import math

def get\_link\_v(v,D):

for i, weight in enumerate(D[v]):

if weight>0:

yield i

def arg\_min(T, S):

amin = -1

m = math.inf # максимальное значение

for i, t in enumerate(T):

if t < m and i not in S:

m = t

amin = i

return amin

D = ((0, 3, 1, 3, 0, 0),

(3, 0, 4, 0, 0, 0),

(1, 4, 0, 0, 7, 5),

(3, 0, 0, 0, 0, 2),

(0, 0, 7, 0, 0, 4),

(0, 0, 5, 2, 4, 0)

N = len(D) # число вершин в графе

T = [math.inf]\*N # последняя строка таблицы

v = 0 # стартовая вершина (нумерация с нуля)

S = {v} # просмотренные вершины

T[v] = 0 # нулевой вес для стартовой вершины

while v != -1: # цикл, пока не просмотрим все вершины

for j in get\_link\_v(v, D): # перебираем все связанные вершины с вершиной v

if j not in S: # если вершина еще не просмотрена

w = T[v] + D[v][j]

if w < T[j]:

T[j] = w

v = arg\_min(T, S) # выбираем следующий узел с наименьшим весом

if v > 0: # выбрана очередная вершина

S.add(v) # добавляем новую вершину в рассмотрение

print(T)