1. Let $G$ be an undirected weighted graph with positive edge weights. Suppose Dijkstra’s algorithm is run on $G$ outputting a shortest path tree $T$. Prove or disprove:

(a) The shortest path tree $T$ is a spanning tree of $G$.
(b) The shortest path tree $T$ is a minimum spanning tree of $G$.

2. Let there be a program which claims to implement Dijkstra’s algorithm. The program produces $v.d$ and $v.\pi$ for each vertex $v \in V$. Give an $O(V + E)$-time algorithm to check the output of the program. It should determine whether the $d$ and $\pi$ attributes match those of some shortest-path tree. You may assume that all edge weights are nonnegative.

3. For this question, all graphs are unweighted, undirected, and connected with no parallel edges nor self-loops (i.e., simple graphs). Also, every path is simple, i.e., contains no repeated vertices. Using say breadth-first search we can determine the shortest path distance between two distinct vertices $s$ and $t$ in a graph $G$. But just how many shortest paths can there be between $s$ and $t$, and how does this affect possible algorithms for related problems? In particular, consider the following.

(a) Show that there is an infinite number of graphs with the following properties:
   - there is exactly one shortest path between $s$ and $t$,
   - when the shortest path distance between $s$ and $t$ is $d$, there are no two vertices $s'$ and $t'$ distinct from $s$ and $t$, respectively, such that $d(s', t') > d$, and
   - the graph is not a simple path $\circ \rightarrow \circ \rightarrow \cdots \rightarrow \circ$ itself.

(b) Show that there is an infinite number of graphs with the following properties:
   - there is at least one pair of distinct vertices $s$ and $t$ such that there is exactly one shortest path between $s$ and $t$, and
   - the number of simple paths between $s$ and $t$ is at least 16.

(c) Show that there is an infinite number of graphs with the following property:
   - the number of shortest paths between $s$ and $t$ is exponential in $V$, i.e., the number of vertices. (Hint: for example, come up with an $n$-vertex graph such that there are $2^{\Theta(n)}$ shortest paths between two vertices $s$ and $t$).
4.

Based on the previous problem, there are graphs that can have many (shortest) paths between two vertices. Despite this fact, there are efficient algorithms for finding a (shortest) path. Do you find this surprising?

(a) Can you name another problem with similar properties, that is, the number of possible solutions is large (i.e., exponential or non-polynomial in the input size) yet there is a polynomial-time algorithm for the problem?

(b) Finally, consider the problem of enumerating (e.g., listing all) shortest paths between two given vertices $s$ and $t$ in a graph. Is there an efficient, i.e., a polynomial-time algorithm for the problem?