The **shortest path problem** is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its constituent edges is minimized.

The shortest path algorithm can be either applied for single-pairs or for all pairs shortest path problem.

* The **single-source shortest path problem**, in which we have to find shortest paths from a source vertex *v* to all other vertices in the graph.
* The **single-destination shortest path problem**, in which we have to find shortest paths from all vertices in the directed graph to a single destination vertex *v*. This can be reduced to the single-source shortest path problem by reversing the arcs in the directed graph.
* The **all-pairs shortest path problem**, in which we have to find shortest paths between every pair of vertices *v*, *v'* in the graph.

There are a lot of algorithms made for the problem of finding the shortest path. The two algorithms belong to two classes of algorithms – Greedy and Dynamic programming.

**Johnson's algorithm** is a way to find the shortest paths between all pairs of vertices in a sparse, edge weighted, directed graph.

In order to implement this algorithm, **Bellman-Ford algorithm** and **Dijkstra’s algorithm** is used to find the shortest distance. Johnson’s algorithm is also used for negative weights (but not negative cycles) but since we are interested in distance, this is used for finding the shortest distance.