## Part A

**Aim:**

1. Dynamic programming
2. All Pair Shortest Path (Floyd Algorithm)

**Prerequisite:** Any programming language

**Outcome:** Algorithms and their implementation

**Theory:**

The all pair shortest path algorithm is also known as Floyd-Warshall algorithm is used to find all pair shortest path problem from a given weighted graph. As a result of this algorithm, it will generate a matrix, which will represent the minimum distance from any node to all other nodes in the graph

**Procedure:**

1. Design algorithm and find best, average and worst-case complexity
2. Implement algorithm in any programming language.
3. Paste output

**Practice Exercise:**

| S.no | Statement |
|---|---|
| 1 | Applying dynamic programming methodology to find all pairs shortest path of a directed graph. |
| 2 | Find the run time complexity of the above algorithm |

**Instructions:**

1. Design, analysis and implement the algorithms.
2. Paste the snapshot of the output in input & output section.

## Part B

### Floyd-Warshall algorithm

**Input:** Adjacency matrix for a graph

**Output:** Shortest path adjacency matrix

**Algorithm:**

Shortest path algorithm:

```
def shortest_path(distance):
   l=len(distance)
   for i in range(l):
      for j in range(l):
         for k in range(l):
            distance[j][k]=min(distance[j][k],distance[j][i]+distance[i][k])
```

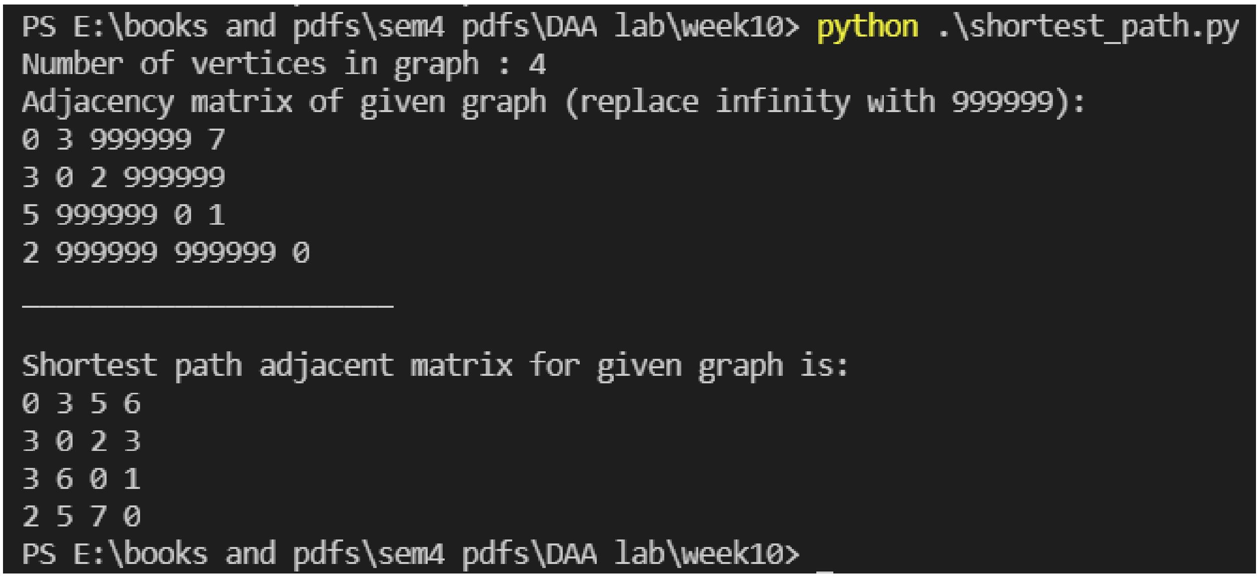
**Code:**

```
def shortest_path():
    global distance
    l=len(distance)
    for i in range(l):
        for j in range(l):
            for k in range(l):
                distance[j][k]=min(distance[j][k],distance[j][i]+distance[i][k])


def print_distance():
    for i in distance:
        print(*i)


n=int(input('Number of vertices in graph : '))
print('Adjacency matrix of given graph (replace infinity with 999999):')
graph=[list(map(int,input().split())) for i in range(n)]
distance=graph.copy()
shortest_path()
print('______________________\n\nShortest path adjacent matrix for given graph is:')
print_distance()
```

**Input & Output:**



```
PS E:\books and pdfs\sem4 pdfs\DAA lab\week10> python .\shortest_path.py
Number of vertices in graph : 4
Adjacency matrix of given graph (replace infinity with 999999):
0 3 999999 7
3 0 2 999999
5 999999 0 1
2 999999 999999 0
______________________

Shortest path adjacent matrix for given graph is:
0 3 5 6
3 0 2 3
3 6 0 1
2 5 7 0
PS E:\books and pdfs\sem4 pdfs\DAA lab\week10> _
```

**Run time complexity of All Pair Shortest Path (Floyd Algorithm) :**

The time complexity of Floyd’s algorithm is O(V^3) where V will be the number of vertices since we will run 3 loops, one for row, 1 for column, and one for intermediatory node.

Space complexity is O(V^2) as we implemented with help of a 2-D list.

**Observation & Learning:**

I have learned and observed that :

i) does not work along with the graph with negative cycles

ii) algorithm works with both directed and undirected graphs

**Conclusion:**

I have successfully written and executed Floyd’s algorithm in the python programming language.