\n");
}
else{
printf("%d\n",ans);
}
return;
}
int32_t main() {
fast
#ifndef ONLINE_JUDGE
freopen("input.txt","r",stdin);
freopen("output.txt","w",stdout);
#endif
tt{
solve();
}
return 0;
}
```

OUTPUT:



Status: ✔ Correct Answer — Submission ID: 84844998

Time: 0.02s

Sub-Task	Task #	Result (time)
1	0	AC (0.004000)
1	1	AC (0.003963)
Subtask Score: 20.00%		Result - AC
2	2	AC (0.017464)
2	3	AC (0.015414)
2	4	AC (0.014350)
2	5	AC

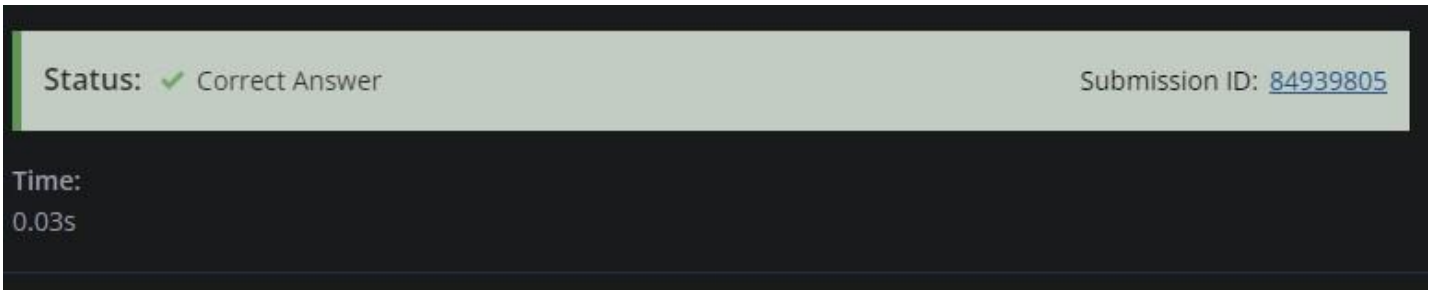
PROBLEM-4. Family Tree

CODE:

```
#include <bits/stdc++.h>
using namespace std;
#define ll long long
#define ld long double
#define pb push_back
#define pf push_front
#define mp make_pair
#define all(v) v.begin(), v.end()
#define test() int t; cin >> t; while(t--)
#define nl cout << endl
```

```
ll n, m, src, dst, cnt, r;
vector<ll> adj[100001]; ll
A[100001], B[100001]; ll
ln[100001], nn[100001];
ll res = -123456789;

void  dfs(ll  u){        for(int  i:
adj[u]){       ln[i] = max(ln[u],
A[i]);         nn[i] = min(nn[u],
A[i]);
        dfs(i);
    }
}

main(){
ios_base::sync_with_stdio(0);
cin.tie(0);                    cout.tie(0);
if(fopen("inp.inp",                  "r")){
freopen("inp.inp", "r", stdin);
        freopen("out.out", "w", stdout);
    }

    cin >> n;
    for(int i = 1; i <= n; i++) cin >>
A[i];     for(int i = 1; i <= n; i++){
cin >> B[i];         if(B[i] != -1)
adj[B[i]].pb(i);         else r = i;
    }
    ln[r] = -123456789;
nn[r] = 123456789;
    dfs(r);
    for(int i = 1; i <= n; i++) res = max(res, ln[i] - nn[i]);
cout << res;
}
```

OUTPUT:



PROBLEM-5. Common Ancestors

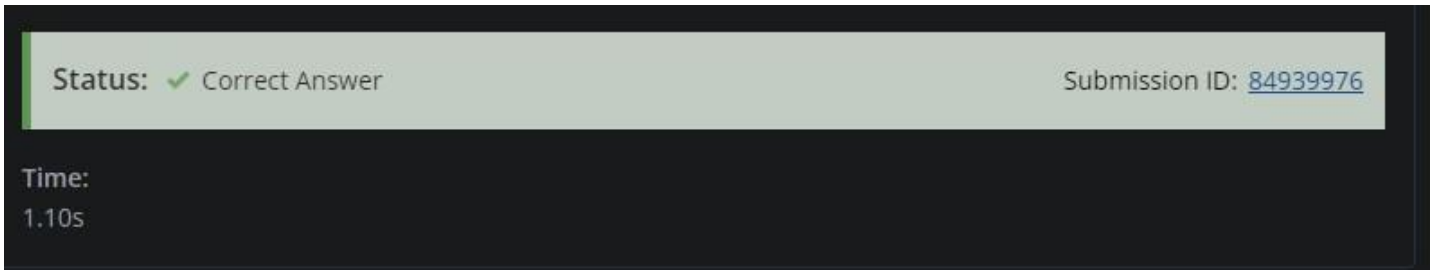
Code:

