```
using System;
using System.Collections.Generic;
using System.Linq;

namespace Graphs
{
        public class Road
        {
            public int Distance { get; set; }
            public int House { get; set; }
        }
        public class House
        {
            public List<Road> Houses { get; set; }

            public House()
            {
                Houses = new List<Road>();
            }
        }
        public class Dijkstra
        {
            public bool Visited { get; set; }
            public int Distance { get; set; }

            public Dijkstra()
            {
                Visited = false;
                Distance = int.MaxValue;
            }
        }
        public class Graph
        {
            private Dictionary<int, House> houses;

            public Graph()
            {
                houses = new Dictionary<int, House>();
            }

            public void AddHouse(int name)
            {
                houses.Add(name, new House());
            }

            public void AddRoad(int start, int end, int distance)
            {
                houses[start].Houses.Add(new Road { House = end, Distance =
distance });

                    houses[end].Houses.Add(new Road { House = start, Distance =
distance });
            }

            public int FindShortestDistance(int start, int end)
            {
                Dictionary<int, Dijkstra> info = new Dictionary<int,
Dijkstra>(houses.Count);
                foreach (int current in houses.Keys)
                {
                    info.Add(current, new Dijkstra());
                }
                info[start].Distance = 0;
                while (!info.Select(x => x.Value.Visited).Aggregate((x, y) => x
& y))
                {
                    int current = info.Where(x => !x.Value.Visited &&
x.Value.Distance == info.Where(y => !y.Value.Visited).Min(y =>
y.Value.Distance)).First().Key;
                    List<Road> neighbors = houses[current].Houses.Where(x => !
info[x.House].Visited).ToList();
                    foreach (Road house in neighbors)
                    {
                        int distance = info[current].Distance + house.Distance;
                        if (info[house.House].Distance > distance)
                            info[house.House].Distance = distance;
                    }
                    info[current].Visited = true;
                }
                return info[end].Distance;
            }
            public void TheMinimumTotaDistance(Graph graph)
            {
                List<int> distances = new List<int>();

                for (int i = 1; i <= houses.Count; i++)
                {
                    int totalDistance = 0;
                    for (int j = 1; j <= houses.Count; j++)
                    {
                        totalDistance += graph.FindShortestDistance(i, j);
                        Console.WriteLine("���������� �� ����  {0} �� ���� {1}
����� : {2} ",i,j, graph.FindShortestDistance(i, j));
                    }
                    Console.WriteLine("����� ��������� �� {0}  ����  �� ����
����� �����: {1} ", i, totalDistance);
                    Console.WriteLine();
                    distances.Add(totalDistance);
                }
                int houseNumberWithTheMinimumTotaDistance =
distances.IndexOf(distances.Min()) + 1;
                Console.WriteLine("����� ���� �� �������� ��������� ��������� ��
���� ��������� ������� ����� ����������� :  {0}",
houseNumberWithTheMinimumTotaDistance);
            }
        }
    class Program
    {
        static void Main(string[] args)
        {
            int[] idHouse = { 1, 2, 3, 4, 5, 6 };

            Graph graph = new Graph();

            for (int i = 0; i < idHouse.Length; i++)
                graph.AddHouse(idHouse[i]);

            graph.AddRoad(1, 2, 7);
            graph.AddRoad(1, 3, 9);
            graph.AddRoad(1, 5, 14);
            graph.AddRoad(2, 3, 18);
            graph.AddRoad(2, 4, 15);
            graph.AddRoad(3, 4, 11);
            graph.AddRoad(3, 5, 2);
            graph.AddRoad(4, 6, 6);
            graph.AddRoad(6, 5, 20);

            graph.TheMinimumTotaDistance(graph);
            Console.ReadLine();
        }
    }
}
```