Library

DEPARTMENT OF MATHEMATICS, IIT - Kharagpur
Mid Semester Examination (Autumn 2016)
MA 21007 Design & Analysis of Algorithms
Instructor: Dr. Sourav Mukhopadhyay
No. of students: 205 Total Points: 30 DURATION: 2 Hours

Answer **ALL QUESTIONS**. All the notations are standard and no query or doubts will be entertained. If any data/statement is missing, identify it in your answer script. Marks are indicated at the end of each question.

1. Consider the recurrence $T(n) = T(n/2) + T(n/4) + n$. Use the substitution method to give a tight upper bound on the solution to the recurrence using O-notation. [2]

2. For each of the following algorithms, (i) give a recurrence that describes its worst-case running time and (ii) its worst-case running time using Θ-notation: (a) Binary search, (b) Insertion Sort, (c) Merge Sort, (d) Randomized quicksort and (e) Strassen's algorithm. [5]

3. Consider the following sorting methods: Insertion Sort, Merge Sort, and Quick Sort. What is the running time using O-notation for each method

   (a) When all the the array values are equal?

   (b) When the values are in order?

   (c) When the values are in reverse order? [3]

   Explain your answers.

4. Consider the following outline of quicksort:

```
procedure QuickSort(List);
begin
  if (list has more than one item) then
     begin
     Choose a pivot element from the list;
     Partition list into two lists, L and R, using the chosen pivot.
     Sort L using QuickSort(L)
     Sort R using QuickSort(R)
     Return(QuickSort(L) followed by QuickSort(R))
     end
   else (Do nothing- list already sorted)
end
```

   (a) What is the worst-case choice for a pivot?

   (b) What is the best-case choice for a pivot?

   (c) The median of a set of n numbers is a number x such that at least $\lfloor \frac{n}{2} \rfloor$ numbers are at most x and at least $\lfloor \frac{n}{2} \rfloor$ are at least x. In other words, if the numbers were to be sorted, the median would be in the middle of the list. Suppose that someone gives you a method FindMedian to find the median of n numbers in $O(n)$ time. How would you use FindMedian to improve the Quicksort method outlined above?

   (d) Write a recurrence relation for the worst-case running time for your new version of Quicksort.

   (e) What is the worst-case running time for the new version of quicksort? You should express your answer using O-notation. [5]

5. (a) Write a pseudo-code for finding the k-th largest element in an array of n elements in linear time **without using any extra storage**.

(b) Illustrate the above algorithm on the following sequence by finding the 3-rd largest element:

$$13, 14, 15, 16, 17, 12, 11, 10, 9$$

(c) Explain why the average computing time of the above algorithm is linear. [2 + 1 + 1]

6. (a) Use the integer hash function $h(x) = x \bmod 11$ and table size 11. Using chaining with separate lists, show the location in the hash table for each integer value in the following sequence:

$$7, 21, 45, 40, 65, 98, 44, 67$$

(b) Use the same hash function and give the table constructed by the linear probe method.

[3]

7. **TRUE OR FALSE?** If the statement is correct, briefly state why. If the statement is wrong, explain why. [8]

(a) By the master theorem, the solution to the recurrence $T(n) = 3T(n/3) + \log_2 n$ is $T(n) = \Theta(n \log_2 n)$.

(b) Every binary search tree on n nodes has height $O(\log_2 n)$.

(c) There exists a comparison sort of 5 numbers that uses at most 6 comparisons in the worst case.

(d) Heapsort can be used as auxiliary sorting routine in radix sort, because it operates in place.

(e) Let S be a set of n integers. One can create a data structure for S so that determining whether an integer x belongs to S can be performed in $O(1)$ time in the worst case.

(f) Suppose that an array contains n numbers, each of which is -1, 0 or 1. Then, the array can be sorted in $O(n)$ time in the worst case.

(g) Let A_1, A_2 and A_3 be three sorted arrays on n real numbers (all distinct). In the comparison model, constructing a balanced binary search tree of the set $A_1 \cup A_2 \cup A_3$ requires $\Omega(n \log_2 n)$.

(h) Let F_k denote the k-th Fibonacci number. Then, the n^2th Fibonacci number F_{n^2} can be computed in $O(\log_2 n)$ time.

——-The End——