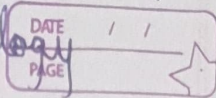


12/10/2023 Implement Dijkstra's algorithm to compute the shortest path for a given topology



```
#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("Enter the 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] = 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[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,  
           distance[i]);
```

```
    printf("\n Path = %d", i);  
    j = i;
```

```
    do
```

```
{  
    j = pred[j];  
    printf("← %d", j);
```

```
} while (j != startnode);
```

```
}  
}
```


O/P

Enter no. of vertices: 4

Enter the adjacency matrix:

0	5	9999	9999
2	0	4	9999
9999	9999	0	6
4	7	5	0

Enter the starting node: 0

Distance of node 1 = 5

Path = $1 \leftarrow 0$

Distance of node 2 = 9

Path = $2 \leftarrow 1 \leftarrow 0$

Distance of node 3 = 15

Path = $3 \leftarrow 2 \leftarrow 1 \leftarrow 0$

✓
N
12/1/23