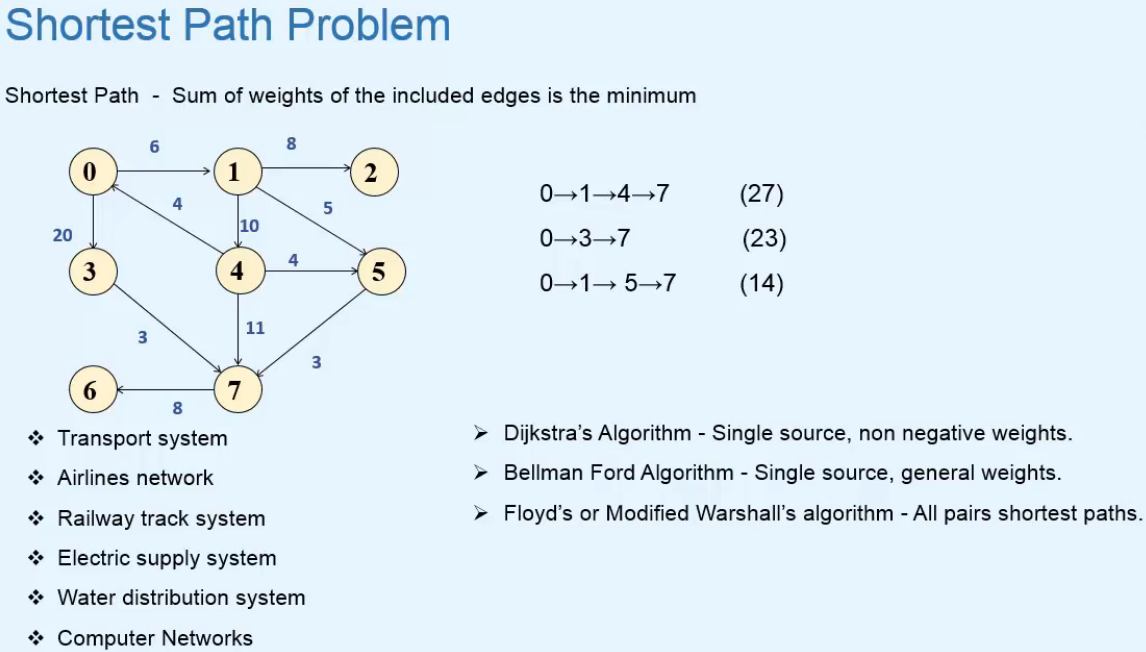
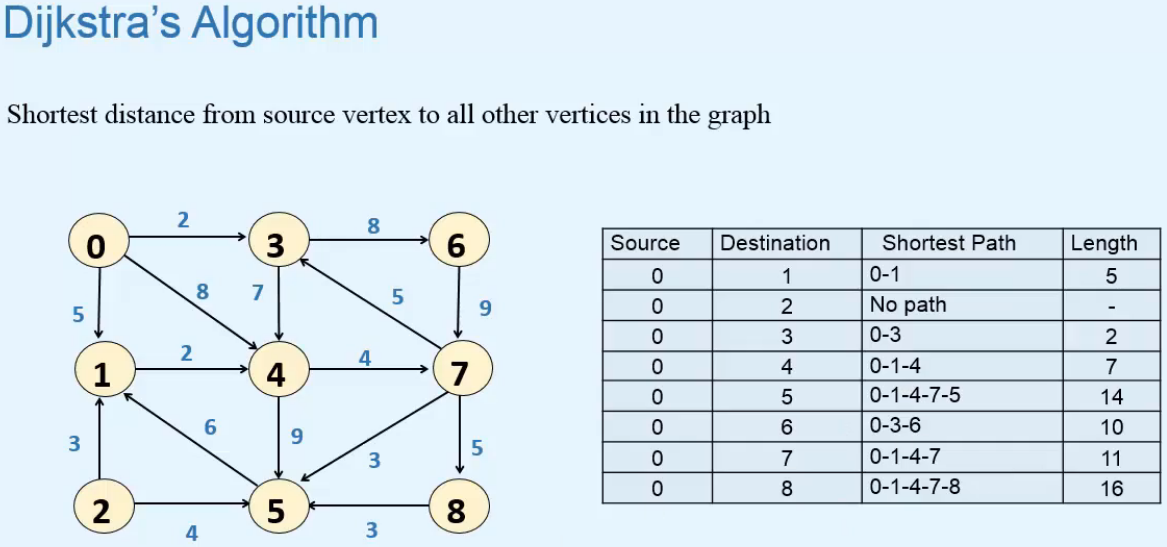
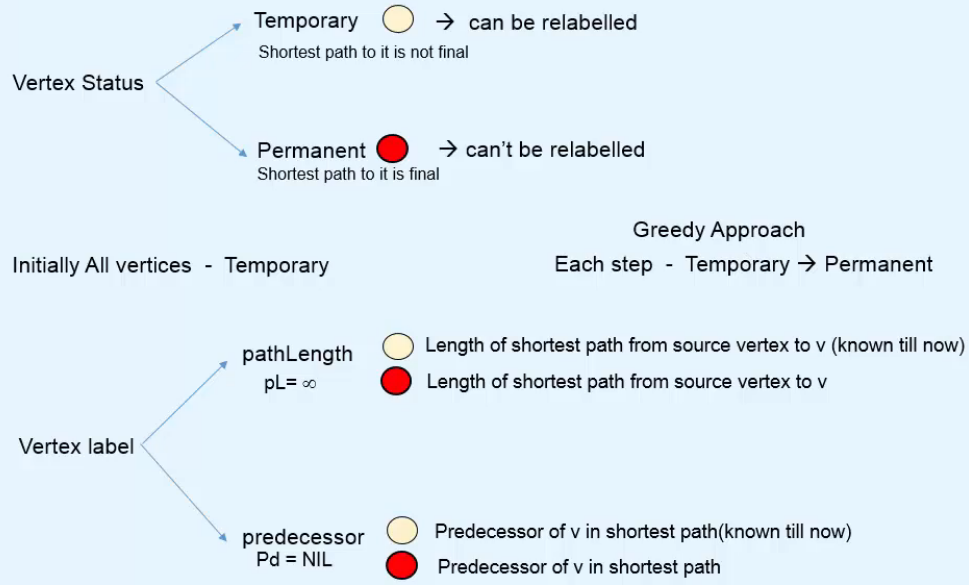
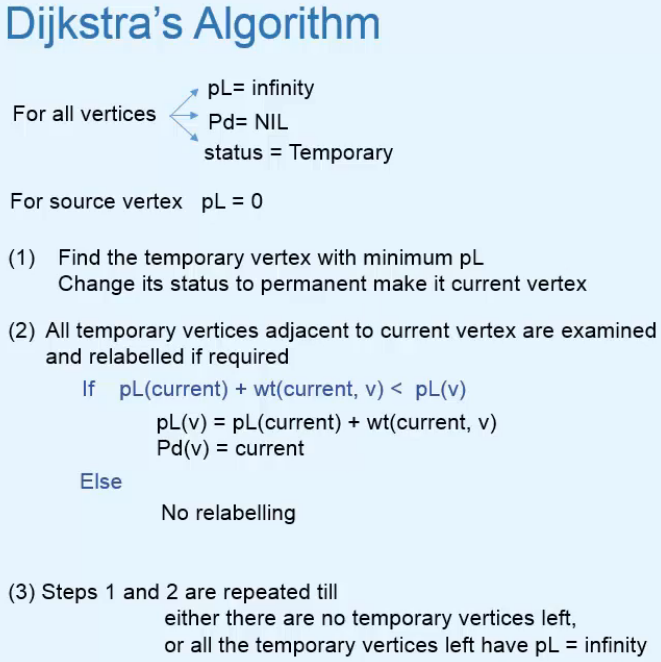
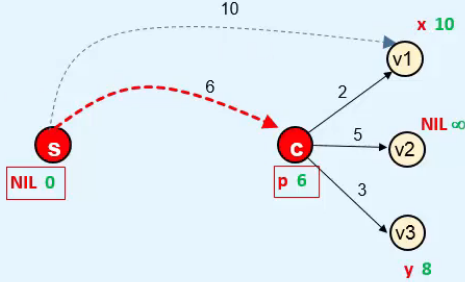
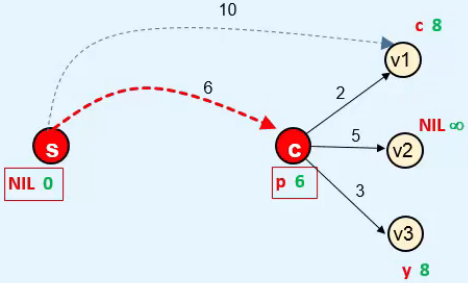
