Exp-9

Venkatesha Havada (4
1BM18CS124

Question: Use distance vector algorithm to find shortest distance between all points in the given topology.

Ans:

code

```
Class Topology :

    def __init__(self, array_of_nodes):
        self.nodes = array_of_nodes
        self.edges = []

    def add_direct_connection(self, P1, P2, cost):
        self.edges.append((P1, P2, cost))
        self.edges.append((P2, P1, cost))

    def distance_vector_routing(self):
        import collections
        for node in self.nodes:
            dist = collections.defaultdict(int)
            next_hop = {node: node}
```

~~for~~ for other_node in self.nodes:

if other_node != node:

dist[other_node] = 10000000

for i in range(len(self.nodes)-1):

for edge in self.edges:

src, dest, cost = edge

~~if dist[dest] = dist[src]~~

~~dist[~~

if dist[src] + cost < dist[dest]:

dist[dest] = dist[src] + cost

if src == node:

next_hop[dest] = dest

elif src in next_hop:

next_hop[dest] = next_hop[src]

self.print_routing_table(node, dist, next_hop)

print()

```
def print_routing_table(self, node, dist, next_hop):
    print(f'Routing table for {node}:')
    print('Dest \t Cost \t Next Hop')
    for dest, cost in dist.items():
        print(f'{dest} \t {cost} \t {next_hop
```

```
nodes = input('Enter the nodes: ').split()
t = Topology(nodes)
edges = int(input('Enter the number of connections:'))
for _ in range(edges):
    src, dest, cost = input('Enter [src] [dest] [cost]: ')
                      .split()
    t.add_direct_connection(src, dest, int(cost))
t.distance_vector_routing()
```

@harikant