* NIKHITHA A[ENG18CS0192]
* NISHA V[ENG18CS0194]
* PAVANA M[ENG18CS0207]
* PAVITHRA H A[ENG18CS0208]

**URBAN INFRASTRUCTURE DEVELOPMENT PROGRAM CODE**

import networkx as nx

import matplotlib.pyplot as plt

import operator

import time

def k\_centers(G, n):

centers = []

cities = list(G.nodes())

#add an arbitrary node, here, the first node,to the centers list

#print(G.nodes()[0])

print(cities)

#centers.append(cities[0])

centers.append(list(G.nodes())[0])

cities.remove(centers[0])

n = n-1 #since we have already added one center

#choose n-1 centers

while n!= 0:

city\_dict = {}

for cty in cities:

min\_dist = float("inf")

for c in centers:

min\_dist = min(min\_dist,G[cty][c]['length'])

city\_dict[cty] = min\_dist

#print city\_dict

new\_center = max(city\_dict, key = lambda i: city\_dict[i])

#print new\_center

centers.append(new\_center)

cities.remove(new\_center)

n = n-1

#print centers

return centers

#takes input from the file and creates a weighted undirected graph

def CreateGraph():

G = nx.Graph()

f = open('input.txt')

n = int(f.readline()) #n denotes the number of cities

wtMatrix = []

for i in range(n):

list1 = list(map(int, (f.readline()).split()))

wtMatrix.append(list1)

#Adds egdes along with their weights to the graph

for i in range(n) :

for j in range(n)[i:]:

print(wtMatrix[i][j])

G.add\_edge(i, j, length = wtMatrix[i][j])

noc = int(f.readline()) #noc,here,denotes the number of centers

return G, noc

#draws the graph and displays the weights on the edges

def DrawGraph(G, centers):

pos = nx.spring\_layout(G)

color\_map = ['blue'] \* len(G.nodes())

#all the center nodes are marked with 'red'

for c in centers:

color\_map[c] = 'red'

nx.draw(G, pos, node\_color = color\_map, with\_labels = True) #with\_labels=true is to show the node number in the output graph

edge\_labels = nx.get\_edge\_attributes(G, 'length')

nx.draw\_networkx\_edge\_labels(G, pos, edge\_labels = edge\_labels, font\_size = 11) #prints weight on all the edges

#main function

if \_\_name\_\_ == "\_\_main\_\_":

start\_time=time.time()

G,n = CreateGraph()

centers = k\_centers(G, n-1)

DrawGraph(G,centers)

end\_time=time.time()

print(end\_time-start\_time)

plt.show()