107. Floyd algorithm

Code:

```
def floyd_warshall(graph):
    V = len(graph)

    dist = [[float('inf')] * V for _ in range(V)]

    for i in range(V):
        for j in range(V):
            if i == j:
                dist[i][j] = 0
            elif graph[i][j] != 0:
                dist[i][j] = graph[i][j]

    for k in range(V):
        for i in range(V):
            for j in range(V):
                if dist[i][j] > dist[i][k] + dist[k][j]:
                    dist[i][j] = dist[i][k] + dist[k][j]

    return dist


graph = [
    [0, 3, float('inf'), 5],
    [2, 0, float('inf'), 4],
    [float('inf'), 1, 0, float('inf')],
    [float('inf'), float('inf'), 2, 0]
]

distances = floyd_warshall(graph)
for row in distances:
    print(row)
```

Output:

```
[0, 3, 7, 5]
[2, 0, 6, 4]
[3, 1, 0, 5]
[5, 3, 2, 0]
```

Time Complexity:

- T(n)= O(V^3)