“Homework 6 part B” Justin Minsk

>>> from tsp\_solver.greedy import solve\_tsp as tsp

>>> D = [[0, 1, 2],[1, 0, 3],[2, 3, 0]]

#sample matrix that the column and row so row 0 and column 0 are the point the rest are the distance from the point in that row

#for example here the first row the point is 0 the second point is 1 unit away and the third is 2 points away

>>> path = tsp(D)

>>> print(path)

[1, 0, 2]

#start at point 1 go to point 0 then point 2, the points being the row of the matrix each column entry is how far away it is to each other point

>>> import numpy

>>> matrix = numpy.random.random((3,3))

#creates a random matrix

>>> matrix

array([[ 0.41390655, 0.7579234 , 0.15275978],

[ 0.18352394, 0.97409547, 0.57803915],

[ 0.80402028, 0.86947759, 0.87691279]])

>>> path = tsp(matrix)

>>> print(path)

[0, 2, 1]

#this one would be shortest to go from point 0 (row 0) to point 2 then 1

>>> matrix = numpy.random.random((10,10))

#create a random matrix with a much larger matrix

>>> path = tsp(matrix)

>>> matrix

array([[ 8.04209808e-01, 4.14786293e-01, 2.23829932e-01,

6.81085420e-02, 4.50777855e-01, 3.53091058e-02,

9.81083068e-01, 4.20394904e-01, 3.88599161e-02,

4.30432650e-01],

[ 7.99749096e-01, 7.76019356e-01, 5.23697591e-01,

8.91501984e-04, 3.15233095e-01, 9.38221564e-01,

9.49344820e-01, 9.74855246e-01, 1.99487649e-01,

2.11510822e-01],

[ 4.46618602e-01, 7.83388571e-01, 3.01371647e-01,

7.01823158e-01, 2.12082263e-01, 3.47255412e-01,

7.42723811e-01, 6.22991338e-01, 2.67061173e-01,

4.55297065e-01],

[ 8.54367977e-01, 9.54187501e-01, 2.31266718e-01,

1.29514759e-01, 9.35719583e-01, 1.74859065e-01,

2.36029336e-01, 4.11806093e-01, 8.18177338e-01,

4.06825151e-01],

[ 6.57826896e-01, 6.72759944e-02, 8.97474381e-01,

2.90777064e-01, 2.90092388e-01, 8.40794020e-01,

6.67711833e-01, 1.21553086e-01, 6.57450254e-01,

5.96611953e-01],

[ 5.63719773e-01, 7.94041164e-01, 1.34639414e-01,

3.04711748e-01, 1.00743891e-01, 7.11799652e-01,

4.12788571e-01, 4.79250613e-01, 6.77670043e-01,

6.45058573e-01],

[ 7.18912619e-01, 6.58158039e-02, 2.25887091e-01,

9.23315583e-01, 4.54879290e-01, 4.86074664e-01,

6.98924605e-01, 6.77793478e-01, 3.76221171e-02,

9.72524904e-01],

[ 2.80560420e-01, 5.21482327e-01, 6.17230309e-01,

3.16716098e-02, 1.82019445e-01, 1.80414239e-01,

7.88617653e-01, 7.28438374e-01, 9.03664496e-01,

1.20484516e-01],

[ 8.92618193e-01, 9.87506921e-01, 7.06334467e-01,

7.62688984e-01, 3.98718009e-01, 3.67814569e-01,

3.11528207e-01, 8.48861597e-01, 3.24467007e-01,

5.49394675e-01],

[ 8.64632437e-01, 6.55664643e-01, 1.07399571e-01,

6.33664727e-01, 8.11799769e-01, 7.42744327e-01,

5.97170457e-01, 4.01909843e-02, 7.43194284e-01,

3.59888199e-02]])

>>> print(path)

[2, 9, 7, 4, 1, 3, 5, 0, 8, 6]

#it is found that you would go in the order above to the go the least distance

Overall you must use a square matrix to get the point then distances from that point. Would need to develop a good loop to fill your matrix or take the time to formulate the matrix, then be able to let it be solved. This is all prep work and if you already had this or could formulate it this will solve it without you needing to write more code.