Lecture 3 of Analysis of Algorithms: Exercises

1. Draw the search tree that we get if we insert the following values in this order into an initially empty binary search tree: $5, 8, 11, 6, 3, 1, 17, 2, 7$.

2. Now remove 5, what does the search tree look like after the deletion?

3. What is the maximum number of values that can be stored in a binary search tree of height h (expressed as a function of h)?

4. What is the minimum number of values that can be stored in a binary search tree of height h (expressed as a function of h)?

5. Write a recursive method that takes the root of a binary search tree and stores in each node the height of that node. How much time does your method take, in $\Theta(..)$-notation, if the tree stores n values?

6. Draw the AVL tree that results when we insert the following values: $6, 16, 11, 15, 14, 13, 12$.

7. Now delete 15 from the AVL-tree you obtained and draw the resulting AVL-tree.

8. Suppose we want to store a *multi-set*, a set where each value can be present multiple times. For example, $\{2, 3, 3, 3, 7, 9, 9\}$ is a multi-set. If we delete 3, the resulting set still has two occurrences of 3.

   Suppose we want to store a multi-set with n values in an AVL-tree in such a way that insert, delete and search operations take $O(\log m)$ time, where $m \leq n$ is the number of *different* values in the multi-set. Suggest how to adapt the AVL-tree, and verify that this time bound for the operations can be attained.

9. Write pseudo-code that takes the root of a binary search tree and two values x and y, and reports all values stored in the tree whose value lies between x and y.