This exercise sheet is the first of three sheets that allow you to earn bonus points for the exam. In particular, you earn 2% of the total exam score as bonus points if you achieve at least 8 points from the following exercises. Please do not forget to justify your answers, when requested.

Exercise 4.1  Statements about $O$-notation.

Indicate which of the following statements are correct, and justify your decision.

a) $\log_2(n) \in \Omega(\sqrt{n})$

b) $n^2 + n \in O(1.1^n)$

c) $n! \in O(n^n)$

d) $n \log n \in \Theta(n^2)$

Exercise 4.2  Questions about sorting algorithms.

Answer the following questions and give a brief explanation of your response.

a) Is the sorted sequence 1, 2, ..., n a Min-Heap?

b) When all the elements in a Max-Heap are different, at which positions could the smallest element be found?

c) A comparison-based algorithm is called stable if the relative order of identical elements is not changed. If the element '5', for example, occurs twice in an array, then the first 5 is never moved past the second 5. Which of the comparison-based sorting methods that you know are stable, or can easily be adapted accordingly?

d) A sorting algorithm is called in-situ if it works on the input sequence using only a constant amount of additional space for storing parts of the sequence. Which sorting algorithms that you know are in-situ, or can easily be adapted accordingly?

e) The worst-case running time of Quicksort is $\Theta(n^2)$, while the worst-case running time of Mergesort is $\Theta(n \log n)$. Guess two (good) reasons why, despite this fact, Quicksort is the more popular solution in practice.

Exercise 4.3  Heapsort.

Given the array [A,L,G,O,R,I,T,H,M,U,S]. Using Heapsort, we want to sort the array in ascending alphabetical order.

a) In the lecture, a method was presented that constructs a heap from an array in linear time. Describe the heap that is obtained when this method is used to the array given above.

b) Now sort the above array in ascending order using Heapsort. After each intermediate step in which a key is moved to its final position, write the current content of the array.

Exercise 4.4  Search trees.

a) Add the keys 5, 9, 11, 8, 7, 17, 15, 20 in this order into an initially empty binary search tree, and then draw the resulting tree. Next, delete the key 9 from the obtained tree, and
draw the resulting tree.

b) Draw the result if the key from part a) are inserted into an initially empty AVL tree. Draw also the resulting tree if after all the inserts, the key 9 is deleted.

**Hand-in:** until Wednesday, 20th March 2013.