

Lab-12

Lab-12



**Name: Talib Husain**

**Roll# 21F-9070**

**Task-1**

#include<iostream>

using namespace std;

//making a class ADT

class Graph{

    int \*\*adm;

    int row,col;

public:

    //Constructor

    Graph(int r = 5,int c =5){

        row = r;

        col = c;

        //Allocating 2-d array

            adm = new int\*[row];

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

                \*(adm+i) = new int[col];

            }

            for (int i=0;i<row;i++){

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

                    adm[i][j] = 0;

                }

            }

    }

    //printing a matrix

    void print(){

    cout<<"Adjacent Matrix\n";

        cout<<"   ";

    for (int i=0;i<col;i++){

        cout<<i<<" ";

    }

        cout<<"\n  ";

    for (int i=0;i<col;i++)

        cout<<"--";

        cout<<"\n";

    //travesiong a matrix

    for (int i=0;i<row;i++){

        cout<<i<<" |";

        for (int j=0;j<col;j++){

            cout<<adm[i][j]<<" ";

        }

        cout<<"\n";

    }

    }

    void addedge(int i, int j){

        adm[i][j] = 1;

    }

 ~Graph(){

        for (int i = 0; i < row; i++)

    delete[] adm[i];

    delete[] adm;

    }

};

int main(){

    //inserting a mtarix

    int n=0,x=0,y=0;

    cout<<"Enter Number of Vertices: ";

        cin>>n;

    Graph m(9,9);

    for (int i=0;i<n;i++){

        cout<<"Enter (x,y): ";

        cin>>x;

        cout<<"\t     ";

        cin>>y;

        m.addedge(x,y);

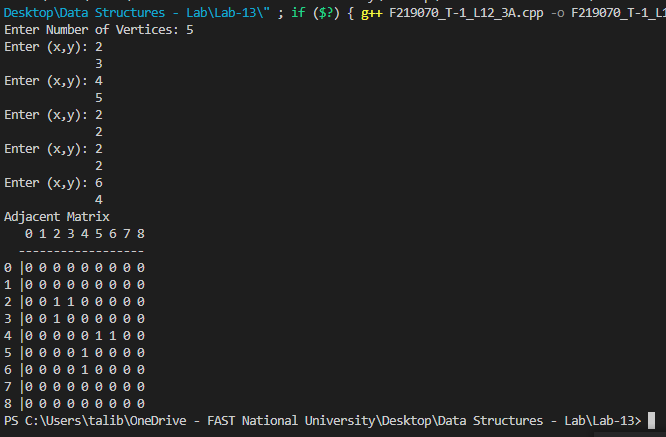
        m.addedge(y,x);

    }

    m.print();//Printing a adjnaacent matricx

}

**Output**

****

**Task-2**

#include<iostream>

using namespace std;

//making a class ADT

class Graph{

    int \*\*adm;

    int row,col;

public:

    //Constructor

    Graph(int r = 9,int c =9){

        row = r;

        col = c;

        //Allocating 2-d array

            adm = new int\*[row];

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

                \*(adm+i) = new int[col];

            }

            for (int i=0;i<row;i++){

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

                    adm[i][j] = 0;

                }

            }

    }

    //printing a matrix

    void print(){

    cout<<"Adjacent Matrix\n";

        cout<<"   ";

    for (int i=0;i<col;i++){

        cout<<i<<" ";

    }

        cout<<"\n  ";

    for (int i=0;i<col;i++)

        cout<<"--";

        cout<<"\n";

    //travesiong a matrix

    for (int i=0;i<row;i++){

        cout<<i<<" |";

        for (int j=0;j<col;j++){

            cout<<adm[i][j]<<" ";

        }

        cout<<"\n";

    }

    }

    void addedge(int i, int j){

        adm[i][j] = 1;

    }

    //count of edges

    int count\_edges(){

        cout<<"count of each node edges\n";

        int count = 0;

        for (int i=0;i<row;i++){

            for (int j=0;j<col;j++){

                if(adm[i][j] == 1){

                    count++;

                }

            }

            if(count !=0)

            cout<<i<<": "<<count<<endl;

            count = 0;

        }

    }

 ~Graph(){

        for (int i = 0; i < row; i++)

    delete[] adm[i];

    delete[] adm;

    }

};

int main(){

       //inserting a mtarix

    int n=0,x=0,y=0;

    cout<<"Enter Number of Vertices: ";

        cin>>n;

    Graph m(10,10);

    for (int i=0;i<n;i++){

        cout<<"Enter (x,y): ";

        cin>>x;

        cout<<"\t     ";

        cin>>y;

        m.addedge(x,y);

        m.addedge(y,x);

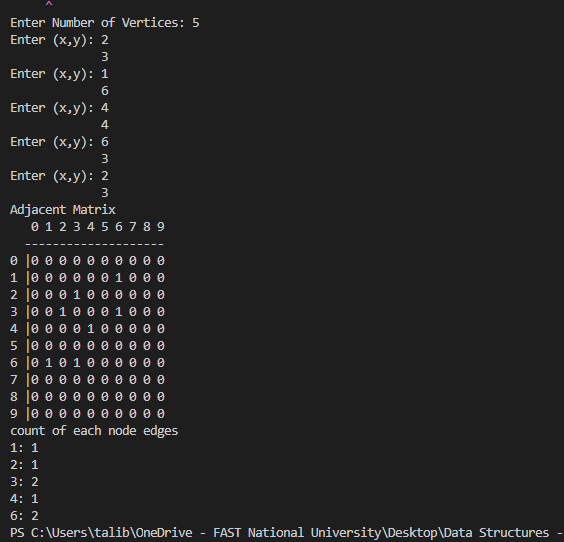
    }

    m.print();//Printing a adjnaacent matricx

    m.count\_edges();

}

**Output**

****

**Task-3**

#include <iostream>

#include <cstdlib>

using namespace std;

//list node

struct AdjListNode

{

    int data;

    struct AdjListNode\* next;

};

 //adj list

struct AdjList

{

    struct AdjListNode \*head;

};

//Graph class

class Graph

{

    private:

        int V;

        AdjList\* array;

    public:

        Graph(int V)

        {

            this->V = V;

            array = new AdjList [V];

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

                array[i].head = NULL;

        }

        void addEdge(int src, int dest){

            AdjListNode\* newNode = new AdjListNode;

            newNode->data = dest;

            newNode->next = NULL;

            newNode->next = array[src].head;

            array[src].head = newNode;

             newNode = new AdjListNode;

            newNode->data = src;

            newNode->next = NULL;

            newNode->next = array[dest].head;

            array[dest].head = newNode;

        }

        void printGraph(){

            cout<<"Adjeceny List\n";

            int v;

            for (v = 0; v < V; ++v){

                AdjListNode\* tmp = array[v].head;

                cout<<v<<":";

                while (tmp)

                {

                    cout<<"-> "<<tmp->data;

                    tmp = tmp->next;

                }

                cout<<endl;

            }

        }

};

int main(){

    Graph gh(5);

    gh.addEdge(0, 1);

    gh.addEdge(0, 4);

    gh.addEdge(1, 2);

    gh.addEdge(1, 3);

    gh.addEdge(1, 4);

    gh.addEdge(2, 3);

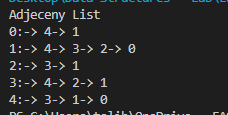
    gh.addEdge(3, 4);

    gh.printGraph();

    return 0;

}

**Output**

****