Can Go Again?

Problem Statement

You will be given N numbers of nodes, E numbers of edges in a graph. For each edge you will be given A, B and W which means there is a connection from A to B only and for which you need to give W cost. The value of nodes could be from 1 to N.

You will be given a source node S. Then you will be given a test case T, for each test case you will be given a destination node D. You need to tell the minimum cost from source node to destination. If there is no possible path from S to D then print **Not Possible**.

Note: If there is a negative weight cycle in the graph, then no answer would be correct. So print one line only - "**Negative Cycle Detected**".

Input Format

- First line will contain N and E.
- Next E lines will contain A, B and W.
- Next line will contain source node S.
- Next line will contain T, the number of test cases.
- For each test case, you will get D.

Constraints

1. $1 \leq N \leq 10^3$
2. $1 \leq E \leq 10^6$
3. $1 \leq S \leq N$
4. $1 \leq T \leq 10^3$
5. $1 \leq D \leq N$
6. $-10^9 \leq W \leq 10^9$

Output Format

- Output the minimum cost for each test case.

Sample Input 0

```
5 7
1 2 10
1 3 -2
3 2 1
2 4 7
3 4 -3
```

```
4 5 5
2 5 2
1
5
1
2
3
4
5
```

Sample Output 0

```
0
-1
-2
-5
0
```

Sample Input 1

```
5 7
1 2 10
1 3 -2
3 2 1
2 4 7
3 4 -3
4 5 5
2 5 2
5
5
1
2
3
4
5
```

Sample Output 1

```
Not Possible
Not Possible
Not Possible
Not Possible
0
```

Sample Input 2

```
5 8
1 2 -2
1 3 -10
3 2 1
2 4 7
4 3 -3
4 5 5
2 5 2
4 1 1
1
5
1
2
3
4
5
```

Sample Output 2

```
Negative Cycle Detected
```

Submissions: 315
Max Score: 20
Difficulty: Easy

Rate This Challenge:

More

C++20

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

#define lli long long int
const lli M = 1e18;
lli dis[1005];

class Edge{
    public:
    int u,v;
    lli c;
    Edge(int u, int v, lli c){
        this->u = u;
        this->v = v;
        this->c = c;
    }
};

int main(){

    int n,e;
    cin >> n >> e;
    vector<Edge> edgeList;
    while(e--){
        int u,v;
        lli c;
        cin >> u >> v >> c;
        edgeList.push_back(Edge(u,v,c));
    }

    for(int i = 1; i <= n; i++){
        dis[i] = M;
    }

    int src;
    cin >> src;
    dis[src] = 0;

    for(int i = 1; i <= n; i++){
        for(Edge child: edgeList){
            int u,v;
            lli c;
            u = child.u;
            v = child.v;
            c = child.c;
            if(dis[u] < M && dis[u]+c < dis[v]){
                dis[v] = dis[u] + c;
            }
        }
```

```
    }

    bool cycle = false;
    for(Edge child: edgeList){
        int u,v;
        lli c;
        u = child.u;
        v = child.v;
        c = child.c;
        if(dis[u] < M && dis[u]+c < dis[v]){
            cycle = true;
            break;
        }
    }

    int test;
    cin >> test;
    while(test--){
        if(cycle){
            cout << "Negative Cycle Detected" << endl;
            break;
        }

        int des;
        cin >> des;
        if(dis[des] < M){
            cout << dis[des] << endl;
        }else{
            cout << "Not Possible" << endl;
        }

    }

    return 0;
}
```

Line: 1 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code