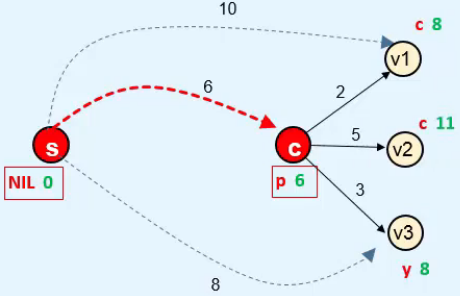
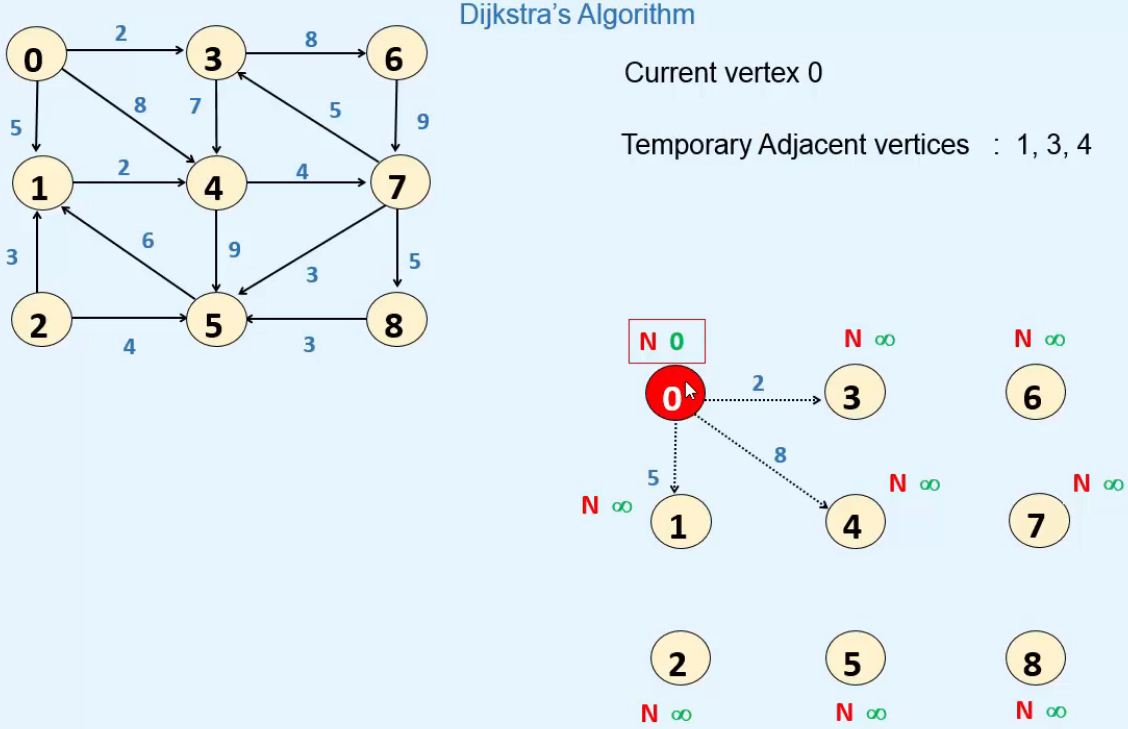
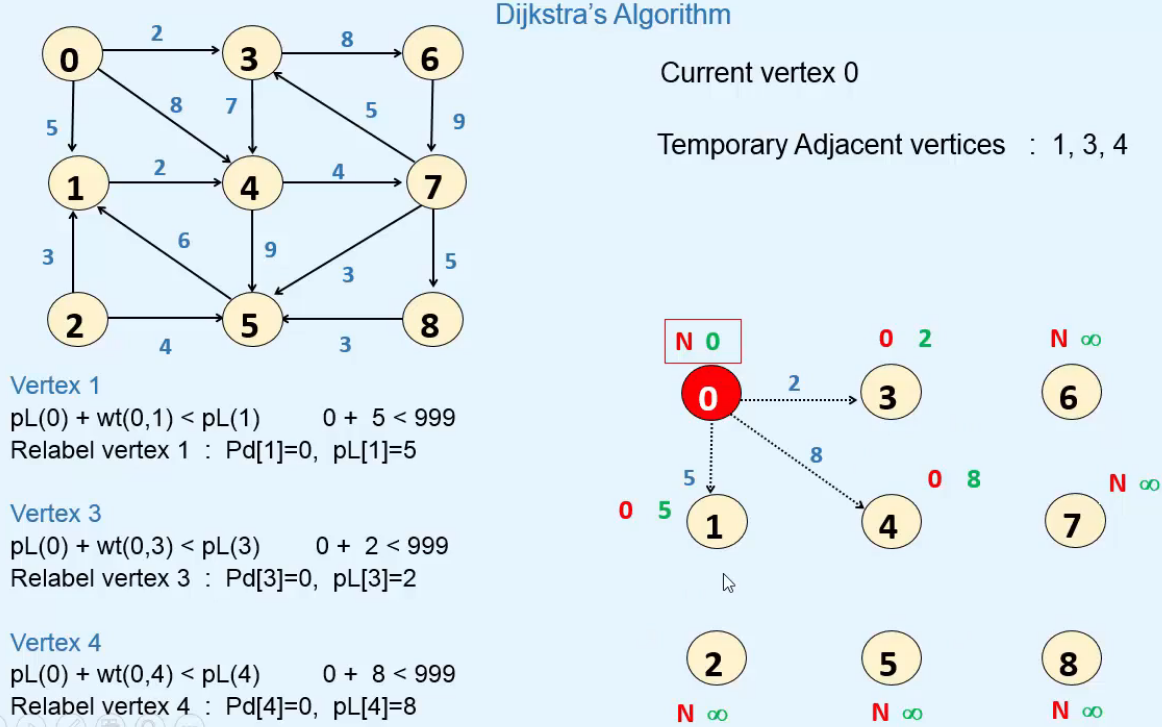
**Shortest Path Problem**  
\* **How to find Shortest Path between 2 vertices of a WEIGHTED Graph**.  
\* **In an unweighted graph, the shortest path between 2 vertices is the one that includes the minimum number of edges and the length is the number of edges included in the path**.  
\* **We’ve seen how we can use BFS to find shortest path in unweighted graphs**.  
  
