CS 624: Analysis of Algorithms

Assignment 3

Due: Tuesday, March 26

Topics: Sorting Bounds, Medians and order statistics, BST.

1. Given $n > 2$ distinct numbers, what is the minimum number of comparisons needed to find a number that is neither the maximum nor the minimum?
2. Sorting the i largest numbers: Given a set of n numbers, we wish to find the i largest **in sorted order** using a comparison-based algorithm. Describe the worst case run time of each of the three algorithms below, both in terms of n and i.
   (a) Sort the numbers, and list the i largest.
   (b) Build a max-priority queue from the numbers, and call EXTRACT-MAX i times.
   (c) Use an order-statistic algorithm to find the i^{th} largest number, partition around that number, and sort the i largest numbers.
3. Show that Randomized-Select never makes a recursive call to a 0-sized array.
4. Exercise 1.1 in Lecture note 7 (binary search trees).
5. Exercise 1.2 in Lecture note 7 (binary search trees).
6. Exercise 1.3 in Lecture note 7 (binary search trees).
7. Exercise 1.4 in Lecture note 7 (binary search trees).
8. Exercise 1.5 in Lecture note 7 (binary search trees).
9. Exercise 1.6 in Lecture note 7 (binary search trees).
10. Exercise 4.1 in the Lecture 7 handout - you have to prove all these properties are correct (for many of the proofs - use contradiction) Please be careful. This is an exercise about binary search trees, not about algorithms used to construct those trees. So all you can use in doing this problem is the definition of a binary search tree. If you write something like, "this can't happen because the algorithm would have placed this element somewhere else", then your reasoning can't possibly be correct.