J.S. Meghana
1BM18CS039

```
class Graph:

def __init__(Self, vertices)

  Self.V = vertices
  Self.graph = []

def add_edge(Self, S, d, w):

   Self.graph.append([S, d, w])

def print_Solution(Self, dist):
  print("distance from Source
                to Vertex")
 for i in range(Self.V):

   print("{0}\t\t{1}".format
                    (i, dist[i]))
def bellman_ford(Self, Src):

 dist = [float("inf")] * Self.V
 dist[Src] = 0
 for _ in range(Self.V-1):
  for S, d, w in Self.graph:

  if dist[S] != float("inf") and
              dist[S] + W < dist[d]:
    dist[d] = dist[S]+W
 for S, d, w in Self.graph:
```

Meg.

JS Meghana
1BM18CS039

```
if dist [s] != float ("inf") and dist [s]
                              + w < dist [d]:
    print ("negative weight cycle)
    return
self. print_solution(dist)

g = Graph (5)
g. add_edge (0, 1, 5)
g. add_edge (0, 2, 4)
g. add_edge (1, 3, 3)
g. add_edge (2, 1, 6)
g. add_edge (3, 2, 2)

g. bellman_ford (0)
```

[illegible]