5.6

(a) Let $D_i$ be the shortest distance from node $i$ to node 1 corresponding to lengths $d_{ij}$. We claim that

$$D_i \leq D_i', \quad \forall i.$$ 

Given the definition of $D_i'$, it will suffice to show that

$$D_i \leq D_i, \quad \forall i \notin \cup_k N_k \cup \{1\}. $$

Indeed consider any node $i \notin \cup_k N_k \cup \{1\}$, and let $P_i$ be a shortest path from $i$ to 1 corresponding to lengths $d_{ij}$. We have $D_m = d_{mn} = D_n$ for all arcs $(m,n)$ of $P_i$, so by the definition of the sets $N_k$, we must have $d_{mn} \leq d_{mn}$ for all arcs $(m,n)$ of $P_i$. Therefore, the length of $P_i$ with respect to arc lengths $d_{ij}$ is no more than its length with respect to arc lengths $d_{ij}$, implying that $D_i \leq D_i$. Thus we have $D_i \leq D_i$ for all $i \notin \cup_k N_k \cup \{1\}$ and $D_i \leq D_i'$ for all $i$.

Now consider the Bellman-Ford method corresponding to arc lengths $d_{ij}$ and starting from two different initial conditions. The first set of initial conditions is the standard $D_i^0 = 0$ for all $i \neq 1$ and $D_1^0 = 0$, and the corresponding iterates are denoted $D_i^k$. The second set of initial conditions is $D_i^0$ as given in the problem statement and the corresponding iterates are denoted $D_i^h$. Since

$$D_i \leq D_i^0 \leq D_i^h, \quad \forall i,$$

we can show by induction that

$$D_i^{k+1} = \min_j [d_{ij} + D_j^k],$$

$$D_i^{h+1} = \min_j [d_{ij} - D_j^h],$$

$$D_i = \min_j [d_{ij} + D_j],$$

that

$$D_i \leq D_i^k \leq D_i^h, \quad \forall i, h.$$ 

Since $D_i^h = D_i$ for $h \geq N - 1$, it follows that $D_i^h = D_i$ for $h \geq N - 1$, proving the desired result.

(b) As stated in the hint, when the length of a link $(i,j)$ on the current shortest path tree increases, the head node $i$ of the link should send an estimated distance $D_i = \infty$ to all nodes $m$ such that $(m,i)$ is a link. These nodes should send $D_m = \infty$ to their upstream neighbors if $i$ is their best neighbor, that is, if link $(m,i)$ lies on the shortest path tree, etc. Before any of the nodes $m$ that sent $D_m = \infty$ to its upstream neighbors recalculates its estimated shortest distance, it should wait for a sufficient amount of time to receive from its downstream neighbors any updated distances $D_n = \infty$ that may have resulted from the transmission of $D_i = \infty$.

5.7

Using the hint, we begin by showing that $h_i > h_j$ for all $i \neq 1$. Proof by contradiction. Suppose that there exists some $i \neq 1$ for which $h_i \leq h_j$. From the Bellman-Ford algorithm we have $D_i^{(h_i-1)} \geq D_i^{(h_j)}$. We define $h_i = 0$ for completeness. Therefore, $D_i^{(h_i-1)} \geq D_i^{(h_j)}$. However, if this held with equality it would contradict the definition of $h_j$, as the largest $h$ such that $D_i^{(h)} = D_i^{(h-1)}$.

Therefore, $D_i^{(h_i-1)} > D_i^{(h_j)}$. Using this strict inequality in the definition of $h$, $D_i^{(h_i)} = D_i^{(h_i-1)} + c_{ij}$, gives $D_i^{(h)} > D_i^{(h-1)} + c_{ij}$. From the Bellman-Ford algorithm, we know that $D_1^{(h-1)} \leq D_i^{(h-1)} = D_i^{(h-1)}$. Using this in the previous expression gives $D_i^{(h)} > D_i^{(h-1)}$ which contradicts the definition of $h$, as the largest $h$ such that $D_i^{(h)} = D_i^{(h-1)}$. Therefore, the supposition that $h_i \leq h_j$, is incorrect. This proves the claim.

