Exercises - Single Source Shortest Paths - Algoritmiek Tutorial

1. **Explorer:** Consider a graph that contains m paths P_i for $1 \leq i \leq m$, each with infinite length, and has a vertex s connecting all P_i's by being adjacent to the first vertex of each P_i. More precisely, s is adjacent to vertex $v_{i,1}$ for $1 \leq i \leq m$ and vertex $v_{i,j}$ is adjacent to $v_{i,j+1}$ for $1 \leq i \leq m$ and $j \geq 1$. Someone left a treasure in one of the vertices in the graph. Our goal is to design an algorithm that starts from vertex s and is able to find the treasure. Keep in mind that there are infinite number of vertices and any algorithm cannot reach the end of path P_i for any i.

   (a) Does DFS necessarily find the treasure? Why?

   (b) Does BFS necessarily find the treasure? Why?

2. **Subpaths of shortest paths are shortest paths:** Consider any directed weighted graph $G = (V, E)$ which has no negative cycle. Prove that the subpaths of shortest paths are also shortest paths. More formally, consider any shortest path from v_0 to v_k, $v_0 \to v_1 \to \cdots \to v_k$, for any $0 \leq i \leq j \leq k$, the subpath $v_i \to v_{i+1} \to \cdots \to v_j$ is a shortest path from v_i to v_j.

3. **Shortest path subgraph:** Show that the predecessor subgraph of $G = (V, E)$ after any single source shortest path algorithm is a tree. (Note: the predecessor subgraph G_π is defined as (V, E_π), where $E_\pi = \{(\pi[v], v) \mid v \in V\}$.)

4. Prove the correctness of DIJKSTRA's algorithm formally.

5. **Path counting:**

(a) Consider a DAG (directed acyclic graph) $G = (V, E)$ and two vertices $s, t \in V$, give an $O(|V| + |E|)$-time algorithm to count the total number of paths from s to t. Analyze your algorithm and give the correctness proof.

(b) Consider an arbitrary directed graph $G = (V, E)$ with weighted edges, where all the weights are non-negative. Given two vertices $s, t \in V$, determine the number of shortest paths from a source vertex s to the target vertex t. This algorithm should run in the same time as Dijkstra's algorithm.

6. Consider a directed graph $G = (V, E)$ with non-negative edge lengths and a starting vertex $s \in V$. The *bottleneck* of a path is defined as the minimum length of one of its edges. Show how to compute correctly, for each vertex $v \in V$, the maximum bottleneck of any $s - v$ path. Your algorithm should run in $O(|V|^2)$ time.