Assignment 7

[**ALGORITHMS LAB**](https://classroom.google.com/u/0/c/NDg4NjQ2MTUwMDM2)

**Implement the Bellman-Ford single source shortest path algorithm. Show that the algorithm indicates if there is a negative cycle. Find out the actual number of assignment operation and  comparison operation for 10 different random data sets.**

#include <bits/stdc++.h>

using namespace std;

#define Node 10

vector<pair<int, pair<int, int>>> adj;

vector<int> dis(Node + 1, 1e9);

int ct = 0;

void Bellman\_ford(int src)

{

  dis[src] = 0;

  for (int i = 1; i < Node; i++)

  {

    for (int j = 0; j < adj.size(); j++)

    {

      if (dis[adj[j].first] != 1e9 && dis[adj[j].second.first] > dis[adj[j].first] + adj[j].second.second)

      {

        ct += 2;

        dis[adj[j].second.first] = dis[adj[j].first] + adj[j].second.second;

      }

      ct += 1;

    }

  }

}

bool checkNegCycle()

{

  int flag = 0;

  for (int j = 0; j < adj.size(); j++)

  {

    if (dis[adj[j].first] != 1e9 && dis[adj[j].second.first] > dis[adj[j].first] + adj[j].second.second)

    {

      dis[adj[j].second.first] = dis[adj[j].first] + adj[j].second.second;

      flag = 1;

    }

  }

  if (flag)

    return true;

  return false;

}

int main()

{

  int edges;

  cin >> edges;

  for (int i = 0; i < edges; i++)

  {

    int u, v, w;

    cin >> u >> v >> w;

    adj.push\_back({u, {v, w}});

  }

  int src = 1;

  Bellman\_ford(1);

  if (checkNegCycle())

    cout << "Negative cycle is Present " << endl;

  else

    cout << "Negative cycle is not Present " << endl;

  for (int i = 1; i <= Node; i++)

  {

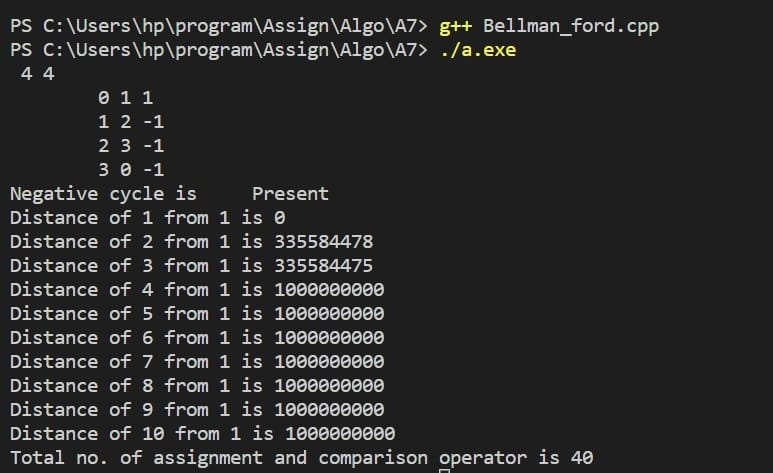
    cout << "Distance of " << i << " from " << src << " is " << dis[i] << endl;

  }

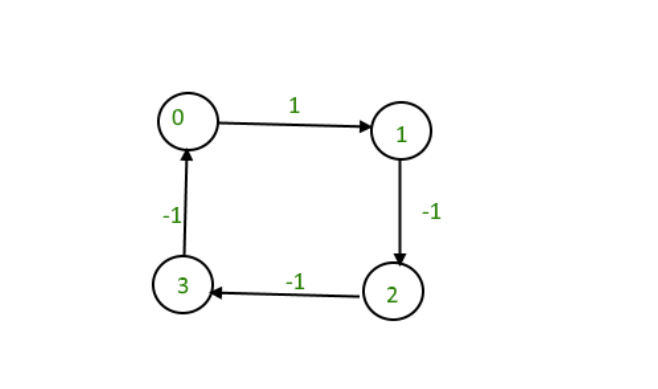
  cout << "Total no. of assignment and comparison operator is " << ct<< endl;

}

**OUTPUT**



**GRAPH**

****