DIJKSTRA's SSSP Algorithm.

INPUT: Directed, connected, weighted graph $G=(V,E)$ and a source vertex $s \in V$, where s can reach all vertices of V.

OUTPUT: $\forall v \in V$, $\delta(s,v)$ = Length of Shortest Path

INITIALIZE: $\forall v \in V: d(v) := \infty;$

$d(s) := 0;\ S := \phi;\ Q := V;$

While $Q \neq \phi$ do

- $u :=$ Extract-Min$[Q]$; $\delta(s,u) := d(u)$;
  DELETE u from Q;
  INSERT u in S;

  for each vertex v such that $(u,v) \in E$ do
  - If $d(u) + wt(u,v) < d(v)$
    then $d(v) := d(u) + wt(u,v)$

weight of edge uv.

Called the RELAX operation

Note that if $v \notin Q$, then the Relax operation will not decrease its $d(v)$ value as $d(v) = \delta(s,v)$ at this point (its in set S).

CORRECTNESS OF DIJKSTRA'S ALGORITHM

CLAIM: For every vertex v:

- At the moment when $d(v)$ is minimum in Q: $d(v) = \delta(s,v)$
- From that moment, $d(v)$ does not change anymore.

<u>C1</u>: $\forall v \in V$: $\delta(s,v) \leq d(v)$ at any moment during the algorithm.

<u>C2</u>: Assume at some moment, $d(v) = \delta(s,v)$. Then during the rest of the algorithm, $d(v)$ does not change.

<u>C3</u>: The minimum d-value in Q never decreases.

<u>C4</u>: Let $v \neq s$. Let $\pi(s,v)$ be shortest path from s to v. Let uv be last edge in the path.

$$s \to \circ \to \circ \to \circ \to \cdots \to u \to v$$

Consider the iteration in which u is chosen as the vertex in Q with minimum d-value.

If $d(u) = \delta(s,u)$ at the begining of this iteration, then $d(v) = \delta(s,v)$ at the end of this iteration.

Proof of C1: Either $d(v) = \infty$ or $d(v)$ is length of some path from s to v.

All paths from s to v have length at least $\delta(s,v)$.

Proof of C2: Algorithm (while-loop) only decreases $d(v)$ value,

but by C1 it is at least $\delta(s,v)$.

Proof of C3: Let $u \in Q$ with minimum $d(u)$ value in an iteration of the While-loop.

Before we execute the for-loop,

$d(u) \leq d(v)$ for all vertices in Q as $d(u)$ is minimum.

In the for-loop we may decrease some $d(v)$ values from its current value to
$d(v) = d(u) +$ weight of the edge $(u,v) \geq d(u)$.

But in any case $d(v) \geq d(u)$ for all $v \in Q$.

Proof of C4:
Let $\pi(s,v)$ be a shortest path.

$$s \to \circ \to \circ \to \circ \to \cdots \to \circ \to u \to v$$

Let (u,v) be the last edge on $\pi(s,v)$.

Consider the iteration when u is chosen to be the vertex from Q with minimum $d(u)$ value.

Since $\pi(s,v)$ is a shortest path,
$\delta(s,v) = \delta(s,u) + wt(u,v)$.

When we execute the for-loop with respect to $d(u)$, and if $d(u) = \delta(s,u)$, then
$$d(v) = \min\{d(v), \delta(s,u) + wt(u,v)\} \leq \delta(s,v).$$

But from C1 we know that $d(v) \geq \delta(s,v)$.

Thus $d(v) = \delta(s,v)$ at the end of this iteration.

Proof of MAIN CLAIM:

Recall the claim
For every vertex v:
- at the moment when $d(v)$ is minimum in Q,
  $d(v) = \delta(s,v)$
- from that moment, $d(v)$ doesn't change anymore.

First a few observations:

1. Subpaths of shortest paths are shortest paths
2. Shortest paths, in graphs where each edge has positive non-zero weight, has no loops (cycles).
3. Size of Q decreases in each iteration

Now the proof of the claim

If $v = s$ (the source vertex) claim is true as $d(s) = 0 = \delta(s,s)$ and $d(s)$ never changes after the first iteration

Suppose $v \neq s$ and consider a shortest path from s to v. Let this path goes through $k \geq 0$ intermediate vertices, $v_1, v_2, v_3, \ldots, v_k$.

$$s \to v_1 \to v_2 \to v_3 \to \cdots \to v_k \to v.$$

Note that since $s \to v_1 \to v_2 \to \cdots \to v_k \to v$ is a shortest path, any subpath $s \to v_1 \to v_2 \to \cdots \to v_i$, for $1 \leq i \leq k$, is a shortest path from s to v_i.

s :

Consider the (first) iteration of While-loop when s is chosen : $d(s) = 0 = \delta(s,s)$

From (C4) : At the end of the (first) iteration when s is removed from Q,

$$d(v_1) = \delta(s, v_1)$$

From (C2) : $d(v_1)$ doesn't change afterwards

Note : v_1 is still in Q. (WHY ?)

v_1

Now Consider the iteration when v_1 is chosen in Q to be deleted, i.e $d(v_1)$ is minimum ; then $d(v_1) = \delta(s, v_1)$. When v_1 is deleted, the for-loop in that iteration sets

From (C4) : $d(v_2) = \delta(s, v_2)$.

From (C3) : $d(v_2) > d(v_1)$; thus $v_2 \in Q$

From (C2) : $d(v_2)$ doesn't change any more.

v_2

Consider the iteration when v_2 is chosen as the vertex in Q to be deleted, i.e $d(v_2)$ is minimum ($d(v_2) = \delta(s, v_2)$)

At the end of this iteration.

From C4 : $d(v_3) = \delta(s, v_3)$

From C2 : $d(v_3)$ doesn't change afterwards

From C3 : Since, $d(v_3) > d(v_2)$, $v_3 \in Q$.

$\vdots \qquad \vdots$

v_K

Consider the iteration when v_K is chosen as the vertex in Q to be deleted, i.e $d(v_K)$ is minimum

– $d(v_K) = \delta(s, v_K)$

At the end of this iteration

From C4 : $d(v) = \delta(s, v)$

From C2 : $d(v)$ doesn't change anymore

From C3 : $d(v) > d(v_K)$ implying that $v \in Q$.

v :

Consider the iteration when v is deleted from Q.

(i.e $d(v)$ is minimum).

$d(v) = \delta(s, v)$.

By C2 : $d(v)$ doesn't change afterwards. ∎