Algorithms

Homework Set #2, due Feb 10

1. In class, we found that the solution to the mergesort recurrence
$$f(n) = f\left(\lceil \frac{n}{2} \rceil\right) + f\left(\lfloor \frac{n}{2} \rfloor\right) + n; \quad f(1) = 0$$
satisfied $f(n) = \Theta(n \log n)$. Find an expression for the exact value of $f(n)$. (Hint: generate the first 20 to 40 values and find a pattern. Prove your conjecture by induction on the number of bits required to express n.)

2. A sorting algorithm is called stable if it preserves the relative order of any two equal elements in its input. That is, if $a[i] = a[j]$ with $i < j$ in the array before sorting, then the entry $a[i]$ will end up in the sorted array in a position before the one where $a[j]$ ends up, for any such pair (i, j). Which of the sorting algorithms we considered are stable? Why might stability be a good property of a sorting algorithm?

3. How could you simply modify the recursion in our version of quicksort to allow the program to switch to a simpler routine, say, insertion sort, for subarrays of length below some cutoff value? Write such a program and experiment with the cutoff sizes 4, 8, and 16 for an array of 5000 randomly generated floats to see what the effect on the number of comparisons might be.

4. Sedgewick, 2.1.2, 2.2.22

5. How many different heaps are there for an array of a) 6 entries? b) 12 entries? c) 18 entries? Explain your answer.

6. Bonus: If b blue balls and r red balls are randomly distributed in b blue boxes and r red boxes, one ball per box, show that the expected number of red balls in blue boxes is
$$\frac{br}{b+r}.$$
Use this fact to show that the partitioning method we used for Quicksort performs $(N + 4)/6$ swaps on average. This is roughly one-sixth the number of comparisons performed.