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

void dijkstra(vector<vector<int>> &graph, int source) {
    int n = graph.size();
    vector<int> dist(n, INT_MAX);
    vector<bool> visited(n, false);
    dist[source] = 0;

    for (int i = 0; i < n - 1; i++) {
        int minDist = INT_MAX, u = -1;
        for (int v = 0; v < n; v++) {
            if (!visited[v] && dist[v] < minDist) {
                minDist = dist[v];
                u = v;
            }
        }

        visited[u] = true;

        for (int v = 0; v < n; v++) {
            if (!visited[v] && graph[u][v] && dist[u] != INT_MAX &&
dist[u] + graph[u][v] < dist[v]) {
                dist[v] = dist[u] + graph[u][v];
            }
        }
    }

    cout << "Shortest distances from source node " << source << ":\n";
    for (int i = 0; i < n; i++) {
        cout << "Node " << i << ": " << dist[i] << endl;
    }
}

int main() {
    vector<vector<int>> graph = {
        {0, 4, 0, 0, 0, 0, 0, 8, 0},
        {4, 0, 8, 0, 0, 0, 0, 11, 0},
        {0, 8, 0, 7, 0, 4, 0, 0, 2},
        {0, 0, 7, 0, 9, 14, 0, 0, 0},
        {0, 0, 0, 9, 0, 10, 0, 0, 0},
        {0, 0, 4, 14, 10, 0, 2, 0, 0},
        {0, 0, 0, 0, 0, 2, 0, 1, 6},
        {8, 11, 0, 0, 0, 0, 1, 0, 7},
        {0, 0, 2, 0, 0, 0, 6, 7, 0}
    };

    int source = 0;
    dijkstra(graph, source);

    return 0;
}
```