DEPARTMENT OF MATHEMATICS, IIT - Kharagpur
Mid Semester Examination (Autumn 2018)
MA 21007 Design & Analysis of Algorithms
Instructor: Dr. Sourav Mukhopadhyay
No. of students: 240. 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. Find a solution to the recurrence $T(n) = T(\frac{n}{3}) + T(\frac{2n}{3}) + \Theta(n)$ [2]

2. Fill in the following table for the times to sort an array of n items. Use only big-O notation, and do not have any extraneous constants in your expressions.

	Worst Case	Average Case	Best Case
Binary search of a sorted array			
Insertion sort			
Merge sort			
Randomised Quicksort			
Heap sort			

[3]

3. a) Write an algorithm for inserting items in a Red-Black tree. What is the computing time of your algorithm?

   b) Start with an empty Red-Black tree and insert the following keys in the given order using your algorithm: $80, 100, 140, 60, 84, 30, 40, 50, 54, 52$

[4]

4. a) Describe a case where quicksort will result in quadratic behavior.

   b) Here is an array which has just been partitioned by the first step of Quicksort (sorting from smallest to largest):

   18 15 17 19 20 23 22 21 24

   Which of these elements could be a possible pivot?

   c) Consider the following array for sorting (sorting from smallest to largest)

   10 8 13 14 6 12 5 7 11 9

   (i) Draw the array after the FIRST iteration of the large loop in an Insersion sort.

   (ii) Draw the array after TWO recursive calls of Mergesort are completed, and before the final merge step has occured.

   (iii) Draw the array after applying Quicksort's partition function and using 10 for the pivot. Clearly describe the working of the partition function with this example.

[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. [6]

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. Binary searches are considerably faster than linear searches. Under what conditions can such an algorithm be used? Trace a binary search for the array $< 3, 6, 11, 13, 15, 19, 34, 35, 47 >$ for the value 6. When calculating an index, round up when needed. [2]

8. **TRUE OR FALSE?** If the statement is correct, briefly state why. If the statement is wrong, explain why. [5]

(a) Radix sort runs correctly when using any correct sorting algorithm to sort each digit.

(b) Given an array $A[1 \,..\, n]$ of integers, the running time of Counting Sort is polynomial in n.

(c) Any n-node unbalanced tree can be balanced using $O(\log n)$ rotations.

(d) 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.

(e) 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.

[End]