23/11/20

AKSHAY MITTUR
1BM18CS010
CN LAB WRITEUP

DISTANCE VECTOR ALGORITHM

```
class Graph:
    def __init__(self, n):
        self.matrix = []
        self.n = n
    def addEdge(self, u, v, w):
        self.matrix.append((u,v,w))
    def printArr(self, dist, src):
        print("Vector Table of {} ".format(chr(ord('A')+src)))
        for i in range(self.n):
            print("{0} \t{1} ".format(chr(ord('A')+i), dist[i]))

    def BellmanFord(self, src):
        dist = [99] * self.n
        dist[src] = 0
        for _ in rang(self.n-1):
            for u, v, w in self.matrix:
                if dist[u] != 99 and dist[u] + w < dist[v]:
                    dist[v] = dist[u] + w
        self.printArr(dist, src)

def main():
    matrix = []
    print("Enter Nodes")

    n = int(input())
```

AKSHAY MITTOR
1BM18CS010

23/11/20

```
print("Enter Adjacency Matrix ")
for i in range(n):
    m = list(map(int, input().split(" ")))
    matrix.append(m)
g = Graph(n)
for i in range(n):
    for j in range(n):
        if matrix[i][j] == 1:
            g.addEdge(i,j,1)
for _ in range(n):
    g.BellmanFord(_)

main()
```