ASSIGNMENT 3

AIM:-

There are flight between 2 cities.If there is a flight between city A and city B there is a edge between the cities.The cost of edge can be the time the flight takes to reach city B .Represent this in a graph.The node represents the city.using adjacency list representation of the graph .Justify the storage representation used.

OBJECTIVE:-

To implement graph as adjacency matrix with cities as vertex and edges as distance between 2 cities.

THEORY:-

Graph is a data structure that consists of following two components:

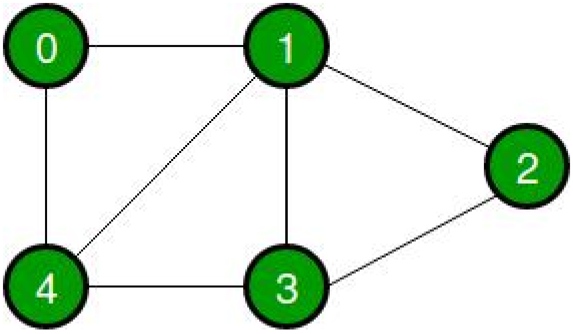
1. A finite set of vertices also called as nodes.

2. A finite set of ordered pair of the form (u, v) called as edge. The pair is ordered because (u, v) is not same as (v, u) in case of a directed graph(di-graph). The pair of the form (u, v) indicates that there is an edge from vertex u to vertex v. The edges may contain weight/value/cost.

Graphs are used to represent many real-life applications: Graphs are used to represent networks. The networks may include paths in a city or telephone network or circuit network. Graphs are also used in social networks like linkedIn, Facebook. For example, in Facebook, each person is represented with a vertex(or node). Each node is a structure and contains information like person id, name, gender and locale.

See this for more applications of graph.

Following is an example of an undirected graph with 5 vertices.



Following two are the most commonly used representations of a graph.

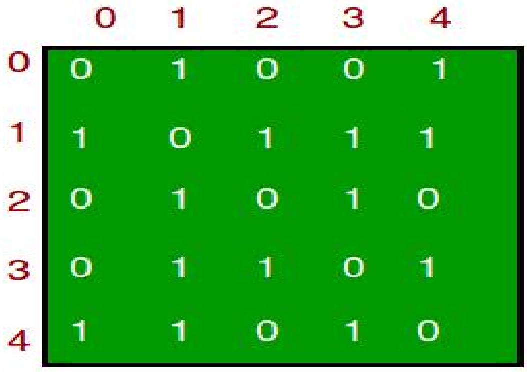
1. Adjacency Matrix

2. Adjacency List

There are other representations also like, Incidence Matrix and Incidence List. The choice of the graph representation is situation specific. It totally depends on the type of operations to be performed and ease of use.

Adjacency Matrix:
Adjacency Matrix is a 2D array of size V x V where V is the number of vertices in a graph. Let the 2D array be adj[][], a slot adj[i][j] = 1 indicates that there is an edge from vertex i to vertex j. Adjacency matrix for undirected graph is always symmetric. Adjacency Matrix is also used to represent weighted graphs. If adj[i][j] = w, then there is an edge from vertex i to vertex j with weight w.

The adjacency matrix for the above example graph is:



	0	1	2	3	4
0	0	1	0	0	1
1	1	0	1	1	1
2	0	1	0	1	0
3	0	1	1	0	1
4	1	1	0	1	0

Pros: Representation is easier to implement and follow. Removing an edge takes O(1) time. Queries like whether there is an edge from vertex 'u' to vertex 'v' are efficient and can be done O(1).

