LAB - Exam - 2

Naman Agarwal
1BM18CS057

```
class Topology:

        def init(self, array):
                self.nodes = array.
                self.edges = []

        def connect(self, point1, point2, weight):

                self.edges.append(point1, point2, weight)
                self.edges.append (point2, point1, weight)

        def distanceVectorRouting (self):
                import collections
                for point in self.nodes:

                        distance = collections.defaultdict(int)
                        next_hop = {~~node~~ point: node }
                        for points in self.nodes:

                                if points != point :

                                        distance(points) = 99999

                for i in range( self.nodes.__len__ - 1):
                        for edge in self.edges:

                                source, destination, weight = edge.
                                if ~~destination~~ distance [source] + weight
                                        < ~~destination~~ distance [destination]:
```

N. Agarwal

```
distance [destination] = distane [destination] + weight

        if   source == point :
                    next_hop [destination ] = destination.

            else if   source in  next_hop:
                    next hop [destination ] = next_hop[sour

    self. print_table (node, distance, next_hop)

def print_table ( self, node, dist, next_hop)

        for  dst , cost  in  dist. items():

                print( f'{dst} \t {cost} \t {next_hop
                                        [dst]}')

nodes = ['A', 'B', 'C', 'D', 'E', 'F', 'G']

    t = Topology (nodes).

    t. ~~add~~ connect ( 'A', 'B', 4)
    t. connect ( 'A', 'C', 5)
       "       ('A', 'D', 3)
       "       ('B', 'C', 2)
       "       ('B', 'F', 3)
       "       ('B', 'G', 4)
       "       ('C', 'D', 6)
       "       ('C', 'E', 4)
       "       ('C', 'F', 4)

       "       ('D', 'E', 3)
       "       ('E', 'F', 2)
       "       ('F', 'G', 5)

    t. distance Vector
       Routing()
```