12/1/23

Lab 9

Aim : Implement Dijkstras algorithm to find shortest path for given topology

```
#include <stdio.h>
# include < conio.h>
#define INFINITY 999
# define MAX 10

void dijkstra int G[MAX][MAX], int n, int startnode);
int node (){
    int G[MAX][MAX], i, j, n, u;
    printf(" Enter no. of vertices");
    scanf("%d", &n);
    printf("\n Enter adjacency matrix: \n ");
    for (i=0; i<n; i++)
        for(j=0; j<n; j++)
            scanf(" %d",& G[i][j] );
    printf(" Enter starting node:");
    scanf("%d",&n);
    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;
  for (i=0; i<n; i++)
    for (j=0; j<n; j++)
       if (G[i][j]==0)
           Cost[i][j]=INFINTY;
       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[i]){
        mindistance = distance[i];
        nextnode = i;
      }
     visited [nextnode]=1;
    for(i=0; i<n; i++)
          if (!visited(i]) {
            if (mindistance nextnode[i]) < distance(i]){
            distance [i]= mindistance + cost[nextnode][i];
            pred(i] =nextnode;
            }
```

```
count++;
for (i=0; i<n; i++)
        if (i!= startnode){
          printf("\n distance of node %d = %d",i,distanc[i
          j=i;
          do{
           j= pred[j];
            printf(" %d",j);
          }
          while (j != startnode);
      }
    }
```

Output:

```
Enter no. of vertices: 4
    Enter adjacency matrix
```

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 2 & 1 & 1 & 0 & 1 \\ 3 & 1 & 0 & 1 & 0 \end{array}$$

```
Enter starting node : 1
Distance of 0 =1
Path = 0 ← 1
Distance of 2 = 1
path = 2 ← 1
Distane of 3 = 2
path = 3 ← 0 ← 1
```