```cpp
#include <iostream>

#define N 5

unsigned int graph[N][N] =
{
    {0, 0, 1, 3, 0},
    {0, 0, 0, 0, 5},
    {0, 2, 0, 1, 0},
    {0, 0, 0, 0, 5},
    {0, 0, 0, 5, 0}
};

// returns index of minimum element in array
size_t find_min(unsigned int* arr, bool* visited)
{
    size_t idx = 0;
    for (size_t i = 1; i < N+1; ++i)
    {
        if (!visited[i-1] && (idx == 0 || arr[idx-1] > arr[i-1]))
        {
            idx = i;
        }
    }
    return idx-1;
}

unsigned int min(unsigned int a, unsigned int b)
{
    return a < b ? a : b;
}

unsigned int* dijkstra(size_t start)
{
    static unsigned int shortest[N];
    bool visited[N];

    size_t current;

    // Init dijkstra
    for (size_t i = 0; i < N; ++i)
    {
        visited[i] = false;
        shortest[i] = UINT32_MAX;
    }

    // Start dijkstra
    shortest[start] = 0;
    for (size_t i = 0; i < N; ++i)
    {
        current = find_min(shortest, visited);
        visited[current] = true;
        if (shortest[current] == UINT32_MAX) continue;
        for (size_t j = 0; j < N; ++j)
        {
            if (graph[current][j] != 0)
            {
                shortest[j] = min(shortest[j], shortest[current] + graph[current][j]);
            }
        }
    }

    return shortest;
}

void print_graph(void)
{
    for (size_t i = 0; i < N; ++i)
    {
        for (size_t j = 0; j < N; ++j)
        {
            std::cout << graph[i][j] << ' ';
        }
```

```
            std::cout << std::endl;
        }
    }

    int main()
    {
        unsigned int* ans;
        size_t start_point;
        std::cout << "Graph:\n";
        print_graph();
        std::cout << "Set start point: ";
        std::cin >> start_point;
        ans = dijkstra(start_point);
        for (size_t i = 0; i < N; ++i)
        {
            std::cout << ans[i] << ' ';
        }
        std::cout << std::endl;
    }

```