Assignment-7

N – Number of vertices = |V|
M – Number of edges = |M|

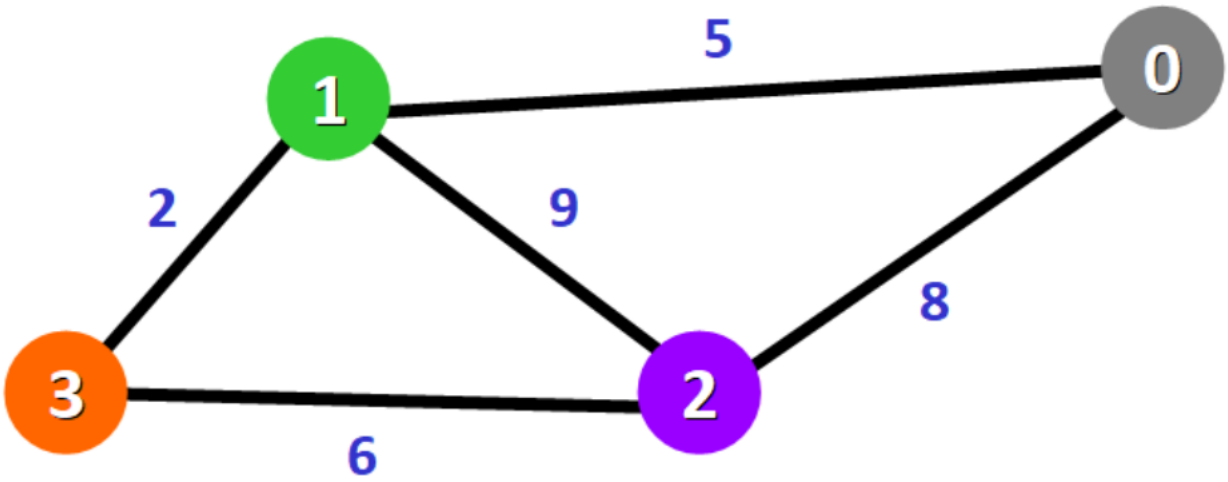
Ans-1) **Dijkstra Algorithm.**

Dijkstra uses a priority queue or min heap to find the shortest path from a single source to all other vertices when there is no negative edges or negative cycles present in the graph. It uses a greedy approach to find the optimal answer by always picking the element which is currently at the minimum distance from the source and then moving on the other vertices. Whenever a vertex is traversed we remove it from the set until the set is empty.

Time Complexity – O((N+M)*logN)

Drawbacks – Can't handle negative weight
– Can't detect negative cycles.

Example:-



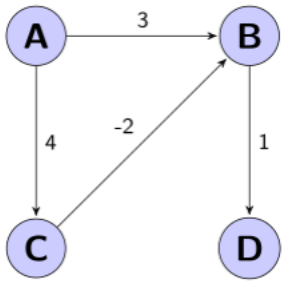
This graph consists only of positive edges and hence Dijkstra can be used in this graph from say source vertex 0.
Thus output distance will be: [0 , 5 , 8 , 7]

Ans-2) **Bellman-Ford Algorithm.**

In Bellman-Ford we relax each edge (n-1) times and try to find the minimum distance each time by relaxing one edge at a time (n-1) times.
This is based upon the idea that the maximum number of nodes except the source to the destination is at most (n-1). This is suffices to run the loop (n-1) times. It can detect that if there is a negative cycle based upon the fact that otherwise running the loop (n-1) times should give us the optimal answer.
So we run the same loop once more and if there is any such vertex where the distance can still be optimized that means there exists a negative cycle and distance decreased as a result of that.

Time Complexity – O(N*M)

Drawbacks – In case input consists of only positive edges it still takes a much higher time complexity and thus is much slower than Dijkstra in that case.



In this graph if we use Dijkstra from node 0 (A) distance is: [0, 3, 4, 4]

However, using Bellman-Ford we can get the optimal distance is: [0, 2, 4, 3]

This is because Dijkstra is based on greedy and fails when there is negative edges in graph.