Assignment 3

Due date: 10:00am, Wednesday, the 31st of August 2016.
This assignment is worth of 7% of the total course mark.

General Instructions

You have to do it **individually**.

Submissions have to include coversheet including names, student ids, and your tutorial groups such that submissions can get marked.

Hand in your solutions to the box "ADSA" on level 4, Ingkarni Wardli (close to reception) **by the deadline**. **No late submissions will be accepted.**

Exercise 1 *Runtime analysis (15 points)*

Assume you are given a list of $n = 2k + 1$ integers as $(k + 1, 1, \ldots, k, k + 2, \ldots, 2k + 1)$. Draw a binary search tree resulted from inserting these integers in the order of the list. What is the best, worst and average time for the searching operation.

Exercise 2 *Algorithm design (10 points for 2.1 + 10 points for 2.2)*

1. Present an $\mathcal{O}(k \log k)$ time algorithm to return the k^{th} minimum element in a binary min-heap H of size n, where $1 \leq k \leq n$.
2. Using a binary min-heap data structure, propose an algorithm to find k smallest elements in an unsorted array. Provide and explain the complexity of your algorithm.

Exercise 3 *Binary Search and Priority Queues (10 points for 3.1 + 10 points for 3.2 + 5 points for 3.3)*

1. Read Chapter 6.2 in the book of Mehlhorn and Sanders. Briefly explain the advantages conferred by using Fibonacci heaps over the simpler binary heap implementation of priority queues (75-150 words).
2. Assume you are given a binary heap containing $n = 2^k$ elements in an array of size n. You are asked to repeatedly extract the minimum element, n times. Assume that insertions are never performed. To make the heap space efficient, you move the heap over to an array of size 2^j whenever an extraction decreases the number of elements to 2^j for any integer j. Suppose that the cost of each such move is $\Theta(2^j)$. What is the total movement cost caused by the n extracting operations starting from the heap of n elements? (Ignore the costs of any other binary heap related operations.)

3. Describe how you would, systematically, construct input cases of size n that would always cause the algorithm for binary search in Fig 2.5 of Melhorn to perform its maximum number of $2 + \lfloor \log n \rfloor$ comparisons. That is, given n come up with a way to generate input data for binary search that maximises the number of comparisons.

Exercise 4 *Binary Search and AVL Trees (15 points for 4.1 + 10 points for 4.2 + 10 points for 4.3 + 5 points for 4.4)*

1. Given n distinct input values *derive* an expression for the number of different degenerate trees that can be generated from different insertion orders of those n values. In your discussion, call this number $\deg(n)$. If you are stuck you may want to consider how to generate all the degenerate trees for the elements $1, 2, 3, 4$ to help you get started. **Note:** to get full marks your derivation has to be in precise language and apply to all integers $n \geq 0$. **Hint:** the answer to this question does not have to be long but it should address the nature of the insertion sequences that generate degenerate trees.

2. Draw a sequence of diagrams showing the insertion of the values:

   `[11,4,2,7,19,9,20]`

   into an empty AVL tree. You must:

   - Show the resulting tree immediately after each insertion step (that is *before* any balancing has taken place).
   - Indicate the node(s) at which each rotation is performed.
   - Where there is a double rotation, show the tree after each single rotation.
   - Show the resulting tree after balancing operation(s).

3. Draw a sequence of diagrams showing the **deletion** of the values:

   `[20,9,13,7,2,4,11]`

   from the tree formed in the question above. The deletions must occur in the order given. Again, show the tree after each deletion and rotation (if any).

4. What order should we insert the elements $\{1, 2, \ldots, 7\}$ into an empty AVL tree so that we don not have to perform any rotations on it?

End of Questions