CN LAB

Chandan C Bagevi
1BM18CS026

program for 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("Vertex Table of {}".format(chr(ord('A')+src)))
        for i in range(self.n):
            print("{}\t\t{}".format(chr(ord('A')+i), dist[i]))
    def BellmanFord(self, src):
        dist = [99] * self.n
        dist[src] = 0
        for _ in range(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 no of nodes:")
    n = int(input())
    print("Enter the Adjacency matrix:")
    for i in range(n):
        n = list(map(int, input().split(" ")))
        matrix.append(n)
    g = Graph(n)
```

```
for i in range(n):
    for j in range(n):
        if matrix[i][j] == 1:
            g.addEdge(i, j, 1)
for i in range(n):
    g.BellmanFord(_)

main().
```