Bangalore Institute of Technology

**Department of Computer Science and Engineering DESIGN AND ANALYSIS OF ALGORITHMS LAB (BCSL404)**

**Program 4**

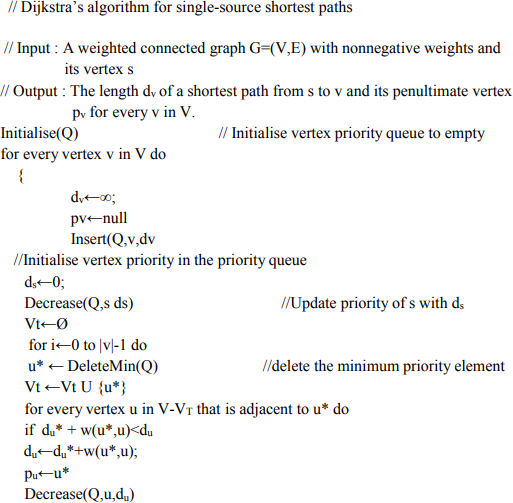
Design and implement C Program to find shortest paths from a given vertex in a weighted connected graph to other vertices using Dijkstra's algorithm.

Dijikstra’s algorithm : For a given source vertex(node) in the graph, the algorithm finds the path with lowest cost between that vertex and every other vertex. It can also be used for finding cost of shortest paths from a single vertex to a single destination vertex by stopping the algorithm once the shortest path to the destination vertex has been determined.

Efficiency:

1. Ѳ(|V2|) graph represented by weighted matrix and priority queue as unordered array
2. O(E log2 V ) graph represented by adjacency lists and priority queue as min- heap

Dijkstra(G,s)



**Program: #include<stdio.h> #define INF 999**

**void dijkstra(int c[10][10],int n,int s,int d[10])**

**{**

**int v[10],min,u,i,j; for(i=1;i<=n;i++)**

**{**

**d[i]=c[s][i]; v[i]=0;**

**}**

**v[s]=1;**

**for(i=1;i<=n;i++)**

**{**

**min=INF; for(j=1;j<=n;j++) if(v[j]==0 && d[j]<min)**

**{**

**min=d[j]; u=j;**

**}**

**v[u]=1;**

**for(j=1;j<=n;j++)**

**if(v[j]==0 && (d[u]+c[u][j])<d[j]) d[j]=d[u]+c[u][j];**

**}**

**}**

**int main()**

**{**

**int c[10][10],d[10],i,j,s,sum,n; printf("\nEnter n value:"); scanf("%d",&n);**

**printf("\nEnter the graph data:\n"); for(i=1;i<=n;i++)**

**for(j=1;j<=n;j++) scanf("%d",&c[i][j]); printf("\nEnter the souce node:"); scanf("%d",&s);**

**dijkstra(c,n,s,d); for(i=1;i<=n;i++)**

**printf("\nShortest distance from %d to %d is %d",s,i,d[i]); return 0;**

**}**

**Input/Output**

1. Enter n value:6 Enter the graph data:

0 15 10 999 45 999

999 0 15 999 20 999

20 999 0 20 999 999

999 10 999 0 35 999

999 999 999 30 0 999

999 999 999 4 999 0

Enter the souce node:2

Shortest distance from 2 to 1 is 35

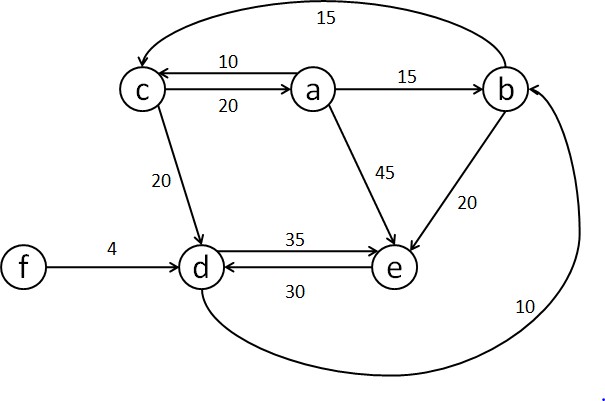
Shortest distance from 2 to 2 is 0

Shortest distance from 2 to 3 is 15

Shortest distance from 2 to 4 is 35

Shortest distance from 2 to 5 is 20

Shortest distance from 2 to 6 is 999



**Output:**

1. **enter the no. of nodes:**

6

enter the cost adjacency matrix,'9999' for no direct path

0 15 10 9999 45 9999

9999 0 15 9999 20 9999

|  |  |  |
| --- | --- | --- |
| 20 9999 | 0 20 | 9999 9999 |
| 9999 10 | 9999 0 | 35 9999 |
| 9999 9999 | 9999 30 | 0 9999 |
| 9999 9999 | 9999 4 | 9999 0 |

enter the starting vertex:

6

Shortest path from starting vertex to other vertices are

6->1=49

6->2=14

6->3=29

6->4=4

6->5=34

**6->6=0**