

Homework 2 (55 pts)

- (5 points) Draw a single binary tree T such that each of the following properties holds:
 - each **internal** node of T stores a single character
 - a **preorder** traversal of T yields COMPILER, and
 - a **inorder** traversal of T yields PMIOLCE.
- (10 points) Give the **pseudocode** for an $O(n)$ -time algorithm that computes the depth of each node of a tree T , where n is the number of nodes of T . Assume the existence of methods $\text{setDepth}(v,d)$ and $\text{getDepth}(v)$ that run in $O(1)$ -time.
- (10 points) Design an algorithm, $\text{inorderNext}(v)$, which returns the node visited after node v in an inorder traversal of binary tree T of size n . Analyze its worst-case running time. Your algorithm should avoid performing traversals of the entire tree.
- (10 points) Let T be a binary tree with n nodes. It is realized with an implementation of the Binary Tree ADT that has $O(1)$ running time for all methods except $\text{positions}()$ and $\text{elements}()$, which have $O(n)$ running time. Give the **pseudocode** for a $O(n)$ time algorithm that uses the methods of the Binary Tree interface to visit the nodes of T by increasing values of the level numbering function p given in Section 2.3.4. This traversal is known as the **level order traversal**. Assume the existence of an $O(1)$ time $\text{visit}(v)$ method (it should get called once on each vertex of T during the execution of your algorithm)
- (a) (5 points) Illustrate the execution of the selection-sort algorithm on the following input sequence:
(21, 14, 32, 10, 44, 8, 2, 11, 20, 26)
(b) (5 points) Illustrate the execution of the insertion-sort algorithm on the following input sequence:
(21, 14, 32, 10, 44, 8, 2, 11, 20, 26)
- Let S be a sequence containing pairs (k, e) where e is an element and k is its key. There is a simple algorithm called count-sort that will construct a new sorted sequence from S provided that all the keys in S are different from each other. For each key k , count-sort scans S to count how many keys are less than k . If c is the count for k then (k, e) should have rank c in the sorted sequence.
 - (5 points) Give the **pseudocode** for count-sort as it is described above.
 - (3 points) Determine the number of comparisons made by count-sort. What is its running time?
 - (2 points) As written, count-sort only works if all of the keys have different values. Explain how to modify count-sort to work if multiple keys have the same value.