10-12-2020

Computer Network

Yashasvi. R
1BM18CS154
5th Sem, B sec

1) Implement Dijkstra's algorithm to compute the shortest path through a graph.

```
#include <bits/stdc++.h>
using namespace std;
#define V 9
int minDistance(int dist[], bool sptSet[])
{
    int min = 9999, min_indesc;
    for (int v=0; v<V; v++)
        if (sptSet[v] == false && dist[v] <= min)
            min = dist[v], min_indesc = v;
    return min_indesc;
}
void printPath(int parent[], int j)
{
    if (parent[j] == -1)
      return;
    printPath(parent, parent[j]);
    cout << j << " ";
}
void printSolution(int dist[], int n, int parent[])
{ int src=0;
    cout << "Vertex\t Distance\t Path" << endl;
    for (int i=1; i<V; i++)
    {
```

```
        cout << "\n" << src << " -> " << i << "\t" << dist[i] << "\t"
                                                    << src << "";
        printPath(parent, i);
    }
}
void dijkstra(int graph[V][V], int src)
{
    int dist[V];
    bool sptSet[V];
    int parent[V];
    for (int i=0; i<V; i++)
    {
        parent[0] = -1;
        dist[i] = 9999;
        sptSet[i] = false;
    }
    dist[src] = 0;
    for (int count=0; count < V-1; count++)
    {
        int u = minDistance(dist, sptSet);
        sptSet[u] = true;
        for (int v=0; v<V; v++)
            if (!sptSet[v] && graph[u][v] && dist[u]+graph[u][v]
                                                < dist[v])
            {
                parent[v] = u;
                dist[v] = dist[u] + graph[u][v];
            }
    }
    printSolution(dist, V, parent);
}
```

```
int main()
{
    int graph[V][V];
    cout << "Enter the graph (Enter 99 for infinity): " << endl;
    for (int i=0; i<V; i++)
    {
        for (int j=0; j<V; j++)
            cin >> graph[i][j];
    }
    cout << "Enter the Source: " << endl;
    int src;
    cin >> src;
    dijkstra(graph, src);
    cout << endl;
    return 0;
}
```