

Experiment 2.2

Student Name: Gaurav Kumar

UID: 22MCC20177

Branch: Computer

Section/Group:- 1/B

Semester: One

Date of Performance: 27/11/2022

Subject Name:- Design & Analysis of Algorithms Lab

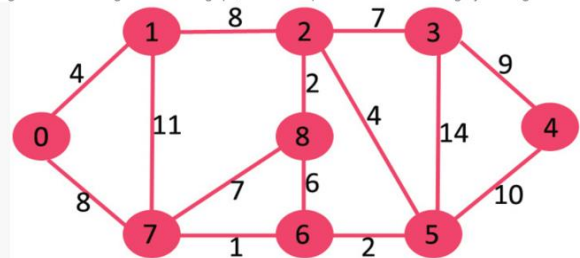
Subject Code: 22CAP-646

Task to be done:

Steps for experiment/practical:

copy and paste your code here/screenshots

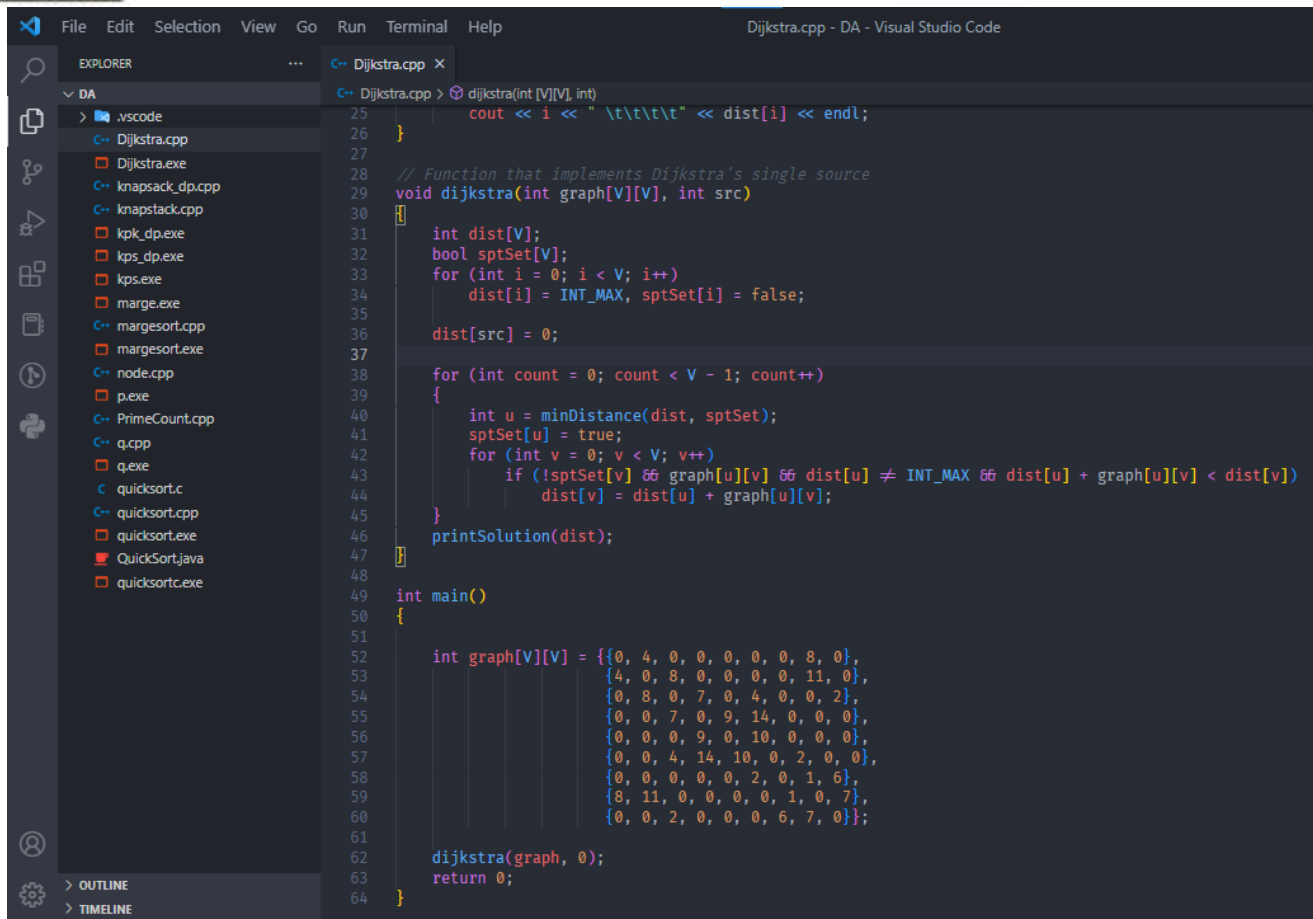
From a given vertex in a weighted connected graph, find shortest paths to other vertices using Dijkstra's algorithm.



```
File Edit Selection View Go Run Terminal Help Dijkstra.cpp - DA - Visual Studio Code

EXPLORER
DA
  .vscode
  Dijkstra.cpp
  Dijkstra.exe
  knapsack_dp.cpp
  knapsack.cpp
  kpk_dp.exe
  kps_dp.exe
  kps.exe
  marge.exe
  margesort.cpp
  margesort.exe
  node.cpp
  p.exe
  PrimeCount.cpp
  q.cpp
  q.exe
  quicksort.c
  quicksort.cpp
  quicksort.exe
  QuickSort.java
  quicksortc.exe

Dijkstra.cpp
1  #include <iostream>
2  #include <limits.h>
3  using namespace std;
4
5  // Number of vertices in the graph
6  #define V 9
7
8  // in shortest path tree
9  int minDistance(int dist[], bool sptSet[])
10 {
11     // Initialize min value
12     int min = INT_MAX, min_index;
13
14     for (int v = 0; v < V; v++)
15         if (sptSet[v] == false && dist[v] <= min)
16             min = dist[v], min_index = v;
17
18     return min_index;
19 }
20
21 void printSolution(int dist[])
22 {
23     cout << "Vertex \t Distance from Source" << endl;
24     for (int i = 0; i < V; i++)
25         cout << i << " \t\t\t\t" << dist[i] << endl;
26 }
27
```



```

File Edit Selection View Go Run Terminal Help
Dijkstra.cpp - DA - Visual Studio Code

EXPLORER
DA
  .vscode
  Dijkstra.cpp
  Dijkstra.exe
  knapsack_dp.cpp
  knapsack.cpp
  kpk_dp.exe
  kps_dp.exe
  kps.exe
  marge.exe
  margesort.cpp
  margesort.exe
  node.cpp
  p.exe
  PrimeCount.cpp
