**Assignment 9**

**Name: Mudit Mohit**

**Enrolment No.: BT21GCS057**

**Section: B2**

**Question:** Implement the Bellman-Ford algorithm to solve the problem of finding the shortest path in a graph from a single source node to all other nodes (ex-single source shortest path problem). Assume that there are n nodes in the graph and that they have associated costs (which may be positive or negative). Analyze the complexity of the algorithm and calculate the optimal path and its cost value.

Ans. class Gra:

    def \_\_init\_\_(self, vertices):

        self.v = vertices

        self.graph = []

    def add\_edge(self, a, b, c):

        self.graph.append([a, b, c])

    def print\_solution(self, cost):

        print("Vertex Distance from Source")

        for i in range(self.V):

            print("{0}\t\t{1}".format(i, cost[i]))

    def bellman\_ford(self, src):

        cost = [float("Inf")] \* self.v

        cost[src] = 0

        for \_ in range(self.v - 1):

            for a, b, c in self.graph:

                if cost[a] != float("Inf") and cost[b] + c < cost[a]:

                    cost[b] = cost[a] + c

        for a, b, c in self.graph:

            if cost[a] != float("Inf") and cost[a] + c < cost[b]:

                print("Graph contain negative weight cycle")

                return

        self.print\_solution(cost)

g = Gra(5)

g.add\_edge(0, 2, 1)

g.add\_edge(0, 4, 8)

g.add\_edge(2, 6, 6)

g.add\_edge(4, 2, 1)

g.add\_edge(6, 4, 4)

g.bellman\_ford(0)

Output:

