What is the fastest, cheapest and most efficient way to get from point A to point B? That consumes him, and all parts of his life.
– Sean Stanton on Travis Kalanick, Uber CEO

1. Suppose that we are doing Dijkstra’s Algorithm on vertex set \( V = \{1, \ldots, 500\} \) with source vertex \( s = 1 \) and at some time we have \( S = \{1, \ldots, 100\} \). What is the interpretation of \( \pi[v], d[v] \) for \( v \in S \)? What is the interpretation of \( \pi[v], d[v] \) for \( v \notin S \)? Which \( v \) will have \( \pi[v] = \text{NIL} \) at this time. For those \( v \) what will be \( d[v] \)?

2. Suppose, as with Dijkstra’s Algorithm, the input is a directed graph, \( G \), a source vertex \( s \), and a weight function \( w \). But now further assume that the weight function only takes on the values one and two. Modify Dijkstra’s algorithm – replacing the \textsc{Min-Heap} with a more suitable data structure – so that the total time is \( O(E + V) \).

3. Let \( G \) be a DAG on vertices \( 1, \ldots, n \) and suppose we are given that the ordering \( 1 \cdots n \) is a Topological Sort. Let \( \text{COUNT}[i,j] \) denote the number of paths from \( i \) to \( j \). Let \( s \), a “source vertex” be given. Give an efficient algorithm (appropriately modifying the methods of the chapter) to find \( \text{COUNT}[s,j] \) for all \( j \).

A clever man commits no minor blunders. – Goethe