Input:

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

int main(){
    // taking number of vertices and edges count from user
    int V;
    cout << "Enter number of vertices: ";
    cin >> V;
    int n_edges;
    cout << "Enter count of edges: ";
    cin >> n_edges;

    // declaring the edges vector and taking input from user
    vector<vector<int>> edges(n_edges, (vector<int>(3, 0)));
    cout << "From to weight: (enter in this format)\n";
    for(int i=0; i<n_edges; i++){
        cout << (i+1) << " edge: ";
        cin >> edges[i][0] >> edges[i][1] >> edges[i][2];
    }

    // declaring dist array and intialzing the distance of source as 0
    int dist[V];
    for(int i=0; i<V; i++) dist[i] = INT_MAX;
    dist[0] = 0;

    // calculating shortest path from 0 to all the vertices
    for(int i=0; i<V-1; i++){
        for(int j=0; j<n_edges; j++){
            int u = edges[j][0]-1, v=edges[j][1]-1;
            int weight = edges[j][2];

            if(dist[u]!=INT_MAX && dist[u]+weight < dist[v]){
                dist[v] = dist[u]+weight;
            }
        }
    }

    // checking once again to find if negative edge cycle is present or not
    for(int i=0; i<V-1; i++){
        bool flag = true;
        for(int j=0; j<n_edges; j++){
            int u = edges[j][0]-1, v=edges[j][1]-1;
            int weight = edges[j][2];
            if(dist[u]!=INT_MAX && dist[u]+weight < dist[v]){
                cout << "\n\nNegative edge cycle is present\n";
                flag = false;
                break;
            }
        }
        if(!flag) break;
    }
    cout << endl << endl;
    cout << "Vertex\tDistance from source\n";

    for(int i=0; i<V; i++){
        cout << i << "\t" <<dist[i] << "\n";
    }

    return 0;
}
```

Output:

Case 2:

```
@somesh4545 ➜ /workspaces/TE-Labs/DAA (main) $ g++ bellman_ford.cpp && ./a.out
Enter number of vertices: 7
Enter count of edges: 10
From to weight: (enter in this format)
1 edge: 1 2 6
2 edge: 1 3 2
3 edge: 1 4 5
4 edge: 2 5 -1
5 edge: 3 2 -2
6 edge: 3 5 1
7 edge: 4 3 -2
8 edge: 4 6 -1
9 edge: 5 7 3
10 edge: 6 7 3


Vertex  Distance from source
0       0
1       0
2       2
3       5
4       -1
5       4
6       2
```

Case 2:

```
@somesh4545 ➜ /workspaces/TE-Labs/DAA (main) $ g++ bellman_ford.cpp && ./a.out
Enter number of vertices: 4
Enter count of edges: 4
From to weight: (enter in this format)
1 edge: 1 2 1
2 edge: 2 3 2
3 edge: 3 4 3
4 edge: 4 1 4


Vertex  Distance from source
0       0
1       1
2       3
3       6
```

Case 3:

```
@somesh4545 ➜ /workspaces/TE-Labs/DAA (main) $ g++ bellman_ford.cpp && ./a.out
Enter number of vertices: 4
Enter count of edges: 5
From to weight: (enter in this format)
1 edge: 1 2 2
2 edge: 2 3 2
3 edge: 2 4 -2
4 edge: 3 4 1
5 edge: 4 1 -1


Negative edge cycle is present


Vertex  Distance from source
0       -3
1       0
2       2
3       -2
```

Case 4:

```
@somesh4545 ➜ /workspaces/TE-Labs/DAA (main) $ g++ bellman_ford.cpp && ./a.out
Enter number of vertices: 5
Enter count of edges: 8
From to weight: (enter in this format)
1 edge: 1 2 -1
2 edge: 1 3 4
3 edge: 2 3 3
4 edge: 2 5 2
5 edge: 4 3 5
6 edge: 5 4 -3
7 edge: 4 2 1
8 edge: 2 4 2


Vertex  Distance from source
0       0
1       -1
2       2
3       -2
4       1
```