//Single Source Shortest Path (Negative Cycle)

//Complexity : O(VE)

#define MAX 1000

#define INF 1e7

vector<int>G[MAX], W[MAX];

int V, E, dist[MAX];

void bellmanFord() {

//set to -INF if max distance is needed

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

dist[i] = INF;

//relax all edges V-1 times

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

for(int u = 0; u < V; u++) //all the nodes

for(int j = 0; j < (int)G[u].size(); j++) {

int v = G[u][i];

int w = W[u][i];

//relax edges

//set to max if max value needed

if(dist[u] != INF) //if there is a negative weight, then INF + negative weight < INF and INF becomes +-INF

dist[v] = min(dist[v], dist[u]+w);

}

}

bool hasNegativeCycle() {

for(int u = 0; u < V; u++)

for(int i = 0; i < G[u].size(); i++) {

int v = G[u][i];

int w = W[u][i];

//if bellmanFord is run for max values, then this code will return true for positive cycle by adding this line

//if(dist[v] < dist[u] + w)

if(dist[v] > dist[u] + w)

return 1;

}

return 0;

}