27/11/2020

Cycle-2

CN - lab

Triveni.y

PAGE NO:
DATE: / /

prgm 3) Implement Dijkstra's algorithm to compute shortest path for a given topology.

```
import sys

class Graph:

    def _init_(self, vertices):
        Self.V = vertices
        self.graph = [[0 for column
                       in range(vertices)]
                      for row in
                       range(vertices)]

 def printSolution(self,dist):
    print("Vertex \t Distance from
              Source")
    for node in range(self,V):
        print (node, "\t", dist[node]).
```

Triveni.y

```
def minDistance (self, dist, Sptset) :

    min = sys.maxSize
    for v in range(self:V):
        if dist[v] <min   and   sptSet[v]==
                                    False:
            min=dist[v]
            min_index=v
    return min_index

def dijkstra ( self,src):

    dist = [sys.maxSize] * self.V
    dist[src]=0
    sptSet =[False] * self.V

    for cout in range (self.V):

        u= self.minDistance(dist,sptSet)
        sptSet[u) = True

        for v in range (self.V).

            if  self.graph[u][v]>0  and
                sptSet[v] ==False
    and dist[v] > dist[u] + self.graph[u][v]:
        dist[v] = Dist[u] + self.graph[u][v].
self.printSolution(dist)
```