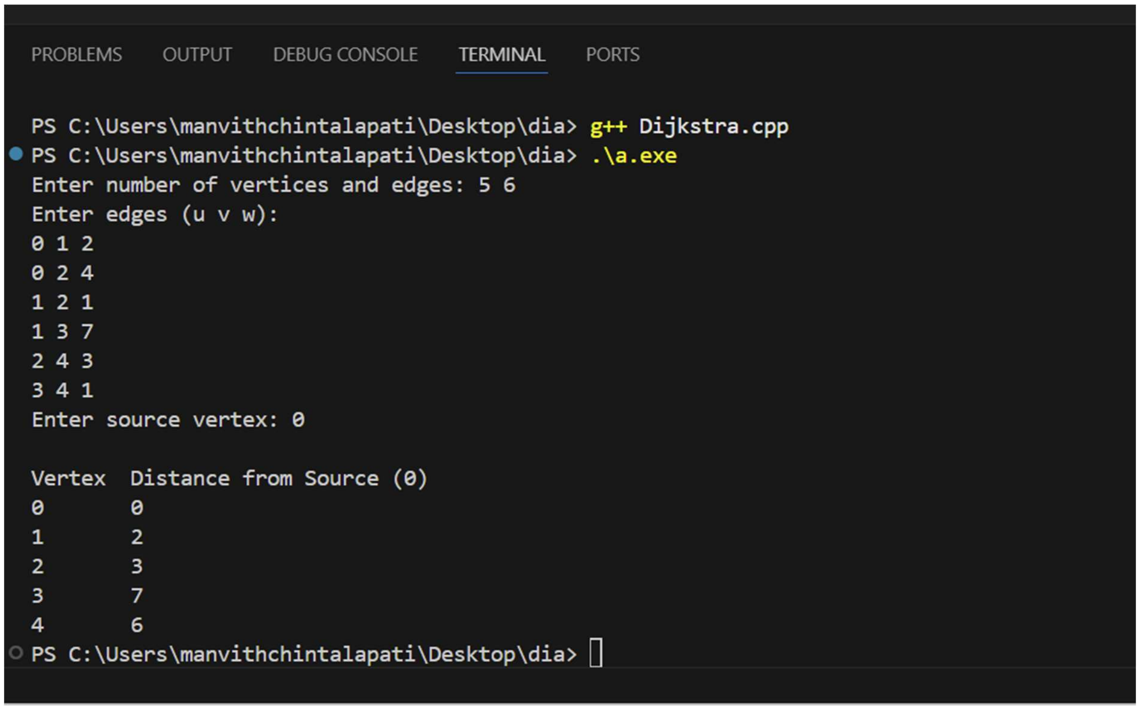
Lab 10

```
#include <iostream>

#include <vector>

#include <queue>

#include <utility>

using namespace std;


#define INF 1e9


void dijkstra(int V, vector<vector<pair<int, int>>> &adj, int src) {

  vector<int> dist(V, INF);


  dist[src] = 0;




  priority_queue<pair<int, int>, vector<pair<int, int>>, greater<pair<int, int>>> pq;


  pq.push({0, src});


  while (!pq.empty()) {

    int u = pq.top().second;

    int d = pq.top().first;

    pq.pop();



    if (d > dist[u])
```

```
            continue;



        for (auto &edge : adj[u]) {
            int v = edge.first;
            int weight = edge.second;



            if (dist[u] + weight < dist[v]) {
                dist[v] = dist[u] + weight;
                pq.push({dist[v], v});
            }
        }
    }



    cout << "\nVertex\tDistance from Source (" << src << ")\n";
    for (int i = 0; i < V; i++) {
        if (dist[i] == INF)
            cout << i << "\tINF\n";
        else
            cout << i << "\t" << dist[i] << "\n";
    }
}

int main() {
    int V, E;
    cout << "Enter number of vertices and edges: ";
    cin >> V >> E;


    vector<vector<pair<int, int>>> adj(V);


    cout << "Enter edges (u v w):\n";
    for (int i = 0; i < E; i++) {
        int u, v, w;
        cin >> u >> v >> w;
        adj[u].push_back({v, w});
        adj[v].push_back({u, w});


    }


    int src;
    cout << "Enter source vertex: ";
    cin >> src;


    dijkstra(V, adj, src);


    return 0;
}
```



```
PS C:\Users\manvithchintalapati\Desktop\dia> g++ Dijkstra.cpp
PS C:\Users\manvithchintalapati\Desktop\dia> .\a.exe
Enter number of vertices and edges: 5 6
Enter edges (u v w):
0 1 2
0 2 4
1 2 1
1 3 7
2 4 3
3 4 1
Enter source vertex: 0

Vertex  Distance from Source (0)
0       0
1       2
2       3
3       7
4       6
PS C:\Users\manvithchintalapati\Desktop\dia> 
```