**Flyod-Warshall Algorithm** : This is an APSP ( All Pair Shortest Path)algorithm, which will traverse through every node of the graph to find out the shortest path to each node. i.e. "n" nodes will be figuring out shortest path to "n-1" nodes. The known worst case complexity for Flyod-Warshall is **O(no. of vertices)3** i.e **O(V)3**

**Dijkstra Algorithm** : This is a Single Source Shortest Path Algorithm, which means it will fix a source first and then will calculate the shortest path to "n-1" nodes. It doesn't go around and find the shortest path for each node available. It fixes it's root node, and from there the shortest path to every other node will be calculated. The known worst case complexity for Dijkstra **is O (|no. of Edges| + |no. of vertices| log|no. of vertices|)** i.e **O(|E|+|V|log|V|)**

**Note: This Worst case complexity is for Single Source shortest path not for All Pairs.**

**Our Project Plan**:

Since the project states clearly that the algorithm has to be run and it has to calculate, the shortest path based on "positive" weights only from one node to another.

**1.** We will take 5-6 input matrices ( graphs as matrices) and calculate the running time for both the algorithms. To be able to compare the running time of both these algorithms , we will run Dijkstra's Algorithm for "n" nodes. [As Dijkstra Algorithm calculates shortest path from single source only ].

a. The problem set for these algorithms will be for both sparse graphs and dense graphs. This being a pre-requisite of implementation project will also help to identify the output of the algorithms for varied types of Input data. So our aim here is to deduce the efficiency via running time for " Input Scenarios " scenarios using both these algorithms.

b. Calculate worst case asymptotic complexity for both these algorithms. This will be calculated for every problem size that we provide to the algorithm, so if we provide 5 input sizes for sparse graphs, then we have 5 complexities for each data set. This will also be deduced for Dense graphs.

**2.** Once we have the output for each problem set, the findings will be represented using a Vertices vs. Time graph, this graph will help us demonstrate and prove, which of the two algorithms is more efficient for a each type of input data set.

**3.** Finally, when we have run varied types of input data, of varied sizes, and plotted them on the graph and worst case asymptotic notation, we will be able to come to a conclusion that how Floyd-Warshall and Dijkstra behave and which one's better.