1. Let $G$ be an arbitrary weighted, directed graph with a negative-weight cycle reachable from the source vertex $s$. Show that an infinite sequence of relaxations of the edges of $G$ can always be constructed such that every relaxation causes a shortest-path estimate to change.

2. Suppose that a weighted, directed graph $G = (V, E)$ has a negative-weight cycle. Give an efficient algorithm to list the vertices of one such cycle.

3. Give a simple example of a directed graph with negative-weight edges for which Dijkstra’s algorithm produces incorrect answers. What does not work?

4. Give an efficient algorithm to count the total number of paths in a DAG.

5. Professor Gaedal has written a program that he claims implements Dijkstra’s algorithm. The program produces $v.d$ and $v.\pi$ for every vertex $x \in V$. Give an $O(|V| + |E|)$-time algorithm to check the output of the professor’s program. It should check whether the $d$ and $\pi$ attributes match those of some shortest-path tree.

6. Let $G = (V, E)$ be a weighted, directed graph with weight function $w : E \to \{0, 1, \ldots, W - 1\}$ for some non-negative integer $W$.

   (a) Modify Dijkstra’s algorithm to compute the shortest paths from a given source vertex $s$ in $O(W \cdot |V| + |E|)$ time.

   (b) Modify your algorithm to run in $O((|V| + |E|) \cdot \log W)$ time. (Hint: How many distinct non-final shortest-path estimates can there be at any point in time?)

7. (Gabow’s scaling algorithm for single-source shortest paths) We are given a directed graph $G = (V, E)$ with non-negative integer edge weights $w$. Let $W = \max_{(u,v) \in E} w(u,v)$. Our goal is to develop an algorithm that runs in $O(|E| \log W)$ time. The algorithm uncovers the bits in the binary representation of the edge weights one at a time, from the most significant bit to the least significant bit. Specifically, let $k = \lceil \log(W + 1) \rceil$ be the number of bits in the binary representation of $W$, and for $i = 1, 2, \ldots, k$, let $w_i(u,v) = \lfloor w(u,v)/2^{k-i} \rfloor$. Let us define $\delta_i(u,v)$ as the shortest-path weight from vertex $u$ to vertex $v$ using weight function $w_i$. Thus, $\delta_k(u,v) = \delta(u,v)$ for all $u, v \in V$. For a given source vertex $s$, the scaling algorithm first computes the shortest-path weights $\delta_1(s,v)$ for all $v \in V$, then computes $\delta_2(s,v)$ for all $v \in V$, and so on, until it computes $\delta_k(s,v)$ for all $v \in V$. We assume throughout that $|E| \geq |V| - 1$, and we shall see that computing $\delta_i$ from $\delta_{i-1}$ takes $O(|E|)$ time, so that the entire algorithm takes $O(k|E|) = O(|E| \log W)$ time.

   (a) Suppose that for all vertices $v \in V$, we have $\delta(s,v) \leq |E|$. Show that we can compute $\delta(s,v)$ for all $v \in V$ in $O(|E|)$ time.

   (b) Show that we can compute $\delta_i(s,v)$ for all $v \in V$ in $O(|E|)$ time. Let us now concentrate on computing $\delta_i$ from $\delta_{i-1}$. 

(c) Prove that for \( i = 2, 3, \ldots, k \), either \( w_i(u, v) = 2w_{i-1}(u, v) \) or \( w_i(u, v) = 2w_{i-1}(u, v) + 1 \). Then, prove that \( 2\delta_{i-1}(s, v) \leq \delta_i(s, v) \leq 2\delta_{i-1}(s, v) + |V| - 1 \) for all \( v \in V \).

(d) Define for \( i = 2, 3, \ldots, k \) and all \((u, v) \in E\), \( \hat{w}_i(u, v) = w_i(u, v) + 2\delta_{i-1}(s, u) - 2\delta_{i-1}(s, v) \). Prove that for \( i = 2, 3, \ldots, k \) and all \( u, v \in V \), the “rewighted” value \( \hat{w}_i(u, v) \) of an edge \((u, v)\) is a non-negative integer.

(e) Now, define \( \hat{\delta}_i(s, v) \) as the shortest-path weight from \( s \) to \( v \) using the weight function \( \hat{w}_i \). Prove that for \( i = 2, 3, \ldots, k \) and all \( v \in V \), \( \delta_i(s, v) = \hat{\delta}_i(s, v) + 2\delta_{i-1}(s, v) \) and that \( \hat{\delta}_i(s, v) \leq |E| \).

(f) Show how to compute \( \delta_i(s, v) \) from \( \delta_{i-1}(s, v) \) for all \( v \in V \) in \( O(|E|) \) time, and conclude that \( \delta(s, v) \) can be computed for all \( v \in V \) in \( O(|E| \log W) \) time.