Problem 1

Show that a forest has \( n - c \) edges, where \( c \) is the number of components (subtrees) in the forest.

Problem 2

Show that a tree \( T \) has at least \( \Delta(T) \) leaves, where

\[
\Delta(T) = \max_{v \in V(T)} d(v),
\]

i.e. the maximum degree of \( T \). (Similar to 5.1.5 p 158.)

Problem 3

Show that for every connected graph \( G \) and for every edge \( e \in E(G) \), there is a spanning tree \( T \) of \( G \) such that \( e \in E(T) \).

Problem 4

Suppose that we have an unweighted, connected graph \( G \). We want to find a spanning tree of \( G \) that does not include a given edge \( e \in E(G) \). We know that \( e \) lies on a cycle.

(a) Prove that such a spanning tree exists.

(b) Describe a way in which we can use one of our known MST algorithms to solve this problem.

(c) What if we want to include the edge \( e \)? (As we proved is possible in the previous question.)
Problem 5  ⋆

Suppose that $G$ is a connected, weighted graph. Show that for every $v \in V(G)$, Kruskal’s algorithm includes an edge of lowest weight incident to $v$.

Problem 6  ⋆

Suppose that we have a complete, weighted graph on $n$ vertices. We would like to find a Hamiltonian cycle of minimum total weight (defined similarly to our minimum weight spanning trees). Clearly, $K_n$ is Hamiltonian, so such a cycle exists. Consider the following greedy algorithm for solving this problem: Start at an arbitrary vertex $v$, and move along the edge of minimum weight among all of its edges that lead to an unvisited vertex. Continue until all vertices have been visited, then take the edge from the last vertex back to $v$. Discuss the correctness of this algorithm.