```
#include <bits/stdc++.h>

using namespace std;

const int MX = 1e6 + 10;

int seg[4 * MX], lazy[4 * MX], beg[MX], fin[MX], dep[MX], cnt; vector<int>
adj[2][MX];

void updateNode(int idx, int v) {
    seg[idx] += v;
    lazy[idx] += v;

    return;
}

void shift(int idx, int st, int ed) {
int lft = 2 * idx, rgt = lft + 1;

    if (lazy[idx]) {
        updateNode(lft, lazy[idx]);
        updateNode(rgt, lazy[idx]);
```

```
        lazy[idx] = 0;
    }

    return;
}

void update(int s, int e, int v, int idx = 1, int st = 0, int ed = cnt - 1) {
if (s > e || e < st || s > ed) return;

    if (s == st && e == ed) {
        updateNode(idx, v);

return;
    }

    int lft = 2 * idx, rgt = lft + 1, mid = (st + ed) / 2;

    shift(idx, st, ed);

    update(s, min(e, mid), v, lft, st, mid), update(max(s, mid + 1), e, v, rgt, mid + 1,
ed);

    seg[idx] = max(seg[lft], seg[rgt]);

    return;
}

void dfs0(int u, int d) {
beg[u] = cnt++;
dep[u] = d;

    for (auto v : adj[0][u]) dfs0(v, d + 1);

    fin[u] = cnt - 1;
```

```
    return;
}

void dfs1(int u, int d, int &ans) {
    if (dep[u] == d) update(beg[u], fin[u], 1);

    ans = max(ans, seg[1]);

    for (auto v : adj[1][u]) {
        dfs1(v, d + 1, ans);
    }

    if (dep[u] == d) update(beg[u], fin[u], -1);

    return;
}

int main() {
ios::sync_with_stdio(false);
    cin.tie(0);
cout.tie(0);

    int t;
    cin >> t;
        while
(t--) {
        cnt = 0;

int n;
        cin >> n;

        for (int i = 0; i <= 4 * n; i++) {
            seg[i] = lazy[i] = 0;
```

```
                    if (i <= n)
{
adj[0][i].clear();
adj[1][i].clear();
          }
       }

       for (int j = 0; j < 2; j++) {
for (int i = 1; i <= n; i++) {
             int p;
cin >> p;

             adj[j][p != -1 ? p : 0].push_back(i);
          }
       }

dfs0(0, 0);

       int ans = 0;

       dfs1(0, 0, ans);
```




```
       cout << ans - 1 << endl;
   }

   return 0;
}
```

OUTPUT:

PROBLEM-6 Lowest Common Ancestors

Code:

```
#include<bits/stdc++.h>

typedef long long ll; const
ll mod = 1e9 + 7; #define
ld long double

using namespace std;

class LCA{
    struct Euler{
        int vertex, height, index;
    };
    template<typename T>class LCASegmentTree{private:ll n;vector<T>dat;public:T
merge(T a,T b){if(a.height>b.height)return b;return
a;}LCASegmentTree(vector<T>v){int
_n=v.size();n=1;while(n<_n)n*=2;dat.resize(2*n1);for(int i=0;i<_n;i++)dat[n+i-
1]=v[i];for(int i=n-2;i>=0;i--
)dat[i]=merge(dat[i*2+1],dat[i*2+2]);} LCASegmentTree(int
_n){n=1;while(n<_n)n*=2;dat.resize(2*n-1); } void set_val(int i,T x){i+=n-
1;dat[i]=x;while(i>0){i=(i-1)/2;dat[i]=merge(dat[i*2+1],dat[i*2+2]);}}T query(int l,int
r){r++;T
left=T{INT_MAX,INT_MAX,INT_MAX},right=T{INT_MAX,INT_MAX,INT_MAX};l+=n-
1;r+=n-1;while(l<r){if((l&1)==0)left=merge(left,dat[l]);if((r&1)==0)right=merge(dat[r-
```

```
1],right);l=l/2;r=(r-1)/2;}return merge(left,right);}};
public:
    int n;
    vector<vector<int>> graph;
vector<bool>              visited;
vector<Euler> eulertour;
    vector<Euler> first;
    LCASegmentTree<Euler> *seg;
LCA(vector<vector<int>> graph){
        this->graph = graph;
this->n = graph.size();
visited.resize(n);
first.resize(n);
        this->makeEuler();
    }

    // Euler tour of tree
    void makeEuler(int root = 0){
        // Euler tour tạo ra verticies, heights, index
        std::fill(visited.begin(), visited.end(), false);
int height =0;
        std::function<void(int)> explore = [&](int
u){             visited[u] = true;             height++;
            eulertour.push_back(Euler{u, height, (int) eulertour.size()});
for (auto v: this->graph[u]){
                if (!visited[v]) {
explore(v);                    height--
;
                    eulertour.push_back(Euler{u, height, (int) eulertour.size()});
                }
            }
        };
        explore(root);
// Tạo ra mảng first
```

