Data Structures (IT205) 2014-15
Second Midsemester-semester Exam
7^{th} October, 2014

Time: 2 hours **marks: 60**

This question paper consists of 3 questions printed on a single page. Check that your question paper is complete. Each question is worth 20 marks.

1. Draw the binary search trees which result after each operation in the following sequence, starting initially from an empty search tree. Insert 5, Insert 10, Insert 63, Insert 8, Insert 4, Delete 5, Insert 6, left-rotate at root node. For deleting a node with two children use the method of replacing it with its **successor**.

2. Consider the array of elements 8, 10, 1, 3, 5, 6, 4, 2, 9, 7. Build a heap from them using the heapify algorithm (for loop from $\lfloor \frac{n}{2} \rfloor$ down to 1) and also using the insert algorithm (for loop from 2 to n). Do you get the same heap or different heaps by these methods?

3. When we use the search and insert as leaf algorithm for a binary search tree with distinct keys we know the resulting tree depends on the order of insertions. Consider any binary search tree T on n nodes with distinct keys $\{1, \ldots, n\}$. In general more than one order of inserts can result in the same tree.

   (a) Which of the following insertion orders would result in the same tree T.
   - in-order tree walk of T
   - pre-order tree walk of T
   - post-order tree walk of T
   - level-order traversal of T
   - zig-zag traversal of T

   Justify your answers.

   (b) Given the tree T and an order O of the input keys (permutation of $\{1, \ldots, n\}$) insertion in order O will result in tree T if and only if each node appears in the order _____ [before/after] all its _____ [ancestors/descendants] in the tree. Fill in the two blanks with the appropriate choice from the adjoining brackets. You need to get the correct combination for the two blanks together. **Justify your answer with an explanation**.

   (c) Write an algorithm/pseudocode to compute the number of orderings which will result in a particular tree. You may assume that you are given as input the tree structure including the root and parent, left-child and right-child pointers at each node. The rank of each node is also available and need not be computed. Your algorithm needs to return the number of orderings leading to the input tree. What is the running time of your algorithm?