\* **In a WEIGHTED Graph, the shortest path is the one in which the sum of weights of the included edges is the minimum**.  
\* **The Shortest Path may not be unique**.  
\* For example Transport System => the vertices represent cities and weights on the edges represent the distance between them  
\* Using the shortest path algorithm, we can find the shortest root for going from city A to city B.  
\* We’ve already seen   
**Single Source Shortest Path Algorithm** => **a vertex is selected as a source/start vertex and at the end of the algorithm, we get the shortest paths from source vertex to all of the vertices of the graph**.  
\* Floyd’s = Modified Warshall’l algorithm because it’s based on the Warshall’s algorithm for finding the Path Matrix.  
=> We get shortest paths between all pairs of vertices of the graph.

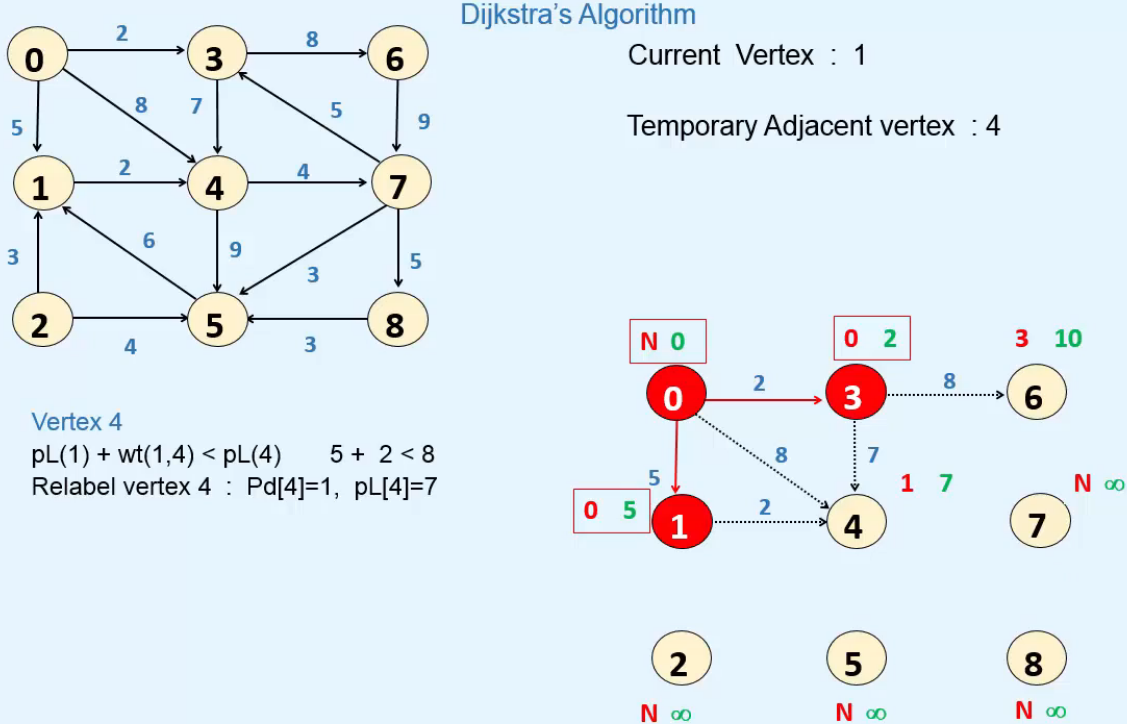
**Dijkstra’s Algorithm**  
  
  
=> **In each step, we’ll finalize the shortest path to a vertex of the graph**.  
\* During the algorithm, the pathLength and predecessor can be changed many times.  
**Greedy Approach** => **to choose a vertex that is to be made Permanent, we’ll use the Greedy Approach - we generally perform an action that appears best at the moment - taking the Temporary vertex with minimum pathLength and make it permanent**.  
  
