Problem 1. Recall that a tree is a connected graph without cycles. Prove:

- Every tree has at least a leaf node, i.e., a node with degree 1 (i.e., a node incident to only one edge).
- Every tree with \( n \) nodes has precisely \( n - 1 \) edges.

Problem 2. Let \( G \) be a simple graph with \( n \) vertices and \( n - 1 \) edges. Prove: if \( G \) is connected (i.e., a path exists between any two vertices in \( G \)), then \( G \) must be a tree.

Problem 3 (one for one, still a tree). Let \( T \) be a tree. Add a new edge between two vertices in \( T \); this gives us a graph \( G \) with a cycle \( cyc \). Now, remove from \( G \) an arbitrary edge \( e' \) of \( cyc \); let \( G' \) be the graph thus obtained. Prove: \( G' \) is a tree.

Problem 4. Let \( S \) be a set of integer pairs of the form \((id, v)\). We will refer to the first field as the \( id \) of the pair, and the second as the \( key \) of the pair. Design a data structure that supports the following operations:

- Insert: add a new pair \((id, v)\) to \( S \) (you can assume that \( S \) does not already have a pair with the same id).
- Delete: given an integer \( t \), delete the pair \((id, v)\) from \( S \) where \( t = id \), if such a pair exists.
- DeleteMin: remove from \( S \) the pair with the smallest key, and return it.

Your structure must consume \( O(n) \) space, and support all operations in \( O(\log n) \) time where \( n = |S| \).

Problem 5. Prove: in a weighted undirected graph \( G = (V, E) \) where all the edges have distinct weights, the minimum spanning tree (MST) is unique.

Problem 6. Describe how to implement the Prim's algorithm on a graph \( G = (V, E) \) in \( O((|V| + |E|) \cdot \log |V|) \) time.