Example: Step by Step operation of Dijkstra algorithm.

Algorithm Dijkstra

1. **INITIALIZE-SINGLE-SOURCE(G, s)**
2. \( S \leftarrow \emptyset \)
3. \( Q \leftarrow V[G] \)
4. **while** \( Q \neq \emptyset \) **do**
5. \( u \leftarrow \text{EXTRACT-MIN}(Q) \)
6. \( S \leftarrow S \cup \{u\} \)
7. **for** each vertex \( v \) in \( Q \) such that \( v \in \text{Adj}[u] \) **do**
8. \( \text{RELAX}(u, v, w) \)

Algorithm RELAX

1. **if** \( d[v] > d[u] + w(u, v) \) **then**
2. \( d[v] \leftarrow d[u] + w(u, v) \)
3. \( \pi[v] \leftarrow u \)

Algorithm INITIALIZE-SINGLE-SOURCE

1. **for** each vertex \( v \in V[G] \) **do**
2. \( d[v] \leftarrow \infty \)
3. \( \pi[v] \leftarrow \text{NIL} \)
4. \( d[s] \leftarrow 0 \)

---

**Step 1.** Given initial graph \( G = (V, E) \). All nodes have infinite cost except the source node, \( s \), which has 0 cost. We initialize \( d[s] \) to 0.

**Step 2.** We choose the node, which is closest to the source node, \( s \) (at the start, it is \( s \) itself). Add \( s \) to \( S \). Relax all nodes adjacent to \( s \). Update predecessor (see red arrow in diagram below) for all nodes updated.
Step 3. Choose the closest node, x. Relax all nodes adjacent to node x. Update predecessors for nodes u, v and y (again notice red arrows in diagram below).

Step 4. Now, node y is the closest node, so add it to S. Relax node v and adjust its predecessor (red arrows remember!).

Step 5. Now we have node u that is closest. Choose this node and adjust its neighbor node v.

Step 6. Finally, add node v. The predecessor list now defines the shortest path from each node to the source node, s.