**Birla Institute of Technology & Science, Pilani**

**Work-Integrated Learning Programmes Division**

**Second Semester 2017-2018**

**Mid-Semester Test**

**(EC-2 Regular)**

Course No. : SS ZG519

Course Title : DATA STRUCTURES AND ALGORITHMS DESIGN

Nature of Exam : Closed Book

No. of Pages = 1

# No. of Questions = 10

Weightage : 30%

Duration : 2 Hours

Date of Exam : 03/03/2018 (AN)

Note:

1. Please follow all the *Instructions to Candidates* given on the cover page of the answer book.
2. All parts of a question should be answered consecutively. Each answer should start from a fresh page.
3. Assumptions made if any, should be stated clearly at the beginning of your answer.
4. What is an efficient algorithm? [3]
5. Solve the following recurrences by giving tight θ-notation bounds

T(n) =3T(n/5) + log2n [3]

1. If we are given two traversal sequences, can we construct the binary tree uniquely? [3]
2. Say you have an array containing information regarding n-people. Each person is described using. A string (their name) and a number (their position along a number line), each person has three friends, which are the three people whose number is nearest their own describe an algorithm to identify each person’s three friends. [3]
3. Find the complexity of the recurrence: T(n) = [5]



1. Draw the AVL-Tree that result from inserting the keys 4,1,2,3,9,5,7,15 in that order into an initially empty AVL-tree, you are request to show the intermediate and finial tree and give a post-order traversal of your finial tree. [5]
2. Explain how you can utilize a minimum heap to sort the list of numbers in descending order. Let n be the number of elements in the list. What is the complexity if your sorting algorithm? [5]

[Justify the following statement, wither valid or Invalid, with reasoning]

1. A complete binary tree with a height of h can have more nodes than a full binary tree with a height of h. [1]
2. When we use a max-heap to implement a priority queue, the time complexity of both the add and delete operations are O(n) [1]
3. Given n-integers a1…..an, the third smallest number among a1…..an can be computed in O(n) time [1]

\*\*\*\*\*\*\*