Design and Analysis of Algorithms
Homework 3

Clearly number your solution to each problem. Staple your solutions and bring them to class on the due date. Express your algorithms in pseudo-code when directed. Always provide justification for your answer when asked to give the running time of an algorithm. Be brief and concise, and draw pictures where appropriate.

1. (a) Illustrate the execution of the heap-sort algorithm on the following sequence: (2, 5, 16, 4, 10, 23, 39, 18, 26, 15). Show the contents of the heap and the sequence at each step of the algorithm.

   (b) Illustrate the execution of the bottom-up construction of a heap (like in Figure 2.49) on the following sequence: (2, 5, 16, 4, 10, 23, 39, 18, 26, 15, 7, 9, 30, 31, 40).

2. Let T be a heap storing n keys. Give the pseudocode for an efficient algorithm for reporting all the keys in T that are smaller than or equal to a given query key x (which is not necessarily in T). For example, given the heap of Figure 2.41 and query key $x = 7$, the algorithm should report 4,5,6,7. Note that the keys do not need to be reported in sorted order. Your algorithm should run in $O(k)$ time, where k is the number of keys reported. Provide justification that your algorithm runs in $O(k)$ time.

3. (a) Insert into an initially empty binary search tree items with the following keys (in this order): 30, 40, 23, 58, 48, 26, 11, 13. Draw the tree after each insertion.

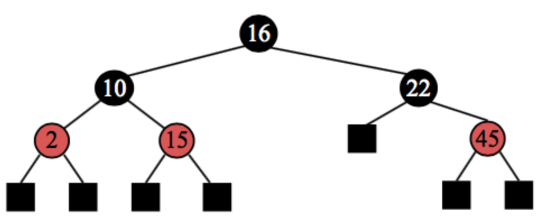
   (b) Remove from the binary search tree in Figure 3.7(a) the following keys (in this order): 32, 65, 76, 88, 97. Draw the tree after each removal.

   (c) A different binary search tree results when we try to insert the same sequence into an empty BST in a different order. Give an example of this with at least 5 elements and show the two different binary search trees that result.

4. (a) Let T be a binary search tree, and let x be a key. Give the pseudocode for an efficient algorithm for finding the smallest key y in T such that $y > x$. Note that x may or may not be in T. Explain why your algorithm has the running time it does.

   (b) Give the pseudocode for a nonrecursive algorithm to print out the keys from a binary search tree in order.

5. (a) Consider the following sequence of keys: (18, 30, 50, 12, 1). Insert the items with this set of keys in the order given into the red-black tree in the figure below. Draw the tree after each insertion.



   (b) Design and give the pseudocode for an $O(\log n)$ algorithm that determines whether a red-black tree with n keys stores any keys within a certain (closed) interval. That is, the input to the algorithm is a red-black tree T and two keys, l and r, where $l \leq r$. If T has at least one key k such that $l \leq k \leq r$, then the algorithm returns true, otherwise it returns false.