CS 300 Midterm Sample Problems

1.(10 pts) The running time of QUICKSORT depends on both the data being sorted and the partition rule used to select the pivot. Suppose we always pick the pivot element to be the key of the median element of the first three keys of the subarray. On a sorted array, determine whether QUICKSORT now takes $\Theta(n)$, $\Theta(n \log n)$, or $\Theta(n^2)$. Justify your answer.

2.(9 pts $\times$ 6 = 54 pts) Give T(True) or F(False) for each of the following statements. Justify your answers. Incorrect justification will earn 0 pts even with correct answer.

(1) We can sort n integers in the range 0 to n^3-1 in O(n) time.

(2) $f(n) = O(g(n))$ implies $2^{f(n)} = O(2^{g(n)})$.

(3) $f(n) = \Theta(f(n/2))$.

(4) If we use a d-ary heap instead of binary heap, EXTRACT-MAX operation that extracts the maximum element from the heap takes $\Theta(\lg n / \lg d)$.

(5) We can find 2 smallest of n elements with $n + \lceil \lg n \rceil - 2$ comparisons.

(6) If we use groups of 7, instead of 5, in the worst-case linear-time selection algorithm, the algorithm still works in linear time. Justify your answer by giving a recurrence for this algorithm.

3. Consider the following divide-and-conquer algorithm to multiply two n-bit numbers X and Y. We divide X into A and B and Y into C and D, where A, B, C, D are n/2-bit numbers.

$$X = 2^{n/2}A + B$$

A	B

$$Y = 2^{n/2}C + D$$

C	D

$$XY = 2^n AC + 2^{n/2}BC + 2^{n/2}AD + BD$$

(a) (10 pts) Give a recurrence for the running time of the algorithm and give a tight asymptotic bound.

(b) (10 pts) Observe that we can also obtain XY as follows : XY =

$$(2^n - 2^{n/2})AC + 2^{n/2}(A+B)(C+D) + (1-2^{n/2})BD$$

Give a recurrence for the running time of this algorithm and give a tight asymptotic bound.

4. (4 pts × 4 = 16 pts) Solve the following recurrences by giving tight Θ-notation bounds. Justify your answer.

(1) $T(n) = 7T\left(\frac{n}{2}\right) + n^2$

(2) $T(n) = 2T\left(\frac{n}{3}\right) + n \log n$

(3) $T(n) = 4T\left(\frac{n}{2}\right) + n^3$

(4) $T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{4}\right) + T\left(\frac{n}{8}\right) + n$

5. (6 pts × 4 = 24 pts) Give T(True) or F(False) for each of the following statements. Justify your answers.

(1) $f_1(n) = \Theta(g_1(n))$ and $f_2(n) = \Theta(g_2(n))$. Then, $f_1(n) \cdot f_2(n) = \Omega(g_1(n) \cdot g_2(n))$.

(2) There exists a comparison sort of 6 numbers that uses at most 9 comparisons in the worst case.

(3) We can use QUICKSORT as the intermediate sort of radix sort.

(4) The running time of QUICKSORT when all elements of array A have the same value is $O(n \log n)$.

6. (10 pts) We are given n points in the unit circle, $p_i = (x_i, y_i)$, such that $0 < x_i^2 + y_i^2 \leq 1$ for $i = 1, 2, \ldots, n$. Suppose that the points are uniformly distributed; that is, the probability of finding a point in any region of the circle is proportional to the area of that region. Design an algorithm with an expected running time of $\Theta(n)$ to sort the n points by their distances $d_i = \sqrt{x_i^2 + y_i^2}$ from the origin. Explain briefly and analyze the running time.

8. (10 points) Let $X[1 \ldots n]$ and $Y[1 \ldots n]$ be two arrays, each containing n numbers already in sorted order. Give an $O(\log n)$-time algorithm to find the k-th smallest element of all 2n element in array X and Y. (You may assume that all elements are distinct.)

9. (10 pts × 2 = 20 pts) We are looking at the price of a given stock over n consecutive days, $i = 1, 2, \ldots, n$. For each day i, we have a price $p(i)$ for the stock on that day. We'd like to know: How should we choose a day i on which to buy the stock and a later day $j > i$ on which to sell it, if we want to maximize the profit, $p(j) - p(i)$? (profit can be negative, if there is no way to make money during the n days.)

(1) Using Divide-and-Conquer approach, find the optimal profit, such that $p(j) - p(i)$ is maximized. Explain your algorithm and analyze the running time.

(2) Using Dynamic Programming, find the optimal profit in $O(n)$ time. Explain your algorithm briefly and analyze the running time and space requirements.

10. Solve the following recurrences by giving tight Θ-notation bounds. Justify your answers. (No justification with correct answer will get no credit.)

(a) (5 pts) $T(n) = 2\,T(n/4) + n \lg n$

(b) (5 pts) $T(n) = T(n-2) + \lg n$

11.(10 pts) Using the substitution method, give a tight asymptotic upper bound on the solution to the following recurrence using O-notation.

$$T(n) = T(n/2) + T(n/4) + n$$

12. (5 pts) Consider a modification to QuickSort, such that each time PARTITION is called, the median of the partitioned array is found (using the SELECT algorithm) and used as a pivot. What is the worst-case running time of this algorithm? Justify your answer.

13. Let A[1..n] be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an ***inversion*** of A.

(a) (5 pts) What array with elements from the set {1, 2, …, n} has the most inversions? How many does it have?

(b) (5 pts) What is the relationship between the running time of insertion sort and the number of inversions in the input array? Justify your answer.

(c) (5 pts) Give an algorithm that determines the number of inversions in any permutation on n elements in $\Theta(n \lg n)$ worst-case time. Analyze the running time of your algorithm.

14. Give True ot False for each of the following statements. Justify your answers. Incorrect justification will earn 0 pts even with correct answer.

(a) (5 pts) For every two positive functions f and g, if g(n) = O(n), then f(g(n)) = O(f(n)).
(b) (5 pts) n^2th Fibonacci number can be computed in O(*lg n*) time.
(c) (5 pts) Heapsort can be used as the auxiliary sorting routine in radix sort, because it operates in place.
(d) (5 pts) There exists a comparison sort of 5 numbers that uses at most 6 comparisons in the worst case.
(e) (5 pts) An adversary can provide randomized quicksort with an input array of length *n* that forces that algorithm to run in $\omega(n\ lg\ n)$ time on that input.