Program (Dijkstra's Algo)

```
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 Distance from Source")
        print("vertex cost")
        for node in range(self.V):
            print(node, "", dist[node])

    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)

g print("Enter number of vertex:")
n = int(input())
g = Graph(n)
print("Enter graph:")
for i in range(n):
    a = []
    a = list(map(int, input().split(" ")))
    g.graph.append(a)

print("Enter src:")
src = int(input())
g.dijkstra(src)
```