

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
Dijkstra-Algo.cpp x
Practical-5 > C++ Dijkstra-Algo.cpp > main()
1 // Author: Anish Tilloo
2 // Roll No. : 34
3 // Program: Dijkstra Algorithm
4
5 #include <bits/stdc++.h>
6 using namespace std;
7
8 // Dijkstra Algorithm
9 class Graph{
10 public:
11     int V;
12     list<pair<int, int> > *l;
13     Graph(int v){
14         V = v;
15         l = new list<pair<int, int> > [V];
16     }
17
18     void addEdge(int ver1, int ver2, int wt, bool undirected = true){
19         l[ver1].push_back({wt, ver2});
20         if (undirected)
21         {
22             l[ver2].push_back({wt, ver1});
23         }
24     }
25
26     int dijkstra(int source, int destination){
27         // dist array
28         vector<int> dist(V, INT_MAX);
29         // set of pair to store wt and node
30         set<pair<int, int> > s;
31
32         // initialize the distance
33         dist[source] = 0;
34         s.insert({0, source});
35
36         // check if the set is empty or not
37         while (!s.empty())
38         {
39             // get the element with minimum distance
40             auto it = *s.begin();
41             int u = it.second;
42             s.erase(it);
43
44             // traverse all adjacent vertices
45             for (auto &v : l[u])
46             {
47                 int wt = v.first;
48                 int v2 = v.second;
49                 if (dist[u] + wt < dist[v2])
50                 {
51                     dist[v2] = dist[u] + wt;
52                     s.insert({dist[v2], v2});
53                 }
54             }
55         }
56         return dist[destination];
57     }
58 };
59
60 int main()
61 {
62     int V, E;
63     cin >> V >> E;
64     Graph g(V);
65     for (int i = 0; i < E; i++)
66     {
67         int u, v, wt;
68         cin >> u >> v >> wt;
69         g.addEdge(u, v, wt);
70     }
71     int source, destination;
72     cin >> source >> destination;
73     int ans = g.dijkstra(source, destination);
74     cout << ans << endl;
75     return 0;
76 }
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
Dijkstra-Algo.cpp x
Practical-5 > C++ Dijkstra-Algo.cpp > main()
18 void addEdge(int ver1, int ver2, int wt, bool undirected = true){
19     l[ver1].push_back({wt, ver2});
20     if (undirected)
21     {
22         l[ver2].push_back({wt, ver1});
23     }
24 }
25
26 int dijkstra(int source, int destination){
27     // dist array
28     vector<int> dist(V, INT_MAX);
29     // set of pair to store wt and node
30     set<pair<int, int> > s;
31
32     // initialize the distance
33     dist[source] = 0;
34     s.insert({0, source});
35
36     // check if the set is empty or not
37     while (!s.empty())
38     {
39         // get the element with minimum distance
40         auto it = *s.begin();
41         int u = it.second;
42         s.erase(it);
43
44         // traverse all adjacent vertices
45         for (auto &v : l[u])
46         {
47             int wt = v.first;
48             int v2 = v.second;
49             if (dist[u] + wt < dist[v2])
50             {
51                 dist[v2] = dist[u] + wt;
52                 s.insert({dist[v2], v2});
53             }
54         }
55     }
56     return dist[destination];
57 }
58
59 int main()
60 {
61     int V, E;
62     cin >> V >> E;
63     Graph g(V);
64     for (int i = 0; i < E; i++)
65     {
66         int u, v, wt;
67         cin >> u >> v >> wt;
68         g.addEdge(u, v, wt);
69     }
70     int source, destination;
71     cin >> source >> destination;
72     int ans = g.dijkstra(source, destination);
73     cout << ans << endl;
74     return 0;
75 }
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
Dijkstra-Algo.cpp x
Practical-5 > C++ Dijkstra-Algo.cpp > main()
26 int dijkstra(int source, int destination){
27
28     // dist array
29     vector<int> dist(V, INT_MAX);
30     // set of pair to store wt and node
31     set<pair<int, int> > s;
32
33     // initialize the distance
34     dist[source] = 0;
35     s.insert({0, source});
36
37
38     // check if the set is empty or not
39     while (!s.empty())
40     {
41         // see what is at the first position in the set
42         auto atFirst = s.begin();
43         // actual node
44         auto node = atFirst->second;
45         // dist of the node covered till now
46         auto distTillNow = atFirst->first;
47
48     }
49
50     // pop the first value
51     s.erase(atFirst);
52
