After moving to a new city, you decide to choose a walking route from your new office to home.

(a) Suppose we allow some or all vertex heights to be equal. Describe and analyze an algorithm to find the shortest downhill walk from \( s \) to \( t \); you may use flat edges in your walk, but you should never use an edge that goes uphill.

**Solution:** We start by building a new edge-weighted directed graph \( G' \) with the same vertex set as \( G \). We add a directed edge \( u \to v \) to \( G' \) for each undirected edge \( uv \) such that \( h(u) \geq h(v) \) and give it a weight of \( w(u \to v) = l(uv) \). Observe that we add both \( u \to v \) and \( v \to u \) to \( G' \) if \( h(u) = h(v) \).

Any path in \( G' \) corresponds directly to a path in \( G \) that only travels along flat or downhill streets. The total weight of the path in \( G' \) is equal to the length of the path in \( G \). Therefore, we want the shortest path in \( G' \) from \( s \) to \( t \). All edge weights are non-negative lengths, so we run Dijkstra’s algorithm with source \( s \) in \( O(E \log V) \) time.

**Rubric:** 5 points total: 3 points for the algorithm. 1 point for the proof of correctness. 1 point for running time analysis.

(b) Now suppose we assume all vertex heights are distinct; i.e., no two heights are equal. Describe and analyze an algorithm to find the shortest downhill walk from \( s \) to \( t \).

**Solution:** We build a new edge-weighted directed graph \( G' \) exactly as described for part (a). The distinct vertex heights implies \( G' \) is a directed acyclic graph; any directed cycle would have to include an edge going uphill to the highest vertex in the cycle, and uphill edges don’t exist from the way we constructed \( G' \). We compute the shortest path in \( G' \) from \( s \) to \( t \) using the \( O(V + E) \) time shortest paths algorithm for directed acyclic graphs.

**Rubric:** 5 points total: 3 points for the algorithm. 1 point for the proof of correctness. 1 point for running time analysis.

A correct \( \Omega(E \log V) \) time algorithm is worth at most 3 points.
Let $G = (V, E)$ be a directed graph with edge weights $w : E \to \mathbb{R}$; edge weights can be positive, negative, or zero, but there are no negative weight cycles.

(a) Describe and analyze an algorithm that reports whether or not there is a cycle of length zero in $G$.

**Solution:** We first run Floyd-Warshall to compute the all pairs shortest paths distance matrix $dist$. Now, consider any edge $u \to v$ in $G$. We claim there is a length zero cycle through $u \to v$ if and only if $dist(v, u) = -w(u \to v)$. If $dist(v, u) = -w(u \to v)$, then $u \to v$ followed by a shortest path from $v$ to $u$ is a cycle with length $w(u \to v) + dist(v, u) = 0$. If there is a zero length cycle through $u \to v$, then there is a path from $v$ to $u$ with length exactly $-w(u \to v)$. There cannot be a shorter path, though, or the cycle of $u \to v$ followed by the shorter path would have negative length. Therefore, $dist(v, u) = -w(u \to v)$.

Based on the above observation, and the fact that a zero length cycle would need to contain at least one edge, we loop over all edges $u \to v$, checking if $dist(v, u) = -w(u \to v)$ for any of them. If the check passes once, we return `True`. Otherwise, there is no zero length cycle through any edge and we can return `False`.

Running Floyd-Warshall takes $O(V^3)$ time. Looping over all the edges takes $O(E) = O(V^2)$ time. **The total running time is $O(V^3)$**.

**Rubric:** 6 points total: 3 points for the algorithm. 2 points for the proof of correctness. 1 point for running time analysis.