Assignment #5 (Due on Dec 8th, before the beginning of class)

This problem set count as the fifth programming assignment, but does not necessarily require programming. Feel free to implement algorithms you design in order to test them thoroughly. If you do this, include your source code and outputs from your testing.

If your score on this assignment is higher than any of the first four, it will count as pa-5, and will additionally replace the lowest score of the first four. You must follow all the instructions for credit! Do not forget your name, start early and take pride in the work you submit.

Instructions

· You may discuss problems with instructors or other students, however all work submitted must be your own!
· Please include this sheet with your solutions and type them or write extremely neatly. Your work must be easily legible in order for it to be graded.
· Show all of your work and reasoning!
· Post to Piazza for clarifications if you have any questions.
· The total value of the assignment is 100 points, where each ⋆ represents 5 points and each ♦ represents 10 points.

Trees and Heaps

1. ⋆ Show that the maximum number of nodes in a binary tree of height $h$ is $2^{h+1} - 1$.

2. ⋆ A full node is a node with two children. Prove that the number of full nodes plus one is equal to the number of leaves in a nonempty binary tree.

3. ⋆ What is the minimum number of nodes in an AVL tree of height 15?

4. ♦ Show the result of inserting 14, 12, 18, 20, 27, 22, 24 into an initially empty AVL tree. Redraw the tree after each insertion, and indicate clearly the type of rotation, if any, being performed.

5. ♦ Design a recursive linear-time algorithm that tests whether a binary tree satisfies the search tree order property at every node.

6. ♦ Show the result of inserting 10, 12, 1, 14, 6, 5, 8, 15, 3, 9, 7, 4, 11, 13, and 2, one at a time, into an initially empty Binary Max Heap.

7. ⋆ Show the result of performing three deleteMax operations in the heap of the previous exercise.
Sorting

1. ★ Clearly show each step of Insertion, Selection, Merge and Quick sorts on the sequence 3, 1, 4, 1, 5, 9, 2, 6, 5.

2. ★ What is the running time of heapsort for presorted input?

3. ♦ Determine the running time of mergesort for:
   (a) sorted input
   (b) reverse-ordered input
   (c) random input (you may assume a uniform random distribution)

4. ★ Suppose we choose the element in the middle position of the array as pivot. Does this make it unlikely that quicksort will require quadratic time? Why or why not? Explain your reasoning.

5. ♦★ When implementing quicksort, if the array contains lots of duplicates, it may be better to perform a three-way partition (into elements less than, equal to, and greater than the pivot) to make smaller recursive calls. Assume three-way comparisons. Give an algorithm that performs a three-way in-place partition of an N-element subarray using only N − 1 three-way comparisons. If there are d items equal to the pivot, you may use d additional Comparable swaps, above and beyond the two-way partitioning algorithm. (Hint: As i and j move toward each other, maintain five groups of elements as shown below):

```
EQUAL    SMALL    UNKNOWN    LARGE    EQUAL
     i          j
```

If you decide to write pseudocode, still include a description, or detailed comments.

6. ◊ Suppose arrays A and B are both sorted and both contain N elements. Give an O(logN) algorithm to find the median of A ∪ B. Write a detailed description or pseudocode, or both.