Directions: To receive full credit:

- Place your name at the top of each page.
- Start each problem on a new page.

1. Let $T$ be a tree (not necessarily a binary tree) where all edges may be traversed in either direction. The diameter of a vertex $v$ in $T$ is the number of edges on a longest path between two vertices (not necessarily leaves) in the subtree rooted at $v$ (this path does not necessarily pass through $v$). Assuming that vertices are already labeled with their height, give a linear time recursive algorithm DIAMETER($v$) to label each vertex with its diameter. Justify why the time is linear and analyze the amount of space used in addition to that already used to store the tree.

2. Present an algorithm that uses at most $n + \lceil \log_2(n) \rceil - 2$ comparisons to find the largest and second largest element of a list of $n$ distinct elements:

3. Let $S$ be a set of $n$ items on which an ordering is defined. Describe how to combine the ideas of open hashing and 2-3 trees into a data structure to store $S$ that supports INSERT and MEMBER operations in $O(1)$ expected time and $O(\log(n))$ worst-case time.

4. With closed hashing, whenever an item is hashed to a position that is already occupied, search for some other place in the table to put it (so items are stored directly in the table). Present pseudo-code for the MEMBER and INSERT operations with linear probing, where you simply scan from that position forward (wrapping around if you get to the end of the table) until an empty position is found.