/\*IMPLEMENT DIJKSTRA’S ALGORITHM\*/

#include<iostream>

#include<climits>

using namespace std;

#define vertex 7 /\*total no of vertices in the graph\*/

int minDist(int dist[], bool Dset[])

{

int min=INT\_MAX,index;

for(int v=0;v<vertex;v++)

{

if(Dset[v]==false && dist[v]<=min)

{

min=dist[v];

index=v;

}

}

return index;

}

void dijkstra(int graph[vertex][vertex],int src) /\*Method to implement shortest path algorithm\*/

{

int dist[vertex];

bool Dset[vertex];

for(int i=0;i<vertex;i++) /\*Initialize distance of all the vertex to INFINITY and Dset as false\*/

{

dist[i]=INT\_MAX;

Dset[i]=false;

}

dist[src]=0; /\*Initialize the distance of the source vertex to zero\*/

for(int c=0;c<vertex;c++)

{

int u=minDist(dist,Dset);

Dset[u]=true;

for(int v=0;v<vertex;v++)

{

if(!Dset[v] && graph[u][v] && dist[u]!=INT\_MAX && dist[u]+graph[u][v]<dist[v])

dist[v]=dist[u]+graph[u][v];

}

}

cout<<"Vertex\t\tDistance from source"<<endl;

for(int i=0;i<vertex;i++) /\*will print the vertex with their distance from the source to the console \*/

{

char c=65+i;

cout<<c<<"\t\t"<<dist[i]<<endl;

}

}

int main()

{

int graph[vertex][vertex]={{0,5,3,0,0,0,0},{0,0,2,0,3,0,1},{0,0,0,7,7,0,0},{2,0,0,0,0,6,0},{0,0,0,2,0,1,0},{0,0,0,7,7,0,0},{0,0,0,0,1,0,0}};

dijkstra(graph,0);

return 0;