Assignment 6

due Friday, March 9, 2012

1. Consider the BUILD-MAX-HEAP' procedure of exercise 6-1 (pp 166-167). Show that in the worst case, BUILD-MAX-HEAP' requires $\Omega(n \log n)$ time. [5 points]

2. A min-max heap $H$ combines properties of both a min-heap and a max-heap. It is defined as a complete binary tree with the following property: let $v$ be a node of $H$ at depth $i$. Then
   - if $i$ is odd, then the key stored in node $v$ is less than or equal to any other value stored in the subtree of $H$ rooted at $v$.
   - if $i$ is even, then the key stored in node $v$ is greater than or equal to any other value stored in the subtree of $H$ rooted at $v$.

   Illustrate a min-max heap by drawing one containing 15 elements. [6 points]

3. Exercise 19.2-1, p 518 [4 points]

4. From the tree used in the previous problem (before the extractMin), decrease 35 to 20. [3 points]

5. Suppose you have an array $S$ of size $n$, where each element in $S$ represents a different vote for class president, where each vote is given as an integer representing the student ID of the candidate. Without making any assumptions about who is running or how many candidates there are, design an $O(n \log n)$ algorithm to determine which candidate receives the most votes. [6 points]

6. Consider a modification to the previous problem to a situation where we know the number $k < n$ of candidates running. Design an $O(n \log k)$ algorithm to determine which candidate receives the most votes. [6 points]

7. Suppose you have a set $A$ of $n$ nuts and a set $B$ of $n$ bolts, such that each nut in $A$ has a unique matching bolt in $B$. The only kind of comparison you can make is to take a nut-bolt pair $(a, b)$, where $a \in A$ and $b \in B$, and test to see whether the threads of $a$ are larger, smaller, or a perfect match to the threads of $b$. Give an efficient (randomized) algorithm to match up all the nut and bolts. What can you say about the run-time of your algorithm? [6 points]

Total: 36 points

Notes:

- Q1: Describe a bad input sequence with $n$ elements, and then show that $\Omega(n \log n)$ time is used on that sequence.
• **Q6**: Do not sort, but think of a BST like structure.

• **Q7**: Partition, partition, partition as with randomized quicksort. This is much like exercise 8-4(c).