=> **Suppose v is a Temporary vertex adjecent to current vertex that we’re examining**.  
=> **Temporary vertices with pL = infinite at the end, are not reachable from the source vertex**.  
  
=> pL(current) = 6, wt(current, v1) = 2 pL(v1) = 10  
=> pL(current) + wt(current, v1) < pL(v1)  
=> we change pL(v1) to 8 and Pd(v1) to current  


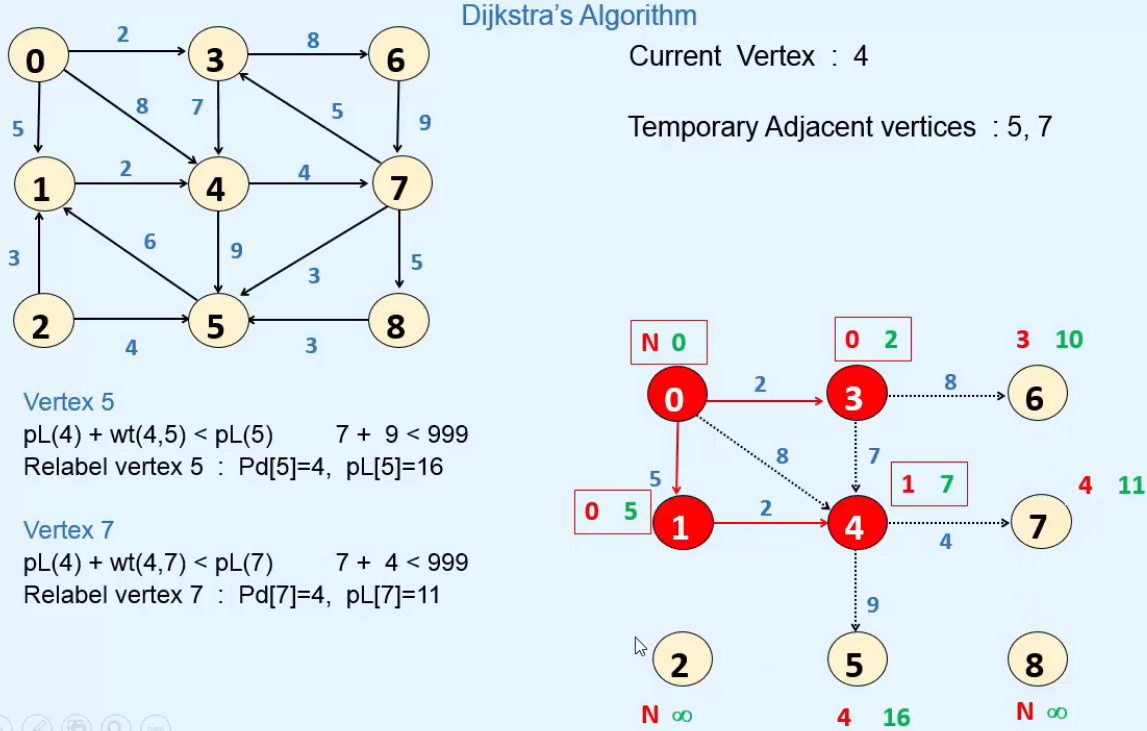
  
=> No relabelling for v3.

