

```

// SL.No.: - 31
// Admission No.: - 21JE0269
// Name: - Chotaliya Zeel Vijaybhai

#include <bits/stdc++.h>
using namespace std;

const int INF = 100; // Using 100 as infinity as per the assignment
const int NODES = 7;

class DijkstraAlgorithm {
private:
    vector<vector<int>> cost_matrix;
    vector<vector<int>> distance_matrix;

public:
    DijkstraAlgorithm() : cost_matrix(NODES, vector<int>(NODES)),
        distance_matrix(NODES, vector<int>(NODES)) {}

    void input() {
        for (int i = 0; i < NODES; ++i) {
            for (int j = i; j < NODES; ++j) {
                cout << "Enter cost (max:100) of the node " << j << " from " << i << endl;
                cin >> cost_matrix[i][j];
                cost_matrix[j][i] = cost_matrix[i][j]; // Symmetric matrix
            }
        }
    }

    void dijkstra(int root) {
        vector<bool> visited(NODES, false); // 's' matrix as per assignment.
        vector<int>& distances = distance_matrix[root]; // 'd' matrix as per assignment.
        distances = cost_matrix[root];

        visited[root] = true;
        distances[root] = 0;

        for (int i = 0; i < NODES - 1; ++i) {
            int u = -1;
            int min_dist = INF;

            for (int j = 0; j < NODES; ++j) {
                if (!visited[j] && distances[j] < min_dist) {
                    u = j;
                    min_dist = distances[j];
                }
            }

            if (u == -1) break;

            visited[u] = true;

            for (int v = 0; v < NODES; ++v) {
                if (!visited[v] && cost_matrix[u][v] != INF) {
                    distances[v] = min(distances[v], distances[u] + cost_matrix[u][v]);
                }
            }
        }
    }

    void costupdate() {
        for (int root = 0; root < NODES; ++root) {
            dijkstra(root);
        }
    }
}

```

```

void display() {
    cout << "The input cost matrix:" << endl;
    for (const auto& row : cost_matrix) {
        for (int cost : row) {
            cout << setw(3) << cost << " "; // 'setw()' function is used to specified white space.
        }
        cout << endl;
    }

    cout << "\nThe updated cost matrix:" << endl;
    for (const auto& row : distance_matrix) {
        for (int dist : row) {
            cout << setw(3) << dist << " "; // 'setw()' function is used to specified white space.
        }
        cout << endl;
    }
}

};

int main() {
    DijkstraAlgorithm dijkstra;
    dijkstra.input();
    dijkstra.costupdate();
    dijkstra.display();
    return 0;
}

```