```cpp
/*
//////////////////////////
vis[u] = descovery time of u
low[u] = ...
comp[u] = the number of componenet where u belongs to in main tree.
bicon[u] = the biconnected component number where u belongs to.
tree[] = will store the bridge tree.

Be careful about multiple edges.
//////////////////////////
*/

const int maxn = 2e5 + 10;
vector<int> adj[maxn], tree[maxn];

int vis[maxn], low[maxn], tym = 1, c = 0;
int comp[maxn], bicon[maxn];

void calc(int u, int par, int c) {
    comp[u] = c;
    vis[u] = low[u] = tym++;
    for(int v : adj[u]) {
        if(vis[v]) {
            if(v != par) low[u] = min(low[u], vis[v]);
            else par = -1;  // This handles multiple edges.
        } else {
            calc(v, u, c);
            low[u] = min(low[u], low[v]);
        }
    }
}
void shrink(int u, int now) {
    bicon[u] = now;
    for(int v : adj[u]) if(!bicon[v]) {
        if(low[v] > vis[u]) {
            tree[now].push_back(c);
            shrink(v, c++);
        } else shrink(v, now);
    }
}
// Lca Build + Stuff here
int main(int argc, char const *argv[]) {
    // Take input here
    c = 1;
    for(int i = 1; i <= n; i++)
        if(!vis[i]) calc(i, 0, c++);
    c = 1;
    vector<int> root;
    for(int i = 1; i <= n; i++) if(!bicon[i]) {
        root.push_back(c);
        shrink(i, c++);
    } tym = 1;
    for(int r : root) build(r, 0);
    // Do stuff.
}
```