Name : lakhan Kumawat

Roll No : 1906055

Branch : CSE -1

Course : CSL4403

Lab Experiment 09

To implement Bellman Ford’s Algorithm

**Program code : C++**

/\*\*

\* Descrption: BellmanFord (Finds the shortest path from source s to all vertices v. Detects a negative weight cycle if present.)

\* Usage: See below O(V.E) = O(n^2)

\*/

#include <iostream>

#include <vector>

#define INT\_MAX 10000

using namespace std;

int V,E;

struct edge{

int src,dst,wt;

};

void bellmanFord(vector<edge>& Edges){

int parent[V]; //To store parent node

//int cost\_parent[V]; //Cost of node to parent

vector<int> distance(V,INT\_MAX);

parent[0]=-1;

distance[0]=0;

bool update;

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

update=false;

for(int j=0;j<E;j++){

int U=Edges[j].src;

int V=Edges[j].dst;

int wt=Edges[j].wt;

if(distance[U]!=INT\_MAX && distance[U]+wt<distance[V]){

distance[V]=distance[U]+wt;

parent[V]=U;

//cost\_parent[V]=distance[V];

update=true;

}

}

if(update==false)

break;

}

//Now let's check it has negative edge cycle or not by relaxing one more time

for(int j=0;j<E && update==true;j++){

int U=Edges[j].src;

int V=Edges[j].dst;

int wt=Edges[j].wt;

if(distance[U]!=INT\_MAX && distance[U]+wt<distance[V]){

cout<<"Graph has negative Edge Cycle :( "<<endl;

return;

}

}

//Print Shortest path

for(int i=1;i<V;i++){

cout<<"U->V: 0->"<<i<<" Cost to reach "<<i<<" from source 0 = "<<distance[i]<<endl;

}

}

int main(){

cin>>V>>E; //no of vertices and edges

vector<edge> Edges(E);

int src,dst,wt;

for(int i=0;i<E;i++){

cin>>src>>dst>>wt;

Edges[i].src=src;

Edges[i].dst=dst;

Edges[i].wt=wt;

}

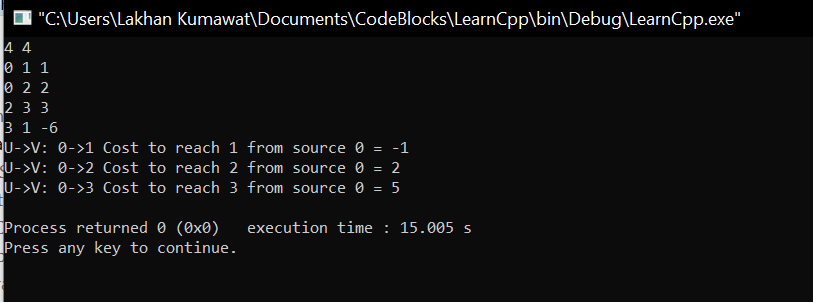
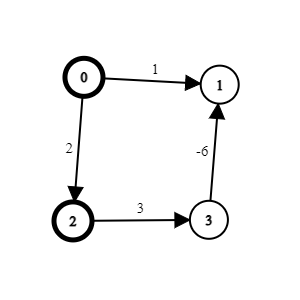
bellmanFord(Edges);

return 0;

}

Output : with no negative edge cycle

Input Graph :



Output : with negative edge cycle

Input Graph :

