Cycle of Edges

Submitted 10 minutes ago • Score: 25.00 Status: Accepted

✔ Test Case #0	✔ Test Case #1	✔ Test Case #2
✔ Test Case #3	✔ Test Case #4	✔ Test Case #5
✔ Test Case #6	✔ Test Case #7	✔ Test Case #8
✔ Test Case #9	✔ Test Case #10	✔ Test Case #11
✔ Test Case #12	✔ Test Case #13	✔ Test Case #14
✔ Test Case #15	✔ Test Case #16	✔ Test Case #17
✔ Test Case #18	✔ Test Case #19	✔ Test Case #20
✔ Test Case #21	✔ Test Case #22	

Submitted Code

Language: C++20 Open in editor

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

//DSU_UNION_BY_SIZE
const int N = 1e5+5;
int parent[N];
int parentSize[N];

void dsu_set(int n)
{
    for(int i = 1; i<=n; i++)
    {
        parent[i] = i;
        parentSize[i] = 1;
    }
}

int dsu_find(int node)
{
    while(parent[node] != node)
    {
        node = parent[node];
    }
    return node;
}
```

```cpp

// This is union by size
void dsu_union(int a, int b)
{
    int leaderA = dsu_find(a);
    int leaderB = dsu_find(b);
    if(leaderA != leaderB)
    {
        if(parentSize[leaderA] > parentSize[leaderB])
        {
            // Leader is 'A'
            parent[leaderB] = leaderA;
            parentSize[leaderA] += parentSize[leaderB];
        }

        else
        {
            // Leader is 'B'
            parent[leaderA] = leaderB;
            parentSize[leaderB] += parentSize[leaderA];
        }
    }
}


int cnt = 0;
int main()
{

    int n , m;
    cin >> n >> m;
    dsu_set(n);
    for(int i = 1; i<=m; i++)
    {
        int u, v;
        cin>> u >> v ;
        int leaderA = dsu_find(u);
        int leaderB = dsu_find(v);
        if(leaderA == leaderB)
        cnt++;
        dsu_union(u,v);
    }

    cout<<cnt<<endl;

    return 0;
}
```