Data Structures and Algorithms

Assignment #5

Due: Friday February 28

This assignment will let you get more acquainted with Big Oh, Omega and Theta and recurrence relations. You'll also get to know your sorting algorithms.

1. (10 points).
Show that $f(N) = 12N^2 + 25N + 500 logN$ is $\Theta(N^2)$ by showing that it is both $O(N^2)$ and $\Omega(N^2)$ (cf. Weiss pp 29-31). Do this by selecting appropriate constants for n_0 and c for O and Ω.

2. (10 points).
What is the running time (in big Oh notation) of the following recursive function?

```
Thirds (int N) : int {
   if (N < 3) then {
       return 1;}
   else {
       return Thirds(N/3) + Thirds(N/3) + Thirds(N/3); }
}
```

First, write down the recurrence relation and then solve it by expanding the terms (it might be convenient to assume that $N = 3^k$).

3. (30 points)

- Weiss 7.1. Write your answer in the form of a table as in Weiss Figure 7.1 .
- Weiss 7.11. First show the result of BuildHeap and then the result after each DeleteMax. Draw both the tree-structured heap and the input array as in Weiss Figures 7.6 and 7.7.
- Show the steps for in-place partitioning using a median-of-three pivot for the sequence 5, 1, 4, 7, 3, 9, 2, 0, 6, 8. Draw figures similar to those of Lecture 14 Slides 26-28.

4. (3 points) Weiss 7.2.

5. (5 points) There are two strategies for handling small arrays in Quicksort.

The first strategy is the one described in class (cf. Lecture 14 Slide 29). Apply Quicksort recursively until the array is smaller than a CUTOFF size, then call Insertionsort to sort the small array.

A second strategy is to apply Quicksort recursively until the array is smaller than the CUTOFF, as above, but then return. In this strategy after Quicksort(A[],1,n) is completed, the array A[1,n] is almost sorted. Now call Insertionsort(A[],1,n) to finish the job. Since A[1..n] is almost sorted then Insertionsort should do a good job.

Explain why the number of comparisons executed by Insertionsort in the two strategies is the same.