Ethics: Any behavior on any homework or exam that could be considered copying or cheating will result in an immediate zero on the assignment for all parties involved and will be reported to academichonesty@iit.edu See the IIT Code of Academic Honesty, https://web.iit.edu/student-affairs/handbook/fine-print/code-academic-honesty

1. (3 points) Give an O(n) algorithm for the following problem and prove its time complexity. Given a list of n distinct positive integers, partition the list into two sublists, each of size n/2, such that the difference between the sums of the integers in the two sublists is maximized. You may assume that n is a multiple of 2.

2. (5 points) Suppose we are given an array of numbers A[0], A[2], . . . , A[n - 1]. You may assume that n is a multiple of 6. We wish to find a number x that simultaneously satisfies two properties: First, x should be in the middle two-thirds of the array (in other words, x = A[i] for some n/6 <= i < 5n/6). Second, after sorting the array, x should still be in the middle two-thirds.

2a) Prove that such an x exists. (Hint: how many values are not in the middle two-thirds of the original array? How many are not in the middle two-thirds of the sorted array?)

2b) Describe a linear time algorithm for finding such an x. (Note, if your solution calls one of the algorithms described in the lectures, you should just refer to it by name, e.g. “quicksort”, rather than explaining how it works.)

3. (4 points) Assume that are given a "black-box" (i.e., you do not have the source code) procedure MEDIAN that takes as parameters an array A and subarray indices p and r, and returns the value of the median element of A[p . . r] in O(n) time in the worst case. Give a simple, linear-time algorithm that uses this procedure MEDIAN to find the ith smallest element. Write the recurrence relation for your algorithm and show the solution is linear growth.

4. (3 points) The set of full binary trees is defined recursively:

Basis step: The tree consisting of a single vertex is a full binary tree.

Recursive step: If T1 and T2 are disjoint full binary trees, there is a full binary tree, denoted by T1 · T2, consisting of a root r together with edges connecting r to each of the roots of the left subtree T1 and the right subtree T2.

Use structural induction to show that l(T), the number of leaves of a full binary tree T, is 1 more than i(T), the number of internal vertices of T.

5. (3 points) An in-order tree walk of an n-node binary search tree can be implemented by finding the minimum element in the tree with TREE-MINIMUM and then making n-1 calls to TREE-SUCCESSOR. Prove that this algorithm runs in Θ(n) time.

6. (3 points) Is the operation of deletion "commutative" in the sense that deleting x and then y from a binary search tree leaves the same tree as deleting y and then x? Argue why it is or give a counterexample.
7. (4 points) We can sort a given set of n numbers by first building a binary search tree containing these numbers (using TREE-INSERT repeatedly to insert the numbers one by one) and then printing the numbers by an in-order tree walk. What are the worst-case and best-case running times for this sorting algorithm? Show detailed analysis.