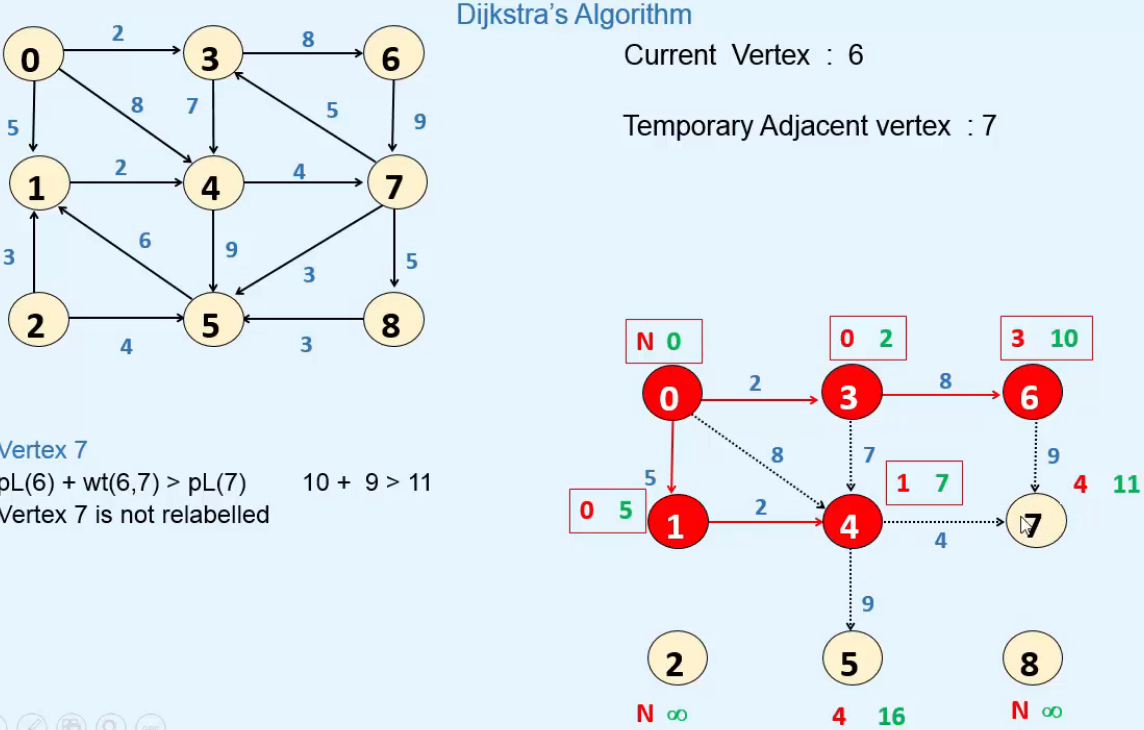
**Dijkstra’s Algorithm: Example**  
\* Let’s choose 0 as the start vertex.  