```
        std::fill(visited.begin(), visited.end(), false);
for (auto e: eulertour){                    if
(!visited[e.vertex]){               visited[e.vertex] =
true;               first[e.vertex] = e;
            }
        }
        // Tạo 1 segment tree để query trên mảng height
this->seg = new LCASegmentTree<Euler>(eulertour);
    }

    int lca(int u, int v){        int uidx =
first[u].index;        int vidx =
first[v].index;        if (uidx > vidx)
swap(uidx, vidx);        Euler a = seg-
>query(uidx, vidx);
        return a.vertex;
    }
    /* Additional functionality*/
    // Trả về chiều cao của 1 đỉnh h[vertex] = v_height; - chiều cao bắt đầu từ 1->n
(1indexed)
    vector<ll>          height(){
vector<ll>    h(this->n,    0);
for (auto e: eulertour){
            h[e.vertex] = e.height;
        }
        return h;
    }

    int lca(int r, int u, int v){ // ar = abtrary root - LCA của u,v với r bất kỳ là
root        int ru = lca(r, u);        int rv = lca(r, v);        int uv = lca(u, v);        if
(ru == rv) return uv;        if (ru == uv) return rv;        return ru;
    }
};

int main(){
```

```
    ios::sync_with_stdio(0);
    cin.tie(0);
    #ifdef DEBUG
        freopen("inp.txt", "r", stdin);
freopen("out.txt", "w", stdout);
    #endif     int N;     cin >> N;
vector<vector<int>> adj(N);
for (int i=0;i<N-1;i++){
        int u, v;
cin >> u>> v;
        u--; v--;
        adj[u].push_back(v);
adj[v].push_back(u);
    }
    LCA lca(adj);

    int q;
cin >>q;
    for (int i=0;i<q;i++){
int r, u, v;         cin >>
r>> u>> v;         r--;u--
;v--;
        cout << lca.lca(r, u, v) +1 <<'\n';
    }
}
```

OUTPUT:

Status: ✔ Correct Answer Submission ID: 84940174

Time:
0.32s

Sub-Task	Task #	Score	Result (time)
1	0	NA	AC (0.003687)
1	1	NA	AC (0.003889)
Final Score - 20.000000			Result - AC
2	2	NA	AC (0.004688)
2	3	NA	AC (0.005614)
2	4	NA	AC (0.014689)
2	5	NA	AC (0.137339)

PROBLEM-7. Cosmic temple

Code:

```
#include<bits/stdc++.h> using
namespace std; typedef long
long ll; #define f(j,a,b) for(int
j=a;j<b;j++)

void bfs(int s,int n,vector<int>adj[])
{
```

```
    vector<int>dist(n,0);
vector<bool>visited(n,0);
visited[s]=1;
queue<int>q;     q.push(s);
    while(q.empty()==0)
    {      int
x=q.front();
q.pop();
     for(auto y:adj[x])
     {
if(visited[y]==0)
        {
dist[y]=dist[x]+1;
visited[y]=1;
q.push(y);
        }
     }
   }
  int k=0;
 for(auto g:dist)
{
   k+=g;
}
cout<<k<<" ";


}

int main()
{
 ios::sync_with_stdio(0);
 cin.tie(0);
  int n;  cin>>n;
vector<int>adj[n];
 f(j,0,n-1)
```

```
{   int
x,y;
  cin>>x>>y;
adj[x].push_back(y);
adj[y].push_back(x);
 }
f(j,0,n)
 {
   bfs(j,n,adj);
  }
cout<<endl;
return 0;
}
```

OUTPUT:

PROBLEM-8. Secret Tree

Code:

```
#include <bits/stdc++.h>
#define endl '\n'
#define PRECISION 9
using namespace std;
using ll = long long;
using ld = long double;
#define fr first
#define sc second using pi2 =
pair<int, int>; using pl2 =
```

```
pair<ll, ll>; #define all(v)
v.begin(), v.end()
#define unq(v) sort( all(v) ); v.erase( unique( all(v) ), v.end() );

vector<int> adj[120]; int cnt[120];

void Main(){
 int t; cin >> t;
 while (t--){
      int n; cin >> n;       for (int
i = 2; i <= n; i++){
            for (int j = 2; j <= n; j++){
                  if (i==j){ continue; }
                  cout << "? " << 3 << ' ' << 1 << ' ' << i << ' ' << j << endl << flush;
                  int res; cin >> res;
                  if (res){ adj[i].push_back(j); cnt[j] += 1; }
            }
      }
      for (int i = 2; i <= n; i++){ adj[1].push_back(i); cnt[i] += 1; }
      queue<int> q; q.push(1);           vector<pi2> v;       while
(!q.empty()){              int now = q.front(); q.pop();              for
(int nxt : adj[now]){
                  //cout << "EDG " << now << ' ' << nxt << endl << flush;
                  cnt[nxt] -= 1;
                  if (cnt[nxt] == 0){ v.push_back({now, nxt}); q.push(nxt); }
            }
      }
      cout << "!" << endl << flush;
 for (pi2 p : v){ cout << p.fr << ' ' << p.sc << endl << flush; }  cout <<
flush;
      for (int i = 1; i <= n; i++){ adj[i].clear(); cnt[i] = 0; }
 }
}
```

```
int main(){
 ios_base::sync_with_stdio(0);
 cin.tie(0); cout.tie(0);
cout.setf(ios::fixed);
cout.precision(PRECISION);
 Main();
}
```

OUTPUT: