Due by the beginning of class, Oct. 1.

1. Define a subtree of be any connected subgraph of a tree (this is different than the definition in the book).
   (a) Prove that the number of subtrees of a complete binary tree is not polynomial in the number of nodes.
   (b) Give an example of a class of trees \( \{T_n\} \) where the number of subtrees is a polynomial in the number of nodes.

2. Show that if you have a polynomial time algorithm for Hamiltonian Path, that you have a polynomial time algorithm for sorting.

3. The **Bounded Degree Spanning Tree** (BDST) problem is the following:
   **Input:** Graph \( G \) and integer \( k \).
   **Output:** Yes, if \( G \) has a spanning tree where every node has degree at most \( k \), No, otherwise.
   Suppose there is no polynomial time algorithm for Hamiltonian Path. Show that there is no polynomial time algorithm for BDST.

4. Let \( T = (V, E) \) be an edge weighted tree such that \( e \in E \) has minimal weight.
   Let \( T_1 \) and \( T_2 \) be the trees derived from \( T \) by removing \( e \). Then we define a **cartesian tree** of \( T \) to be a binary tree such that \( e \) is the root, and the left and right children of \( e \) are the cartesian trees of \( T_1 \) and \( T_2 \), respectively. If either \( T_1 \) or \( T_2 \) are singleton nodes, then their cartesian trees are empty.
   Give an algorithm for finding a cartesian tree of a tree. Give an analysis of its running time. The faster the algorithm, the better your grade. (Hint: Read about the \( O(n \log n) \) algorithm for Union-Find in the book or online.)

5. Give a lower bound of \( \Omega(n \log n) \) for constructing the cartesian tree of a tree.