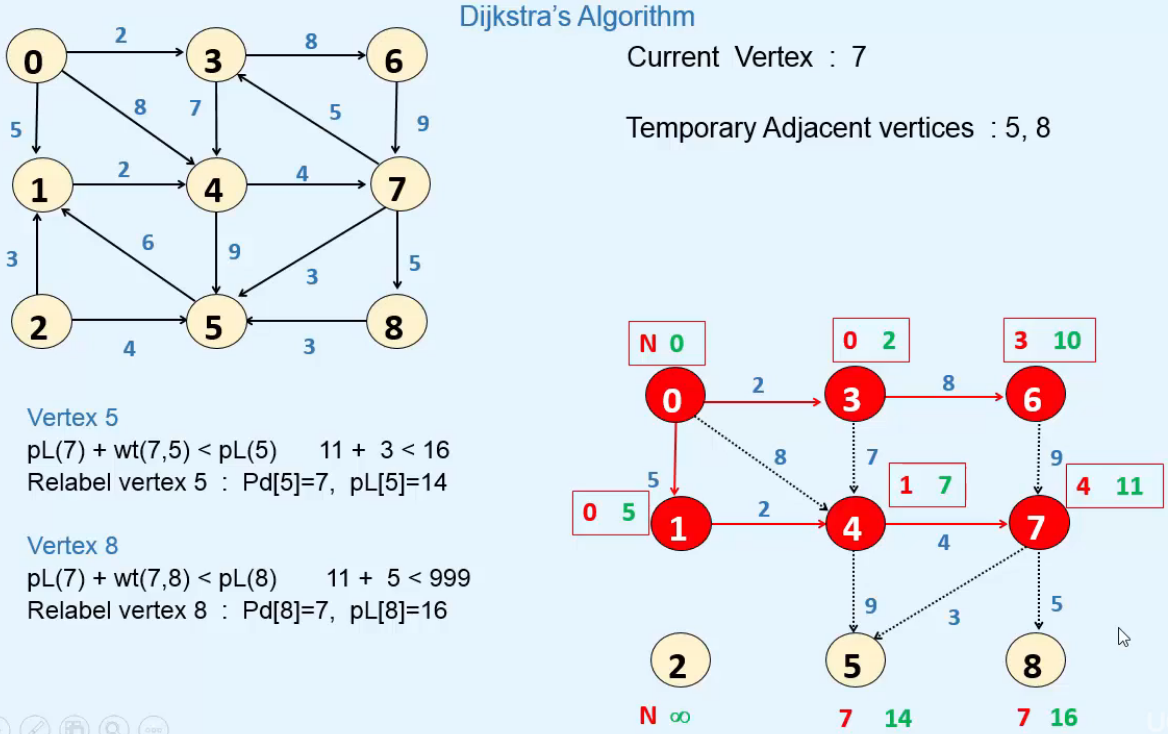


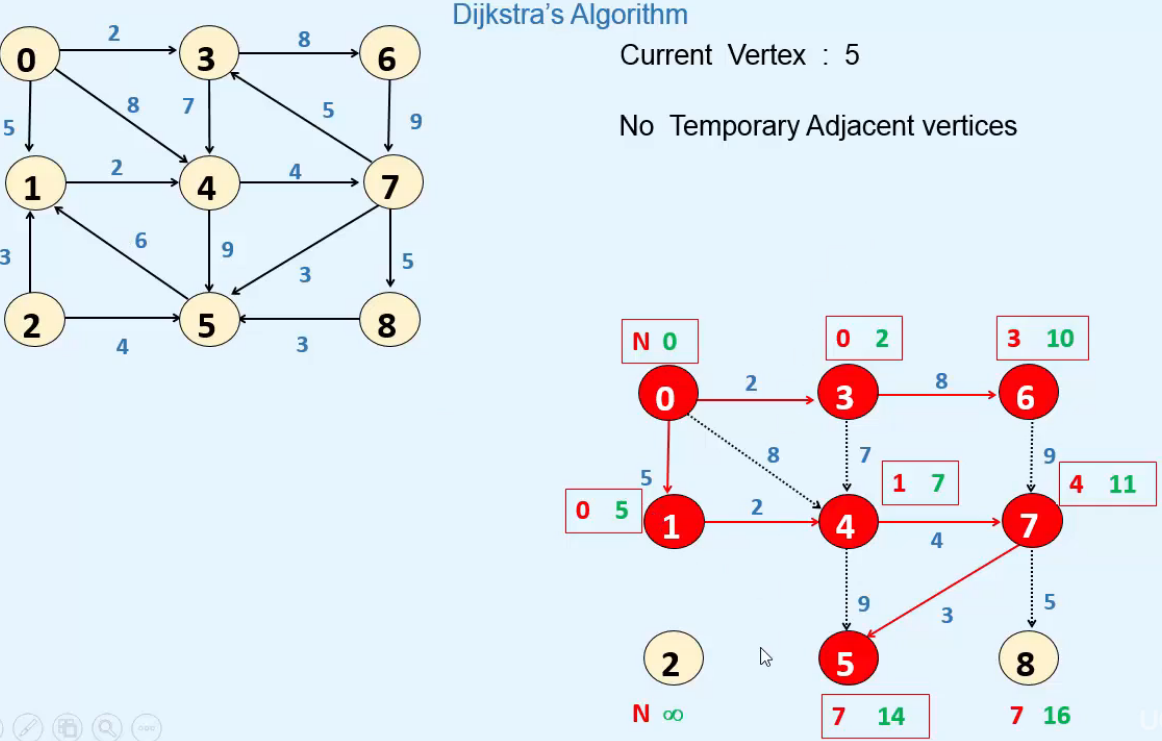
\* **Now from all the Temporary vertices, we have to choose the one which has the minimal pathLength   
=> vertex 3   
=> we make vertex 3 Permanent   
=> now 3 is the current vertex**  
**=>** **examine all the Temporary vertices that are adjacent to vertex 3**  