  q.cpp
  q.exe
  quicksort.c
  quicksort.cpp
  quicksort.exe
  QuickSort.java
  quicksort.exe

Dijkstra.cpp
25     cout << i << " \t\t\t" << dist[i] << endl;
26 }
27
28 // Function that implements Dijkstra's single source
29 void dijkstra(int graph[V][V], int src)
30 {
31     int dist[V];
32     bool sptSet[V];
33     for (int i = 0; i < V; i++)
34         dist[i] = INT_MAX, sptSet[i] = false;
35
36     dist[src] = 0;
37
38     for (int count = 0; count < V - 1; count++)
39     {
40         int u = minDistance(dist, sptSet);
41         sptSet[u] = true;
42         for (int v = 0; v < V; v++)
43             if (!sptSet[v] && graph[u][v] && dist[u] != INT_MAX && dist[u] + graph[u][v] < dist[v])
44                 dist[v] = dist[u] + graph[u][v];
45     }
46     printSolution(dist);
47 }
48
49 int main()
50 {
51
52     int graph[V][V] = {{0, 4, 0, 0, 0, 0, 0, 8, 0},
53                       {4, 0, 8, 0, 0, 0, 0, 11, 0},
54                       {0, 8, 0, 7, 0, 4, 0, 0, 2},
55                       {0, 0, 7, 0, 9, 14, 0, 0, 0},
56                       {0, 0, 0, 9, 0, 10, 0, 0, 0},
57                       {0, 0, 4, 14, 10, 0, 2, 0, 0},
58                       {0, 0, 0, 0, 0, 2, 0, 1, 6},
59                       {8, 11, 0, 0, 0, 0, 1, 0, 7},
60                       {0, 0, 2, 0, 0, 0, 6, 7, 0}};
61
62     dijkstra(graph, 0);
63     return 0;
64 }
  
```

Output (screenshots)

```

PS D:\Saurav\Sem 1\Practical\DA> g++ .\Dijkstra.cpp -o .\Dijkstra && .\Dijkstra
Vertex    Distance from Source
0
1          4
2         12
3         19
4         21
5         11
6          9
7          8
8         14
PS D:\Saurav\Sem 1\Practical\DA>
  
```

Learning outcomes (What I have learnt): Times new roman 12 size

Evaluation Grid:

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.	Demonstration and Performance (Quiz)		22
2.	Worksheet		8