CS 381, Fall 1999

Sample Midterm Problems

1.) (i) Order the following functions according to their asymptotic growth rate. Indicate which functions belong to the same complexity class.
$4n \log n$, $2^n \log n$, $n^2 + 8n \log n$, 2^n, $(n+4)(n-6)$, $\sqrt{n}$, 2^{n-4}, $2^{n/2}$

(ii) Which are true: $5n = O(n \log n)$; $12n^2 = O(n \log n)$; $\frac{n}{\log n} = \Theta(n)$.

2.) Assume A is an array of size n containing integers in arbitrary order and A_s is an array of size n containing integers in sorted order. Give the running times (in big-O notation) for the specified operations. Give a brief explanation of each entry below the table.

	Given x, determine whether x is **not** in the array	Given x, determine whether x occurs at least $n/2$ times	Given x, determine the smallest element y in the array with $y > x$
A (not sorted)			
A_s (sorted)			

3.) (i) Use the master theorem to determine the tight asymptotic bounds of the following recurrence relations:

$$T(n) \leq \begin{cases} T(n/2) + c \log n & \text{if } n > 2 \\ 1 & \text{for } n = 2 \end{cases}$$

$$T(n) = \begin{cases} T(n/2) + \sqrt{n} & \text{if } n > 2 \\ 1 & \text{for } n = 2 \end{cases}$$

(ii) Consider the recurrence relation $T(n) = 3T(n-1) + 2$ with $T(1) = 1$.
Show by induction that $T(n) = O(3^n)$.

4.) Describe a data structure to implement the following version of a priority queue Q. Each operation should take $O(\log n)$ time, where n is the current number of elements in Q.

Insert(Q, x) - insert element x into Q

DeleteMax(Q) - delete the largest element from Q

DeleteMin(Q) - delete the smallest element from Q.

Sketch how each operation is implemented and analyze the achieved time bounds.

5.) Assume you are given two sets S_1 and S_2 (not sorted), which contain a total of n integers, and an integer x. Determine whether there exists an element in S_1 and an element in S_2 such that the sum of the two elements is equal to x. The running time should be $O(n \log n)$.

6.) Let A be an array of even size, say n, containing integers. The problem is to partition the elements in A into $n/2$ pairs with the following property: for every pair formed, determine the sum. Let $s_1, s_2, \ldots, s_{n/2}$ be these sums. Pairs should be formed so that the maximum of the s_i's is a minimum.

(i) For $A = [4, -6, 14, 8, 1, 5, -2, 23, 7, 15]$, give the partition into pairs minimizing the maximum sum.

(ii) Describe an efficient algorithm to determine a partitioning minimizing the maximum sum.