**Name : Ayush Patil**

**Roll no : TYCOC210**

**Batch C-3**

**prn no : 122B1B210**

**Assignment - 3**

**Title** :

You have a business with several offices; you want to lease phone lines to connect them up with each other; and the phone company charges different amounts of money to connect different pairs of cities.You want a set of lines that connects all your offices with a minimum total cost. Solve the problem by Floyd-Warshall algorithm.

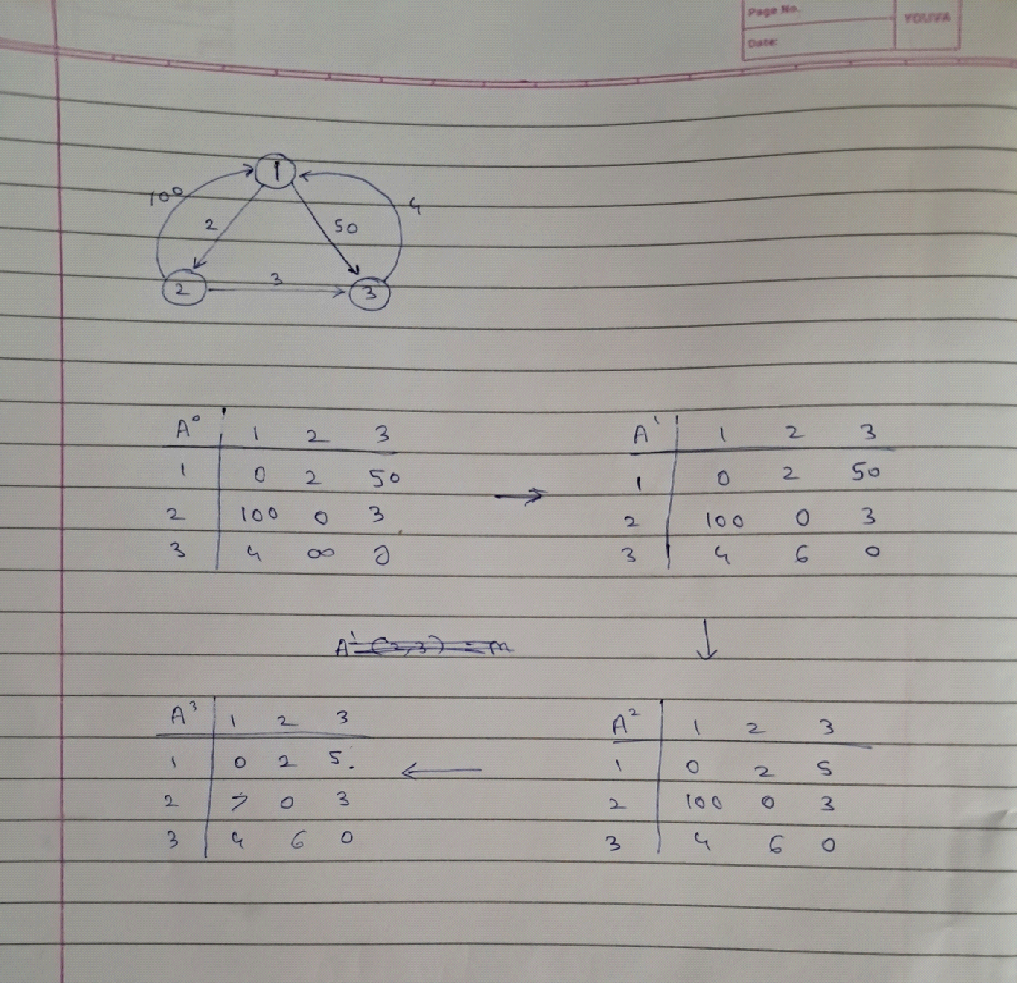
**Theory :**

The Floyd-Warshall algorithm is a dynamic programming technique used to find the shortest paths between all pairs of vertices in a weighted graph. This algorithm is particularly useful for graphs with dense connections and can handle both positive and negative edge weights, though it cannot handle negative cycles.

The Floyd-Warshall algorithm works by considering all pairs of vertices and updating the shortest paths iteratively. It uses a matrix to represent the distances between each pair of vertices, initially setting the distance to infinity for all pairs except for the diagonal (distance from a vertex to itself), which is set to zero.

Time Complexity: O(V3), where V is the number of vertices in the graph and we run three nested loops each of size V.

Auxiliary Space: O(V2), to create a 2-D matrix in order to store the shortest distance for each pair of nodes.



**Code :**

import numpy as np

def floyd\_warshall(cost\_matrix):

num\_vertices = len(cost\_matrix)

dist = np.array(cost\_matrix, copy=True)

for k in range(num\_vertices):

for i in range(num\_vertices):

for j in range(num\_vertices):

if dist[i][j] > dist[i][k] + dist[k][j]:

dist[i][j] = dist[i][k] + dist[k][j]

return dist

def main():

num\_offices = int(input("Enter the number of offices: "))

cost\_matrix = []

print("Enter the cost matrix")

for i in range(num\_offices):

row = input().split()

cost\_row = []

for cost in row:

if cost.lower() == 'inf':

cost\_row.append(float('inf'))

else:

cost\_row.append(int(cost))

cost\_matrix.append(cost\_row)

distances = floyd\_warshall(cost\_matrix)

print("\nShortest paths between all pairs of offices:")

print(np.array(distances))

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

main()

**Output :**