The subgraph mentioned in the problem contains $N - 1$ arcs. To show that it is a spanning tree, we must show that it connects every node to node 1. To see this label each node $i$ with $h_i$. Since the Bellman-Ford algorithm converges in at most $N - 1$ iterations, we have $0 < h_i \leq N - 1$ for all $i \neq 1$. Furthermore, $h_i = 0$ and $h_i > h_j$, for all $i \neq 1$. Each node $i \neq 1$ is connected to a neighbor with a smaller label. We can trace a path in the subgraph from every node to node 1, therefore the subgraph must be a spanning tree.

Since the path lengths $D_i^{(h_i)}$ along the subgraph satisfy Bellman's equation, the spanning tree is a shortest path spanning tree rooted at node 1.
5.11

(a) We have $D_1 = 0$ throughout the algorithm because initially $D_1 = 0$, and by the rules of the algorithm, $D_1$ cannot change.

We prove property (1) by induction on the iteration count. Indeed, initially (1) holds, since node 1 is the only node $j$ with $D_j < \infty$. Suppose that (1) holds at the start of some iteration in which a node $i$ is removed from $V$. If $i = 1$, which happens only at the first iteration, then at the end of the iteration we have $D_j = d_{ij}$ for all inward neighbors $j$ of 1, and $D_j = \infty$ for all other $j \neq 1$, so $D_j$ has the required property. If $j \neq 1$, then $D_j < \infty$ (which is true for all nodes of $V$ by the rules of the algorithm), and (by the induction hypothesis) $D_j$ is the length of some walk $P_j$ starting at $j$, ending at 1, without going twice through 1. When $D_j$ changes as a result of the iteration, $D_j$ is set to $d_{ij} + D_i$, which is the length of the walk $P_j$ consisting of $P_i$ preceded by arc $(i, j)$. Since $i \neq 1$, $P_i$ does not go twice through 1. This completes the induction proof of property (1).

To prove property (2), note that for any $j$, each time $j$ is removed from $V$, the condition $D_j \leq d_{ij} + D_i$ is satisfied for all $(i, j) \in A$ by the rules of the algorithm. Up to the next entrance of $j$ into $V$, $D_j$ stays constant, while the labels $D_i$ for all $i$ with $(i, j) \in A$ cannot increase, thereby preserving the condition $D_j \leq d_{ij} + D_i$.

(b) We first introduce the sets

$$I = \{ i \mid D_i < \infty \text{ upon termination} \},$$

$$\overline{I} = \{ i \mid d_i = \infty \text{ upon termination} \},$$

and we show that we have $D_j \in \overline{I}$ if and only if there is no walk to 1 from $j$. Indeed, if $i \not\in I$, then, since $i \not\in V$ upon termination, it follows from condition (2) of part (a) that $j \not\in I$ for all $(j, i) \in A$. Therefore, if $j \not\in I$, there is no walk from node $j$ to any node of $I$ (and in particular, node 1). Conversely, if there is no walk from $j$ to 1, it follows from condition (1) of part (a) that we cannot have $D_j < \infty$ upon termination, so $j \not\in I$.

We show now that for all $j \in I$, we have $d_j = \min_{(j, i) \in A} \{ d_{ji} + D_i \}$ upon termination. Indeed, conditions (1) and (2) of part (a) imply that upon termination we have, for all $i \in I$,

$$D_j \leq d_{ji} + D_i, \quad \forall j \text{ such that } (j, i) \in A$$

while $D_i$ is the length of some walk $P_i$ from $i$ to 1. Fix a node $m \in I$. By adding this condition over the arcs $(j, i)$ of any walk $P$ from $m$ to 1, we see that the length of $P$ is no less than $D_m$. Hence $P_m$ is a shortest walk from $m$ to 1. Furthermore, the equality $D_j = d_{ji} + D_i$ must hold for all arcs $(j, i)$ on the shortest walks $P_m, m \in I$, implying that $D_j = \min_{(j, i) \in A} \{ d_{ji} + D_i \}$.

(c) If the algorithm never terminates, some $D_j$ must decrease strictly an infinite number of times, generating a corresponding sequence of distinct walks $P_j$ as per condition (1) of part (b). Each of these walks can be decomposed into a path from $j$ to 1 plus a collection of cycles. Since the number of paths from $j$ to 1 is finite, and the length of the walk $P_j$ is monotonically decreasing, it follows that $P_j$ eventually must involve a cycle with negative length. By replicating this cycle a sufficiently large number of times, one can obtain walks from $j$ to 1 with arbitrarily small length.

(d) Clear from the statement of Dijkstra's algorithm.