SPRING MID SEMESTER EXAMINATION-2018

Design & Analysis of Algorithms

[CS-2008]

Full Marks: 25 **Time: 1.5 Hours**

Answer any four questions including question No.1 which is compulsory.
The figures in the margin indicate full marks.
Candidates are required to give their answers in their own words as far as practicable and all parts of a question should be answered at one place only.

Q1 Answer the following questions: (2 x 5)

a) Define Big-Omega nonation.

b) Represent the execution time of the following code in O-Notation (least upper bound).

```
int fun(int n)
{
    int i = 1, sum = 1;
    while (sum ≤ n)
    {
       i++;
       sum = sum + i;
    }
    return sum;
}
```

c) Suppose we are sorting an array of eight integers using quick sort and we have just finished the first partitioning with the array looking like this

2, 5, 1, 7, 9, 12, 11, 10

Which of the following statement is correct?

i. The pivot could be either the 7 or the 9
ii. The pivot could be 7, but not 9
iii. The pivot is not the 7, but it could be the 9
iv. Neither the 7 nor the 9 is pivot

d) In KBC, Abhishek Bachan write down a number between 1 and 1000. Amitav Bachan must identify that number by asking "yes/no" question to Abhishek. If Amitav Bachan uses an optimal strategy then he will determine the answer at the end of exactly how many questions in worst case?

e) An array is intially sorted. When new elements are added, they are inserted at the end of the array and counted. Whenever the number of elements reached 20, the array is resorted and counter is cleared. Which sorting algorithm would be good choice to use for resorting the array?

Q2 a) Describe the step count and asymptotic efficiency of algorithms with INSERTION-SORT as an example. (2.5)

b) Solve the following recurrences. (2.5)
$T(n) = 2T(n/4) + \sqrt{n} \log_2 n$ with $T(1) = 1$

Q3 a) Given an unsorted array A[1..n], where one-third elements are sorted in ascending order and rest of elements are sorted in descending order. Design an algorithm to sort the array in O(n) worst-case time, in ascending order. (2.5)

b) Design an algorithm to calculate x^n in $O(\log_2 n)$ time, where x is real and n is a non-negative integer. (2.5)

Q4 a) Write algorithm for finding maximum and minumum element stored in an array of n integers by using divide and conquer approach. Show that T(n)=3n/2-2, where T(n) is the number of element comparisions. (2.5)

b) Write the PARTITION() algorithm of Quick-Sort. Describe in a step by step process to get the pass1 result by applying PARTITION() algorithm by taking 6th element (underlined) as pivot on the following array elements. (2.5)

5, 4, 3, 6, 2, 5, 9, 8, 7, 3, 6, 1

Q5 Write an algorithm to find the majority element in an array in O(n) time. The majority is an element that occurs for more than half of the size of the array. For example, the number 2 in the array {1, 2, 3, 2, 2, 2, 5, 4, 2} is the majority element because it appears five times and the size of the array is 9. (5)

==========XXXXXXXX==========