```c
// Given a graph find shortest paths from source to all nodes using Dijkstraâ€™s shortest path
// algorithm.

#include<stdio.h>//Standard input output
int main()//Main function
{
    int i,j,n,source,min=999,u,w;

    printf("Enter no of vertices:");
    scanf("%d",&n);//Input number of vertices of graph
    int visited[n+1],cost[n+1][n+1],d[n+1];
    printf("Enter the cost adjacency matrix(Enter 999 for not connnected)\n");
    for(i=1;i<=n;visited[i++]=0)
        for(j=1;j<=n;j++)
            scanf("%d",&cost[i][j]);//Input cost matrix
    printf("\nEnter the source node(1 indexed):");
    scanf("%d",&source);//Input source node
    int path[n+1];
    for(i=1;i<=n;path[i++]=source)
        d[i]=cost[source][i];//Calculating initial distance from source
    visited[source]=1;//Making source node as visited
    d[source]=0;//Distance source node is 0
    for(j=2;j<=n;j++,min=999)
    {   for(i=1;i<=n;i++)
            if(!visited[i]&&d[i]<min)
                    min=d[u=i];//Find minimum index and cost
                                //from current node
    visited[u]=1;//Mark next node as visited which has minimum distance

    for(int w=1;w<=n;w++)
        if(!visited[w])//Check if not visited
            if(d[w]>cost[u][w]+d[u])//Find if there is a path with lower cost
            {   d[w]=cost[u][w]+d[u];//If yes,make that as minimum distance
                path[w]=u;
            }
    }
    for(i=1;i<=n;i++)
        if(i!=source)
        {   printf("\nShortest path from %d to %d is %d\nShortest Path=%d",source,i,d[i],i);
            j=i;
            do
            {printf("<-%d",j=path[j]);
            }while (j!=source);
        }
    printf("\n");
    //Print the shortest path between source node and all the other vertices
}
```