Practical 6

18BCE243

Code of Practical 6

```
#ifndef GTLAB_PRACTICAL6
#define GTLAB_PRACTICAL6

#include <iostream>
#include <climits>
#include <vector>
using namespace std;

#define INF 99999

static void _print_matrix(vector<vector<int> > dist, const int V)
{
    cout << "The following matrix shows the shortest distances"
            " between every pair of vertices.\n";

    for (int i = 0; i < V; i++)
    {
        for (int j = 0; j < V; j++)
        {
            if (dist[i][j] == INF)
                cout << "INF" << "     ";
            else
                cout << dist[i][j] << "     ";
        }
        cout << endl;
    }
}


void floyd_warshall(vector<vector<int> > graph, const int V)
{
    vector<vector<int> > dist(V);
    for(int i=0 ; i<V ; i++) dist[i].resize(V);

    int i, j, k;

    for (i = 0; i < V; i++)
        for (j = 0; j < V; j++)
            dist[i][j] = graph[i][j];
```

```
    for (k = 0; k < V; k++)
    {
        // Pick all vertices as source one by one
        for (i = 0; i < V; i++)
        {
            // Pick all vertices as destination for the
            // above picked source
            for (j = 0; j < V; j++)
            {
                // If vertex k is on the shortest path from
                // i to j, then update the value of dist[i][j]
                if (dist[i][k] + dist[k][j] < dist[i][j])
                    dist[i][j] = dist[i][k] + dist[k][j];
            }
        }
    }

    // Print the shortest distance matrix
    _print_matrix(dist, V);
}

#endif // GT_PRACTICAL6
```

Test Driver (with Inputs)

```
#include "practical6.h"

int main()
{
    /*

            10
    (0)------->(3)
        |     /|\
    5 |     |
        |     | 1
    \|/     |
    (1)------->(2)
            3

    */
    vector<vector<int> > graph =
                      {
                        {0   , 5   , INF, 10},
                        {INF, 0   , 3   , INF},
                        {INF, INF, 0   , 1},
```

```
                    {INF, INF, INF, 0}
                };

    // Print the solution
    floyd_warshall(graph, 4);
    return 0;
}
```

Output

```
The following matrix shows the shortest distances
between every pair of vertices.

0     5     8     9
INF     0     3     4
INF     INF     0     1
INF     INF     INF     0
```