A picture containing text

Description automatically generated



Department of Software Engineering

Bahria University, Islamabad

Data Structure & Algorithm Lab

(spring - 2025)

Teacher: Madam Raheela

Name: Muhammad Shoaib Akhtar

Enrollment No: 01-131232-067

LAB 12

**Comments:**

**Signature**

**Task 1:**

**Code:**

#include <iostream>

using namespace std;

#define MAXVERTEXS 50

#define TRUE 1

#define FALSE 0

struct Edge {

int adj; // TRUE if edge exists, else FALSE

int weight; // weight of the edge

};

class WeightedGraph {

private:

Edge edges[MAXVERTEXS][MAXVERTEXS];

int numVertices; // number of vertices in the graph

public:

WeightedGraph(int n) {

if (n > MAXVERTEXS) {

cout << "Number of vertices exceeds maximum allowed (" << MAXVERTEXS << ")." << endl;

numVertices = MAXVERTEXS;

}

else {

numVertices = n;

}

// Initialize adjacency matrix - no edges at start

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

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

edges[i][j].adj = FALSE;

edges[i][j].weight = 0;

}

}

}

void joinwt(int v1, int v2, int wt) {

if (v1 >= 0 && v1 < numVertices && v2 >= 0 && v2 < numVertices) {

edges[v1][v2].adj = TRUE;

edges[v1][v2].weight = wt;

}

else {

cout << "Invalid vertices for joinwt: " << v1 << ", " << v2 << endl;

}

}

void remv(int v1, int v2) {

if (v1 >= 0 && v1 < numVertices && v2 >= 0 && v2 < numVertices) {

edges[v1][v2].adj = FALSE;

edges[v1][v2].weight = 0;

}

else {

cout << "Invalid vertices for remv: " << v1 << ", " << v2 << endl;

}

}

bool adjacent(int v1, int v2) {

if (v1 >= 0 && v1 < numVertices && v2 >= 0 && v2 < numVertices) {

return (edges[v1][v2].adj == TRUE);

}

else {

cout << "Invalid vertices for adjacent check: " << v1 << ", " << v2 << endl;

return false;

}

}

void printGraph() {

cout << "Adjacency Matrix (weights shown, 0 means no edge):\n";

cout << " ";

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

cout << j << " ";

}

cout << endl;

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

cout << i << ": ";

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

if (edges[i][j].adj == TRUE)

cout << edges[i][j].weight << " ";

else

cout << "0 ";

}

cout << endl;

}

}

};

int main() {

// Create a weighted graph with 5 vertices

WeightedGraph g(5);

// Add some edges with weights

g.joinwt(0, 1, 10);

g.joinwt(0, 4, 20);

g.joinwt(1, 2, 30);

g.joinwt(3, 4, 40);

// Print adjacency matrix

g.printGraph();

// Check adjacency

cout << "Is 0 adjacent to 1? " << (g.adjacent(0, 1) ? "Yes" : "No") << endl;

cout << "Is 1 adjacent to 4? " << (g.adjacent(1, 4) ? "Yes" : "No") << endl;

// Remove an edge

g.remv(0, 1);

cout << "After removing edge 0->1:\n";

g.printGraph();

return 0;

}

**SCREENSHOT:  
A computer screen with numbers and symbols

AI-generated content may be incorrect.**