53     for(auto nbrPair : l[node]){
54         // neighbour node
55         auto nbr = nbrPair.second;
56         // weight of the current edge you are in or want to pass
57         auto currentEdge = nbrPair.first;
58
59         if (distTillNow + currentEdge < dist[nbr])
60         {
61             auto ifExist = s.find({dist[nbr], nbr});
62             if (ifExist != s.end())
63                 s.erase(ifExist);
64             s.insert({distTillNow + currentEdge, nbr});
65         }
66     }
67
68     return dist[destination];
69 }
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
Dijkstra-Algo.cpp x
Practical-5 > C++ Dijkstra-Algo.cpp > main()
38 // check if the set is empty or not
39 while (!s.empty())
40 {
41     // see what is at the first position in the set
42     auto atFirst = s.begin();
43     // actual node
44     auto node = atFirst->second;
45     // dist of the node covered till now
46     auto distTillNow = atFirst->first;
47
48     // pop the first value
49     s.erase(atFirst);
50
51     for(auto nbrPair : l[node]){
52         // neighbour node
53         auto nbr = nbrPair.second;
54         // weight of the current edge you are in or want to pass
55         auto currentEdge = nbrPair.first;
56
57         if (distTillNow + currentEdge < dist[nbr])
58         {
59             auto ifExist = s.find({dist[nbr], nbr});
60             if (ifExist != s.end())
61                 s.erase(ifExist);
62             s.insert({distTillNow + currentEdge, nbr});
63         }
64     }
65
66     return dist[destination];
67 }
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
Dijkstra-Algo.cpp x
Practical-5 > C++ Dijkstra-Algo.cpp > main()
57         if (distTillNow + currentEdge < dist[nbr])
58         {
59             auto ifExist = s.find({dist[nbr], nbr});
60             if (ifExist != s.end())
61             {
62                 s.erase(ifExist);
63             }
64
65             dist[nbr] = distTillNow + currentEdge;
66             s.insert({dist[nbr], nbr});
67         }
68     }
69
70 }
71
72 for (int i = 0; i < V; i++)
73 {
74     cout << "Distance from 0 to " << i << " is " << dist[i] << endl;
75 }
76 return dist[destination];
77 }
78 };
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
Dijkstra-Algo.cpp x
Practical-5 > C++ Dijkstra-Algo.cpp > main()
79
80 int main(){
81     int n;
82     cout << "Enter the number of vertices" << endl;
83     cin >> n;
84     Graph g(n);
85     int m;
86     cout << "Enter the no of edges" << endl;
87     cin >> m;
88     cout << endl;
89
90     cout << "Edges " << endl;
91     for (int i = 0; i < m; i++)
92     {
93         int vex1, vex2, weight;
94         cout << "Vertex One: ";
95         cin >> vex1;
96         cout << "vertex Two: ";
97         cin >> vex2;
98         cout << "Weight: ";
99         cin >> weight;
100        cout << endl;
101    }
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
Dijkstra-Algo.cpp x
Practical-5 > C++ Dijkstra-Algo.cpp > main()
97     cin >> vex2;
98     cout << "Weight: ";
99     cin >> weight;
100    cout << endl;
101
102    g.addEdge(vex1, vex2, weight);
103
104    cout << endl;
105
106    int src, dest;
107    cout << "Enter the source node: " << endl;
108    cin >> src;
109    cout << "Enter the distnation node: " << endl;
110    cin >> dest;
111
112    g.dijkstra(src, dest);
113
114    return 0;
115 }
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL
PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-5> cd "e:\College-Work\Sixth-Semester\DAA-practical\Practical-5\" ; if ($?) {
g++ Dijkstra-Algo.cpp -o Dijkstra-Algo } ; if ($?) { .\Dijkstra-Algo }
Enter the number of vertices
5
Enter the no of edges
6
Edges
Vertex One: 0
vertex Two: 1
Weight: 1

Vertex One: 1
vertex Two: 2
Weight: 1

Vertex One: 0
vertex Two: 2
Weight: 4

Vertex One: 0
vertex Two: 3
Weight: 7

Vertex One: 3
vertex Two: 2
Weight: 2

Vertex One: 3
```

```
File Edit Selection View Go Run Terminal Help Dijkstra-Algo.cpp - DAA-practical - Visual Studio Code
PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL
Weight: 1
Vertex One: 0
vertex Two: 2
Weight: 4
Vertex One: 0
vertex Two: 3
Weight: 7
Vertex One: 3
vertex Two: 2
Weight: 2
Vertex One: 3
vertex Two: 4
Weight: 3
...
Enter the source node:
0
Enter the distnation node:
4
Distance from 0 to 0 is 0
Distance from 0 to 1 is 1
Distance from 0 to 2 is 2
Distance from 0 to 3 is 4
Distance from 0 to 4 is 7
PS E:\College-Work\Sixth-Semester\DAA-practical\Practical-5> |
```