Can Go Again?

Submitted an hour 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	✔ Test Case #23
✔ Test Case #24	✔ Test Case #25	✔ Test Case #26
✔ Test Case #27	✔ Test Case #28	✔ Test Case #29
✔ Test Case #30	✔ Test Case #31	✔ Test Case #32
✔ Test Case #33	✔ Test Case #34	✔ Test Case #35
✔ Test Case #36	✔ Test Case #37	✔ Test Case #38
✔ Test Case #39	✔ Test Case #40	✔ Test Case #41
✔ Test Case #42	✔ Test Case #43	✔ Test Case #44
✔ Test Case #45	✔ Test Case #46	✔ Test Case #47
✔ Test Case #48	✔ Test Case #49	✔ Test Case #50
✔ Test Case #51	✔ Test Case #52	✔ Test Case #53
✔ Test Case #54	✔ Test Case #55	✔ Test Case #56
✔ Test Case #57		

Submitted Code

Language: C++20 Open in editor

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 #define pii pair<pair<int, int>, int>
4
5 const int N = 1e3 + 13;
6 int n, m;
```

```
const long long INF = 1e18 + 8;
vector<pii> edgeList;
long long dist[N];

void initializeDistance()
{
    for (int i = 1; i <= n; i++)
    {
        dist[i] = INF;
    }
}

void BellmanFord(int src)
{
    initializeDistance();

    dist[src] = 0;

    for (int i = 1; i < n; i++)
    {
        for (auto el : edgeList)
        {
            long long u = el.first.first;
            long long v = el.first.second;
            long long w = el.second;

            if (dist[u] != INF and dist[u] + w < dist[v])
            {
                dist[v] = dist[u] + w;
            }
        }
    }
}

int main()
{

    cin >> n >> m;
    for (int i = 1; i <= m; i++)
    {
        long long u, v, w;
        cin >> u >> v >> w;
        edgeList.push_back({{u, v}, w});
    }

    // for(auto el : edgeList)
    // {
    //     cout<<el.first.first<<" "<<el.first.second<<": "<<el.second<<endl;
    // }

    int src;
    cin >> src;
    BellmanFord(src);

    bool cycle = false;
    for (auto el : edgeList)
    {
        long long u = el.first.first;
        long long v = el.first.second;
        long long w = el.second;

        if (dist[u] != INF and dist[u] + w < dist[v])
        {
            cycle = true;
            break;
            dist[v] = dist[u] + w;
```

```
        }
    }

    int q;
    cin >> q;
    while (q--)
    {
        int dest;
        cin >> dest;
        if (cycle == false)
        {
            if (dist[dest] == INF)
            {
                cout << "Not Possible" << endl;
            }
            else
            {
                cout << dist[dest] << endl;
            }
        }
    }

    // for (int i = 1; i <= n; i++)
    // {
    //     // cout << "D of " << i << ": " << dist[i] << endl;
    // }
    // cout<<"cycle = "<<cycle<<endl;
    if (cycle)
        cout << "Negative Cycle Detected" << endl;

    return 0;
}
```