# SSN College of Engineering, Kalavakkam

# Department of Computer Science and Engineering

# III Semester - CSE 'A ',’B’ & ‘C’

# UCS 1312 Data Structures Lab

# Academic Year: 2019-2020 Batch: 2018-2022

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Name : Rahul Ram M**

**Class : CSE – B**

**Register Number : 185001121**

**Assignment : 12**

**//Djikstra.h**

#include<stdio.h>

#define INFINITY 9999

#define MAX 10

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=1;i<=n;i++)

for(j=1;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=1;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)

{

mindistance=INFINITY;

//nextnode gives the node at minimum distance

for(i=1;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=1;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=1;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);

}

}

---------------------------------------------------------------------------------------------

**//main**

#include “Djikstra.h”

int main()

{

int G[MAX][MAX],i,j,n,u,ch=1;

while(ch!=0)

{

printf("Enter no. of vertices:");

scanf("%d",&n);

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

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

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

scanf("%d",&G[i][j]);

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

scanf("%d",&u);

dijkstra(G,n,u);

printf("Enter 1 to continue and 0 to exit:");

scanf("%d",&ch);

}

return 0;

}

----------------------------------------------------------------------------------------------------

**Output:**

Enter no. of vertices:7

Enter the adjacency matrix:

0 2 0 1 0 0 0

0 0 0 3 10 0 0

4 0 0 0 0 5 0

0 0 2 0 2 8 4

0 0 0 0 0 0 6

0 0 0 0 0 0 0

0 0 0 0 0 1 0

Enter the starting node:1

Distance of node2=2

Path=2->1

Distance of node3=3

Path=3->4->1

Distance of node4=1

Path=4->1

Distance of node5=3

Path=5->4->1

Distance of node6=6

Path=6->7->4->1

Distance of node7=5

Path=7->4->1

Enter 1 to continue and 0 to exit:1

Enter no. of vertices:6

Enter the adjacency matrix:

0 5 0 6 10 0

5 0 1 0 2 7

0 1 0 0 0 8

6 0 0 0 3 0

10 2 0 3 0 4

7 0 8 0 5 0

Enter the starting node:1

Distance of node2=5

Path=2->1

Distance of node3=6

Path=3->2->1

Distance of node4=6

Path=4->1

Distance of node5=7

Path=5->2->1

Distance of node6=11

Path=6->5->2->1

Enter 1 to continue and 0 to exit:0

------------------------------------------------------------------------------------------------------------