*//undirected graph*

*#include* <iostream>  
*#include* <fstream>  
  
*using namespace* std;  
*int* main() {  
  
 *int* adj\_mat[100][100]={0};  
 ifstream file\_obj;  
 file\_obj.open("D:\\Programming\\C++\\Graph\\graph.txt",ios::in);  
 *if*(file\_obj.is\_open()){  
 *int* n,e;  
 file\_obj>>n>>e;  
 *for*(*int* ind=0; ind<e; ind++){  
 *int* node1,node2;  
 file\_obj>>node1>>node2;  
  
 adj\_mat[node1][node2]=1;  
 adj\_mat[node2][node1]=1;  
 }  
 *//printing the matrix  
 for*(*int* row=1; row<=n; row++){  
 *for*(*int* col=1; col<=n; col++){  
 cout<<adj\_mat[row][col]<<" ";  
 }  
 cout<<endl;  
 }  
 } *else*{  
 cout<<"No file "<<endl;  
 }  
 *return* 0;  
}

directed graph:

*#include* <iostream>  
*#include* <fstream>  
  
*using namespace* std;  
*int* main() {  
  
 *double* adj\_mat[100][100]={0};  
 *//int adj\_mat[100][100]={0};* ifstream file\_obj;  
 file\_obj.open("D:\\Programming\\C++\\Graph\\graph.txt",ios::in);  
 *if*(file\_obj.is\_open()){  
 *int* n,e;  
 file\_obj>>n>>e;  
 *for*(*int* ind=0; ind<e; ind++){  
 *int* node1,node2;  
 *double* weight;  
 file\_obj>>node1>>node2>>weight;  
  
 adj\_mat[node1][node2]=weight;  
*// adj\_mat[node2][node1]=weight;* }  
 *//printing the matrix  
 for*(*int* row=1; row<=n; row++){  
 *for*(*int* col=1; col<=n; col++){  
 cout<<adj\_mat[row][col]<<" ";  
 }  
 cout<<endl;  
 }  
 } *else*{  
 cout<<"No file "<<endl;  
 }  
 *return* 0;  
}

**adjacency list**

#include<fstream>

#include<vector>

using namespace std;

int main(){

vector<int>adj\_list[100];

ifstream file\_obj;

file\_obj.open("D:\\Programming\\C++\\Graph\\graph.txt",ios::in);

if(file\_obj.is\_open()){

int n,e;

file\_obj>>n>>e;

for(int ind=0; ind<e; ind++){

int node1,node2;

file\_obj>>node1>>node2;

adj\_list[node1].push\_back(node2);

adj\_list[node2].push\_back(node1);

}

for(int ind=1; ind<=n; ind++){

cout<<ind<<" : ";

vector<int>neighbors=adj\_list[ind];

for(int i=0; i<neighbors.size(); i++){

cout<<neighbors[i]<<" ";

}

cout<<endl;

}

}

return 0;

}

**directed graph**

#include<iostream>

#include<fstream>

#include<vector>

using namespace std;

int main(){

vector<pair<int,double>>adj\_list[100];

ifstream file\_obj;

file\_obj.open("D:\\Programming\\C++\\Graph\\graph.txt",ios::in);

if(file\_obj.is\_open()){

int n,e;

file\_obj>>n>>e;

for(int ind=0; ind<e; ind++){

int node1,node2;

double wt;

file\_obj>>node1>>node2>>wt;

// adj\_list[node1].push\_back(node2);

// adj\_list[node2].push\_back(node1);

pair<int,double>p(node2,wt);

adj\_list[node1].push\_back(p);

}

for(int ind=1; ind<=n; ind++){

cout<<ind<<" : ";

vector<pair<int,double>>neighbors=adj\_list[ind];

for(int i=0; i<neighbors.size(); i++){

cout<<"("<<neighbors[i].first<<","<<neighbors[i].second<<")";

}cout<<endl;

}

}

return 0;

}