ALGORITHM:-

```
Void graph()
cout<<"Enter the no of vertices";
     cin>>n;
```

```
    for(i=0;i<n;i++) {
        g[i]=NULL;                      // initially array with NULL values
        cout<<“Enter the no of edges";
        cin>>no_of_edges
        for(i=0;i<no_of_edges;i++)    {
            cout<<“Enter an edge(u,v)";
            cin>>vi>>vj;

}
```

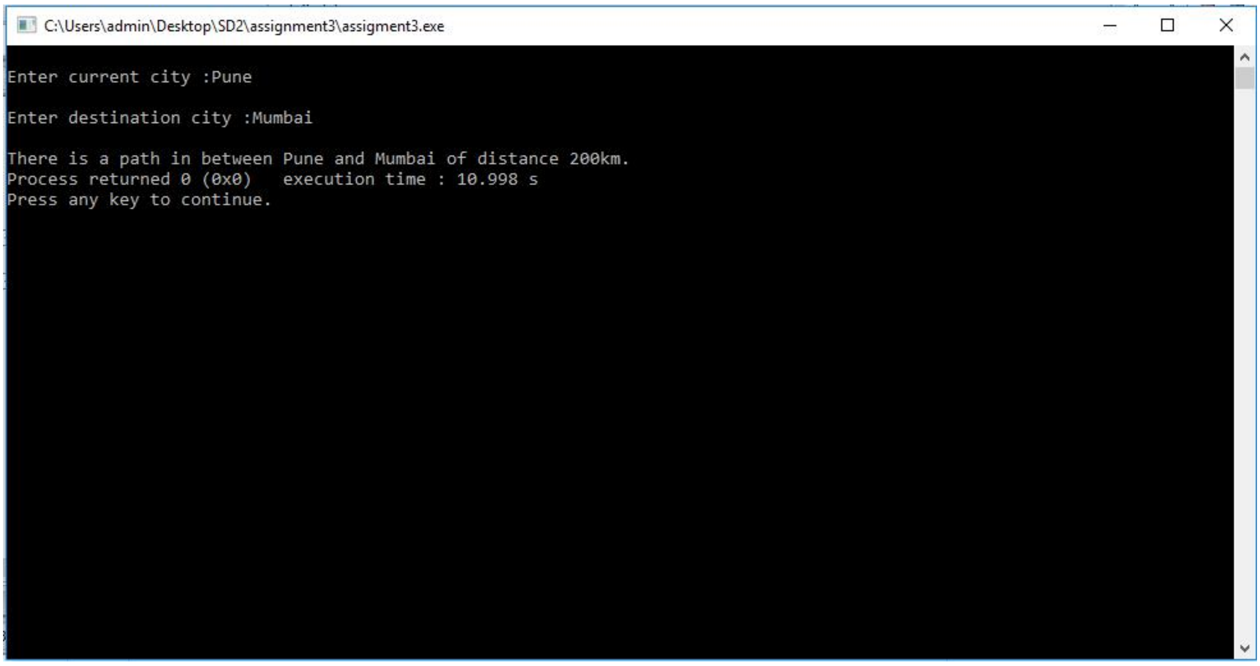
CODE:-

```
#include<iostream>
using namespace std;
#define SIZE 10
int main ()
{
    string cities [SIZE]={"Pune","Mumbai","Chennai","Delhi"};
    int dis[SIZE][SIZE]={
                            {0,200,0,1200},
                            {200,0,1400,1000},
                            {0,1400,0,0},
                            {1200,1000,0,0}
                        };
    string current,destination;
    cout<<"\nEnter current city :";
    cin>>current;
    cout<<"\nEnter destination city :";
    cin>>destination;
    if (current==destination)
        cout<<"Error! Current and Destination cities cannot be same.";
    else
    {
        int cr,cc;
        for (int i=0;i<4;i++)
        {
            if (cities[i]==current)
                cr=i;
            //else
                //cout<<"\nEnter valid city.";
            if (cities[i]==destination)
                cc=i;
            //else
                //cout<<"\nEnter valid city.";
        }


        if (dis[cr][cc]!=0)
            cout<<"\nThere is a path in between "<<current<<" and 
"<<destination<<"of distance "<<dis[cr][cc]<<"km.";
        else
            cout<<"\nThere is no path between these cities.";
    }

    return 0;
}
```

OUTPUT:-



```
Enter current city :Pune

Enter destination city :Mumbai

There is a path in between Pune and Mumbai of distance 200km.
Process returned 0 (0x0)   execution time : 10.998 s
Press any key to continue.
```

CONCLUSION:-

We have successfully implemented the concept of graph by representing it in adjacency matrix.