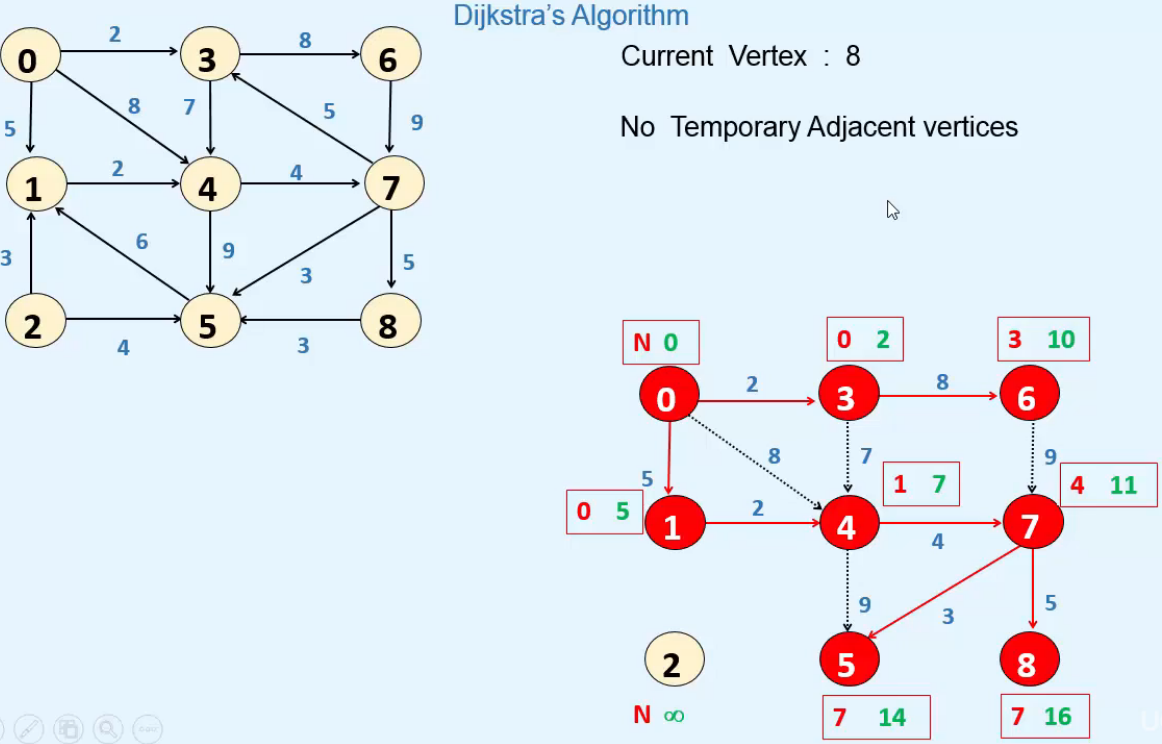







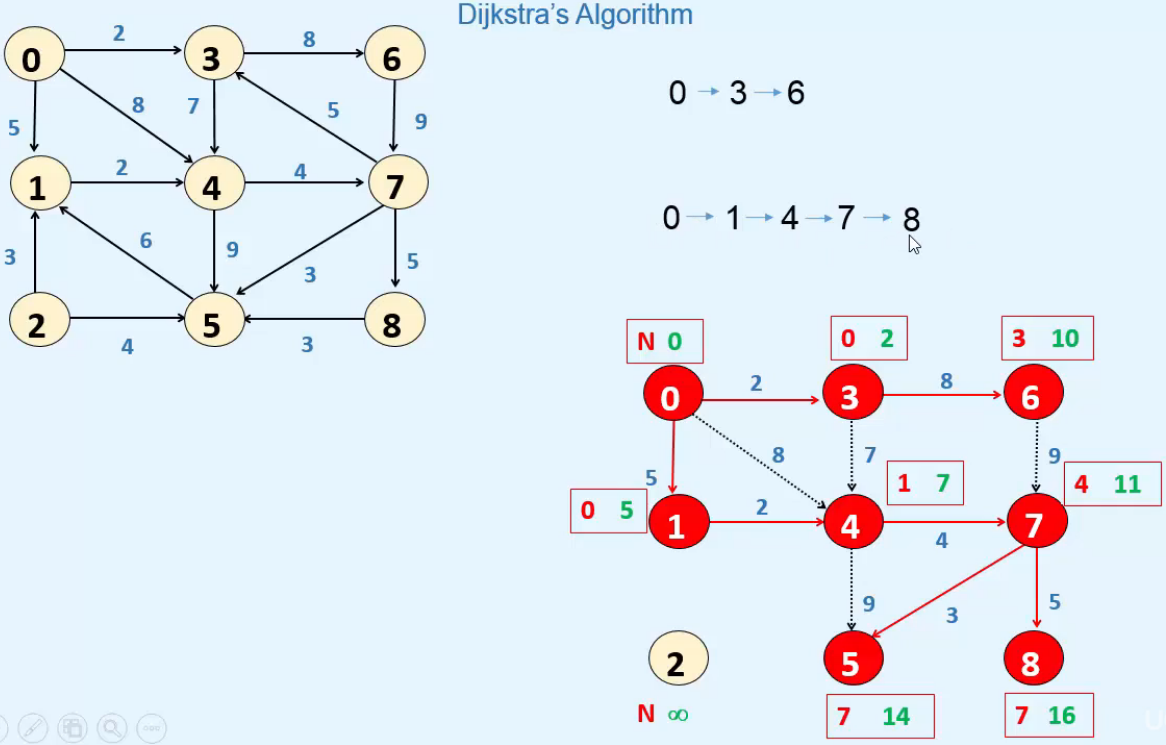






=> **Now there’s only 1 Temporary vertex left and it’s INFINITY => we stop**.

=> **Now we can easily find Shortest Path from vertex 0 to any other vertex by following the predecessors till we reach the source vertex**.



**Dijkstra’s Algorithm in Java**  
