Assignment 4

R-2.8 Illustrate the performance of the selection-sort algorithm on the following input sequence (22, 15, 26, 44, 10, 3, 9, 13, 29, 25).

R-2.9 Illustrate the performance of the insertion-sort algorithm on the input sequence of the previous problem.

R-2.10 Give an example of a worst-case sequence with n elements for insertion-sort runs in $\Omega(n^2)$ time on such a sequence.

R-2.13 Suppose a binary tree T is implemented using a vector S, as described in Section 2.3.4. If n items are stored in S in sorted order, starting with index 1, is the tree T a heap? Justify your answer.

R-2-18 Draw an example of a heap whose keys are all the odd numbers from 1 to 49 (with no repeats), such that the insertion of an item with key 32 would cause up-heap bubbling to proceed all the way up to a child of the root (replacing that child's key with 32).

C-2.32 Let *T* be a heap storing n keys. Give 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 on 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. Ideally, your algorithm should run in O(k) time, where k is the number of keys reported.

Design an algorithm, **isPermutation(A,B)** that takes two sequences A and B and determines whether or not they are permutations of each other, i.e., same elements but possibly occurring in a different order. **Hint**: A and B may contain duplicates.

What is the worst case time complexity of your algorithm? Justify your answer.