MP-2 Tutorial - 3

190031920

Avuthu Nikhil Reddy

In [1]:
```
import sys
INF = sys.maxsize
```

In [2]:
```
def floydWarshall(graph):
  n = len(graph)
  dist = [[] for i in range(n)]
  for i in range(n):
    for j in range(n):
      dist[i].append(graph[i][j])


  for k in range(n):
    for i in range(n):
      for j in range(n):
        dist[i][j] = min(dist[i][j],dist[i][k]+ dist[k][j])

  print("shortest distance between every pair of vertices")
  for i in range(n):
    for j in range(n):
      if dist[i][j]==INF:
        print("%7s" %("INF"),end =' ')
      else:
        print("%7d" %(dist[i][j]),end=' ')
    print()
graph = [[0,5,INF,10],
             [INF,0,3,INF],
             [INF, INF, 0,   1],
             [INF, INF, INF, 0]
        ]
# Print the solution
floydWarshall(graph);
```

```
shortest distance between every pair of vertices
      0       5       8       9
    INF       0       3       4
    INF     INF       0       1
    INF     INF     INF       0
```

In [3]:
```
# Floyd Warshall Algorithm in python
# The number of vertices
nV = 4
INF = 999
```

In [4]:
```
# Algorithm implementation
def floyd_warshall(G):
    distance = list(map(lambda i: list(map(lambda j: j, i)), G))
    # Adding vertices individually
    for k in range(nV):
        for i in range(nV):
            for j in range(nV):
                distance[i][j] = min(distance[i][j], distance[i][k] + distance[k][j])
    print_solution(distance)
```

In [5]:
```
def print_solution(distance):
    for i in range(nV):
        for j in range(nV):
            if(distance[i][j] == INF):
               print("INF", end=" ")
            else:
                print(distance[i][j], end=" ")
        print(" ")
```

In [6]:
```
# Printing the solution

G = [[0, 3, INF, 5],
         [2, 0, INF, 4],
         [INF, 1, 0, INF],
        [INF, INF, 2, 0]]
floyd_warshall(G)
```

```
0 3 7 5
2 0 6 4
3 1 0 5
5 3 2 0
```