**Dijkstra’s Algorithm**

#include<stdio.h>

#include<conio.h>

#define INFINITY 9999

#define MAX 10

void dijkstra(int G[MAX][MAX],int n,int startnode);

int main()

{

    int G[MAX][MAX],i,j,n,u;

    printf("Enter no. of vertices:");

    scanf("%d",&n);

    printf("\nEnter the adjacency matrix:\n");

    for(i=0;i<n;i++)

        for(j=0;j<n;j++)

            scanf("%d",&G[i][j]);

    printf("\nEnter the starting node:");

    scanf("%d",&u);

    dijkstra(G,n,u);

    return 0;

}

void dijkstra(int G[MAX][MAX],int n,int startnode)

{

    int cost[MAX][MAX],distance[MAX],pred[MAX];

    int visited[MAX],count,mindistance,nextnode,i,j;

    //pred[] stores the predecessor of each node

    //count gives the number of nodes seen so far

    //create the cost matrix

    for(i=0;i<n;i++)

        for(j=0;j<n;j++)

            if(G[i][j]==0)

                cost[i][j]=INFINITY;

            else

                cost[i][j]=G[i][j];

    //initialize pred[],distance[] and visited[]

    for(i=0;i<n;i++)

    {

        distance[i]=cost[startnode][i];

        pred[i]=startnode;

        visited[i]=0;

    }

    distance[startnode]=0;

    visited[startnode]=1;

    count=1;

    while(count<n-1)

    {

        mindistance=INFINITY;

        //nextnode gives the node at minimum distance

        for(i=0;i<n;i++)

            if(distance[i]<mindistance&&!visited[i])

            {

                mindistance=distance[i];

                nextnode=i;

            }

            //check if a better path exists through nextnode

            visited[nextnode]=1;

            for(i=0;i<n;i++)

                if(!visited[i])

                    if(mindistance+cost[nextnode][i]<distance[i])

                    {

                        distance[i]=mindistance+cost[nextnode][i];

                        pred[i]=nextnode;

                    }

        count++;

    }

    //print the path and distance of each node

    for(i=0;i<n;i++)

        if(i!=startnode)

        {

            printf("\nDistance of node%d=%d",i,distance[i]);

            printf("\nPath=%d",i);

            j=i;

            do

            {

                j=pred[j];

                printf("<-%d",j);

            }while(j!=startnode);

    }

}