AN OLD MIDTERM

`Warning:` There are some changes in the syllabus from year to year – this exam was given earlier in the term and doesn't include later material.

Open book. Open Notes. Total Points: 130. Do all problems.

1. (15) Let $A[1 \cdots N]$ be an array with all entries integers between 0 and $N-1$. How long would `RADIX-SORT` take to sort A assuming that we use base 2 (that is, binary)? (Assume the enries $A[I]$ are already given as binary strings in the input.) You *must* give an argument for your answer. Give (no proofs required!) a *faster* way to sort this data.

2. (15) `Frank` is an algorithm similar to the Karatsuba algorithm discussed in class. It multiplies two n digit numbers by making five recursive calls to multiplication of two $n/4$ digit numbers plus forty additions and subtractions. Each of the additions and subtractions take time $O(n)$. Give the recursion for the time $T(n)$ for `Frank` and use the Master Theorem to find the asymptotics of $T(n)$. Compare with the time $\Theta(n^{\log_2 3})$ of Karatsuba. Which is faster when n is large? (If you don't have a calculator handy, tell what quantities you need compare to tell which is faster.)

3. (20) Let A be an array of length 1023 in which the values are distinct and in increasing order.

   (a) In the procedure `BUILD-MAX-HEAP(A)` *precisely* how many times will two elements of the array be exchanged? (Reason, please!)

   (b) Now suppose the values are distinct and in decreasing order. Again, in the procedure `BUILD-MAX-HEAP(A)` *precisely* how many times will two elements of the array be exchanged? (Reason, please!)

4. (20) Consider the recursion

$$T(3n) = 9T(n) + 4n^2$$

   with initial value $T(1) = 5$.

   (a) (5) Find $T(9)$ *precisely*.

   (b) (5) Use the Master Theorem to give the asymptotics of $T(n)$ in Theta-land. (Brief explanation, please.)

(c) (10) Using a suitable auxilliary variable find a *precise* formula for $T(n)$ where n is a power of 3. (Write $n = 3^t$. Your formula can use n and/or t.)

5. (20) Give an algorithm `TINYPIECES` that does the following. As *input* you have an array $PRICE[1 \cdots N]$ where, for $1 \leq i \leq N$, $PRICE[i]$ is the price of a rod of length i. You are given a rod of total length N^5. You wish to cut it into pieces (*but* all pieces must be of length at most N) so as to maximize the total price. Your algorithm should output $VALUE$, where this represents the maximal total price. (Note: You are not being asked to find the actual cutting of the rod.) Analyze (in Θ-land) the total time your algorithm takes. You **must** give a description in clear words of what the algorithm is doing.

6. (20) Describe the algorithm `QUICKSORT`(p,r) which sorts the elements $A[i]$, $p \leq i \leq r$. (You can assume $p \leq r$.) You may, and should, use auxilliary arrays. Subroutines must be described in full. Explain *in clear words* what the algorithm is doing. Give (without proof!) both the average and the worst-case time for `QUICKSORT`(1,n).

7. (20) Here is a psuedocode sorting algorithm that uses Binary Search Tree. We wish to sort $A[1 \cdots N]$. (There are no records here, each $A[I]$ is itself the key.) Begin with an empty BST T.
   Part I: FOR $I = 1$ to N; INSERT $A[I]$ into T; ENDFOR
   Part II: Apply IN-ORDER-TREE-WALK to T
   Analyze *both* the average time and the worst case time for this algorithm.