26/12/22 LAB PROGRAM: 2

Q. Implement Dykystra's to a graph to find the shortest minimum distance

Code:

```
#include<stdio.h>
#include<conio.h>
#define INFINITY 9999
#define MAX 10

void dykstra(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("\n Enter the adjacency matrix :\n");
for(i=0; i<n; i++)
```

```
void dijkstra (int G[MAX][MAX],
int startnode)
{
int cost[MAX][MAX], distance[i
pred[MAX];
int visited[MAX], count, mindista
nextnode, i, j;
for(i=0; i<n; i++)
for(j=0; j<n; j++)
if(G[i][j]==INFINITY)
else
cost[i][j]=G[i][j]
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
for(i=0; i<n; i++)
if(distance[i]<mindistance&visited
{
mindistance=distance[i]
nextnode=i;
```

```
}
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++;
}
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
{
i = pred[j];
printf(" <- %d", j)
} while (j != startnode);
}
}
```

output:-

Enter no of vertices : 4

Enter the adjacency matrix :

0 0 0 1
0 0 1 1
1 1 0 1
1 1 1 0

Enter the starting node = 1

Distance of node 0 = 2
Path = 0 ← 2 ← 1
Distance of node 2 = 1
path = 2 ← 1
Distance of node 3 = 1