**Assignment No. 2**

Code:

#include<bits/stdc++.h>

using namespace std;

vector<int> bellman\_ford(int n, int m, int src, vector<vector<int>> &vec)

{

vector<int> dist(n, 1e9);

dist[src] = 0;

for(int i=0; i<n-1; i++)

{

for(int j=0; j<m; j++)

{

int u = vec[j][0];

int v = vec[j][1];

int wt = vec[j][2];

if((dist[u]!=1e9) && ((dist[u]+wt)<dist[v]))

{

dist[v] = dist[u]+wt;

}

}

}

// Check for negative weight cycle

bool flag = false;

for(int j=0; j<m; j++)

{

int u = vec[j][0];

int v = vec[j][1];

int wt = vec[j][2];

if((dist[u]!=1e9) && ((dist[u]+wt)<dist[v]))

{

flag = true;

break;

}

}

if(flag==false) return dist;

vector<int> dist1;

dist1.push\_back(-1);

return dist1;

}

int main()

{

cout << "Enter the number of nodes and edges: ";

int nodes, edges; cin >> nodes >> edges;

vector<vector<int>> vec;

for(int i=0; i<edges; i++)

{

cout << "Enter u v and wt: ";

int u, v, wt; cin>> u >> v >> wt;

vector<int> temp;

temp.push\_back(u); temp.push\_back(v); temp.push\_back(wt);

vec.push\_back(temp);

}

vector<int> dist;

dist = bellman\_ford(nodes, edges, 0, vec);

cout << "Distance of every node from source" << endl;

for(auto &it: dist)

{

cout << it << " ";

}

cout << endl;

}

Output:

Enter the number of nodes and edges: 3 4

Enter u v and wt: 0 1 5

Enter u v and wt: 1 0 3

Enter u v and wt: 1 2 -1

Enter u v and wt: 2 0 1

Distance of every node from source

0 5 4