

# OSCN LAB 1

MOHAMMED NIZAMUDDIN  
M.TECH CSE  
2503B05144

**Write a C++ program to implement Dijkstra's Single Source Shortest Path Algorithm for a graph represented using an adjacency matrix.**

Number of vertices: 5

Edges:

0 1 4

0 2 8

1 4 6

2 3 2

3 4 10

Source vertex: 0

CODE:

```
#include <iostream>
#include <vector>
#include <queue>
#include <climits>

using namespace std;

vector<vector<vector<int>>> constructAdj(vector<vector<int>> &edges, int V) {
```

```
vector<vector<vector<int>>> adj(V);

for (const auto &edge : edges) {
    int u = edge[0];
    int v = edge[1];
    int wt = edge[2];

    adj[u].push_back({v, wt});
    adj[v].push_back({u, wt});
}

return adj;
}
```

```
vector<int> dijkstra(int V, vector<vector<int>> &edges, int src) {
    vector<vector<vector<int>>> adj = constructAdj(edges, V);

    priority_queue<vector<int>, vector<vector<int>>, greater<vector<int>>> pq;
    vector<int> dist(V, INT_MAX);

    dist[src] = 0;
    pq.push({0, src});

    while (!pq.empty()) {
        int u = pq.top()[1];
        pq.pop();

        for (const auto &v : adj[u]) {
            if (dist[v[0]] > dist[u] + v[1]) {
                dist[v[0]] = dist[u] + v[1];
                pq.push({dist[v[0]], v[0]});
            }
        }
    }

    return dist;
}
```

```
for (auto x : adj[u]) {  
    int v = x[0];  
    int weight = x[1];  
  
    if (dist[v] > dist[u] + weight) {  
        dist[v] = dist[u] + weight;  
        pq.push({dist[v], v});  
    }  
}  
  
return dist;  
}  
  
  
int main() {  
    int V = 5;  
    int src = 0;  
  
    vector<vector<int>> edges = {  
        {0, 1, 4},  
        {0, 2, 8},  
        {1, 4, 6},  
        {2, 3, 2},  
        {3, 4, 10}  
    };
```

```

vector<int> result = dijkstra(V, edges, src);

cout << "Shortest distances from source vertex " << src << ":\n";
for (int dist : result)
    cout << dist << " ";
cout << endl;
return 0;
}

```

The screenshot shows a code editor interface with the following details:

- Run Code**: Untitled
- Save**: C++
- Output**: Finished
- Clear Console**

The code in the editor is identical to the one above, including the `#include` statements and the two function definitions.

The output window displays the results of the execution:

```

Finished in 0 ms
Shortest distances from source vertex 0:
0 4 8 10 10

```