1. Let $G = (V, E)$ be a directed graph in which each vertex $v$ is labeled with a unique integer $L[v]$. For each $v \in V$, let $R(v)$ denote the set of vertices reachable from $v$, and let $M(v) = \min\{L[w] : w \in R(v)\}$. Design and analyze an algorithm that runs in time $O(|V| + |E|)$ and computes $M(v)$ for all vertices $v$. As usual, assume that $G$ is represented using adjacency lists. Hint: solve the problem first for DAGs, and then generalize to arbitrary graphs using strongly connected components.

2. Design and analyze an $O(n^2)$-time algorithm that finds the longest monotonically increasing subsequence of a sequence of $n$ numbers.

3. You are given a directed graph $G = (V, E)$, where each edge $(u, v) \in E$ has a label $L(u, v)$. The labels are, say, integer values, but need not be unique. You are also given a “starting vertex” $v_0 \in V$, and a sequence $\sigma_1, \ldots, \sigma_k$ of labels. Design and analyze an efficient algorithm that finds a path $(v_0, v_1, \ldots, v_\ell)$ in $G$, which starts at $v_0$, and that is as long as possible, such that $L(v_{i-1}, v_i) = \sigma_i$ for $i = 1 \ldots \ell$, where $\ell \leq k$, of course. Your algorithm should run in time $O(|V| + |E| \cdot k)$. 