NAME: ASIF ERFAN KHAN

ROLL NUMBER: 546

COURSE: MSc CS

SUBJECT: ALGORITHM

TOPIC: FLOYD-WARSHALL ALGORITHM

PRACTICAL 5

```
# Python3 Program for Floyd Warshall Algorithm


# Number of vertices in the graph

V = 4


# Define infinity as the large

# enough value. This value will be

# used for vertices not connected to each other

INF = 99999


# Solves all pair shortest path

# via Floyd Warshall Algorithm



def floydWarshall(graph):

        """ dist[][] will be the output

        matrix that will finally

                have the shortest distances

                between every pair of vertices """

        """ initializing the solution matrix

        same as input graph matrix

        OR we can say that the initial

        values of shortest distances

        are based on shortest paths considering no

        intermediate vertices """


        dist = list(map(lambda i: list(map(lambda j: j, i)), graph))


        """ Add all vertices one by one

        to the set of intermediate
```

```
vertices.
---> Before start of an iteration,
we have shortest distances
between all pairs of vertices
such that the shortest
distances consider only the
vertices in the set
{0, 1, 2, .. k-1} as intermediate vertices.
----> After the end of a
iteration, vertex no. k is
added to the set of intermediate
vertices and the
set becomes {0, 1, 2, .. k}
"""
for k in range(V):

    # pick all vertices as source one by one
    for i in range(V):

        # Pick all vertices as destination for the
        # above picked source
        for j in range(V):

            # If vertex k is on the shortest path from
            # i to j, then update the value of dist[i][j]
            dist[i][j] = min(dist[i][j],
                            dist[i][k] + dist[k][j]
                            )
printSolution(dist)
```

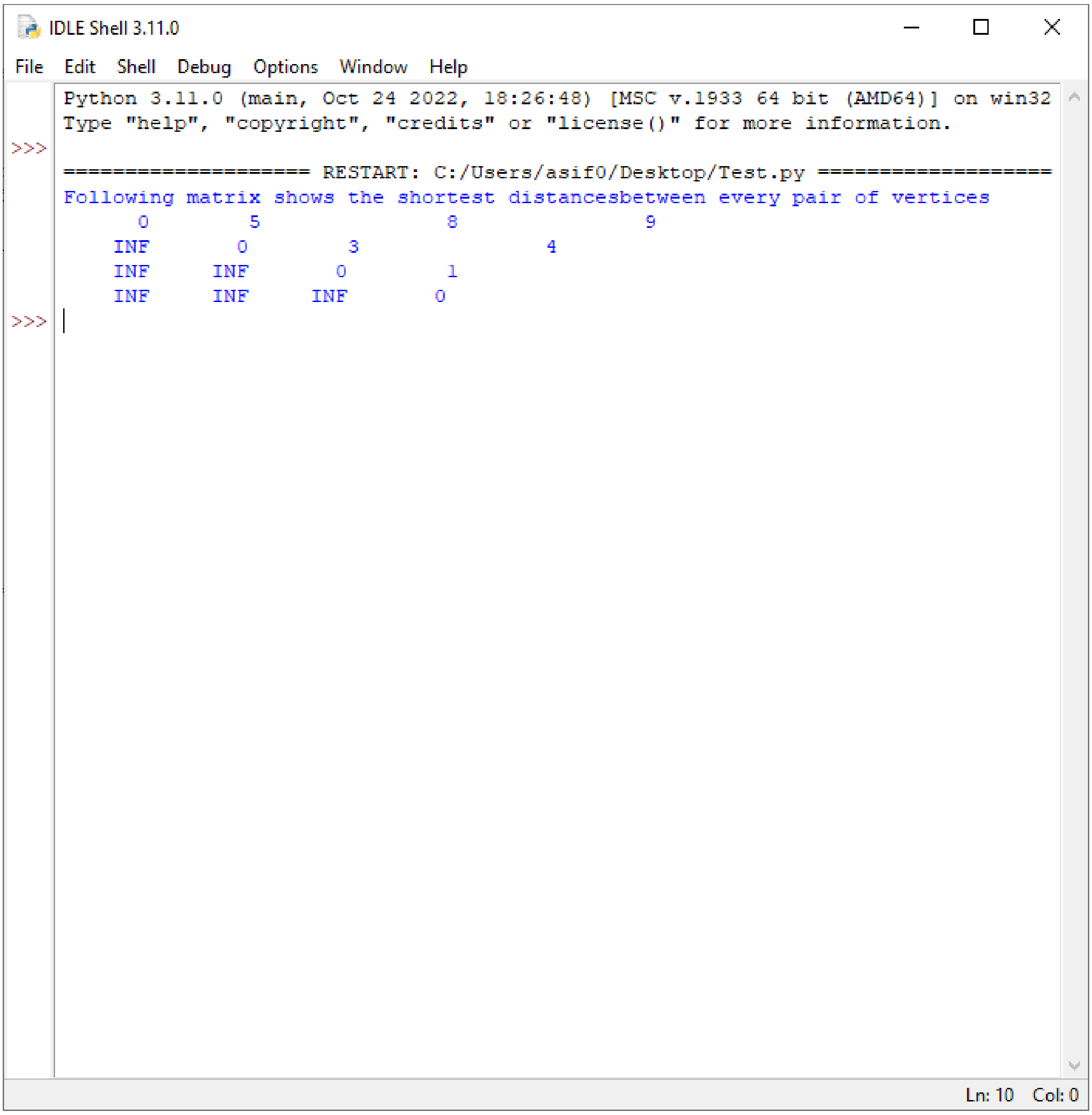
```
# A utility function to print the solution
def printSolution(dist):
    print("Following matrix shows the shortest distances\
between every pair of vertices")
    for i in range(V):
        for j in range(V):
            if(dist[i][j] == INF):
                print("%7s" % ("INF"), end=" ")
            else:
                print("%7d\t" % (dist[i][j]), end=' ')
            if j == V-1:
                print()



# Driver's code
if __name__ == "__main__":
   """
            10
        (0)------->(3)
        |           /|\
        5 |          |
        |            | 1
        \|/          |
        (1)------->(2)
            3               """
graph = [[0, 5, INF, 10],
        [INF, 0, 3, INF],
        [INF, INF, 0, 1],
        [INF, INF, INF, 0]
        ]
# Function call
```

floydWarshall(graph)

OUTPUT:



```
Python 3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 
==================== RESTART: C:/Users/asif0/Desktop/Test.py ===================
Following matrix shows the shortest distancesbetween every pair of vertices
      0        5               8               9
    INF       0        3               4
    INF     INF       0        1
    INF     INF     INF       0
>>> |
```