Universität Freiburg
Institut für Informatik
Prof. Dr. G. Lausen
Dr. Fang Wei

Georges-Köhler-Allee, Geb. 051
D-79110 Freiburg i. Br.
Freiburg, April 21, 2009

Theory I - Exercise sheet 1

Submission due Tuesday, May 5.
Questions concerning the exercises can be directed to Thomas Hornung (*hornungt AT informatik.uni-freiburg.de*).

Exercise 1:

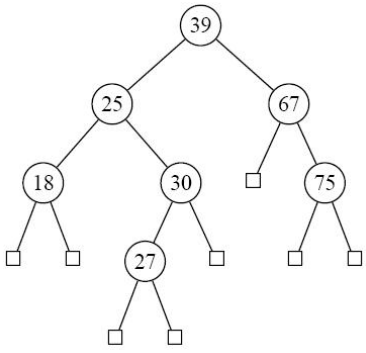
a) A node in a binary tree can be described by the following Java class:

```
class TreeNode {
    int key;           // key value of the node
    TreeNode left;     // reference to the left child
    TreeNode right;    // reference to the right child
}
```

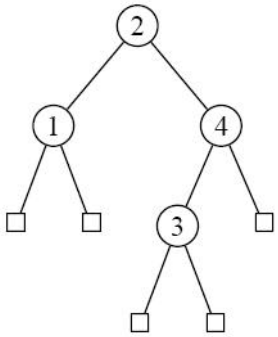
Here, **key** contains the key of the node, **left** and **right** point to the left and right child of the node (or *null*, if no such child exists).
We want to traverse a binary tree in *postorder*. Complete the code of method **traversePostorder(TreeNode n)**, which – called with the root node as initial argument – recursively outputs the keys contained in the tree in postorder. (For output, use **System.out.println(...)**.)

```
static void traversePostorder(TreeNode n) {
    ...
}
```

b) List the keys contained in the following tree in *preorder* and in *inorder*:



Exercise 2:



Consider the above binary search tree with four internal nodes.

a) Which sequences (permutations) of the keys $1, 2, 3, 4$ will result in this shape of the tree, if the keys are inserted one after the other into the initially empty tree?
Hence, if the keys are inserted in random order, what is the probability that this tree will result (assuming that all permutations are equally likely to be chosen)?

b) Draw all *structurally different* binary trees with four internal nodes. (You don't have to draw the leaf nodes.)

Exercise 3:

a) Let $N \geq 2$. Derive the recursive formula (1) for the number B_N of structurally different binary trees with N internal nodes:

$$B_N = B_0 \cdot B_{N-1} + B_1 \cdot B_{N-2} + \cdots + B_{N-1} \cdot B_0 \tag{1}$$

(Just for completeness: you can assume that $B_0 = B_1 = 1$.)

b) Show that B_N is also the number of possibilities to bracket a product of N factors (in an associative ring).

Exercise 4:
The *internal path length* $l(t)$ of a search tree t has been defined as follows:

$$l(t) = \begin{cases} 0 & \text{if } t \text{ is empty} \\ l(t_l) + l(t_r) + size(t) & \text{otherwise} \end{cases}$$

where t_l is the left and t_r the right subtree of the root of t and $size(t)$ is the number of internal nodes in t.

a) What is the internal path length of the tree shown in Exercise 1b)?

b) Using induction, show that

$$l(t) = \sum_{p \text{ internal node of } t} (depth(p) + 1)$$

where $depth(p)$ is the distance of node p from the root of t.

Your proof should proceed as follows:
Start with the base case and show that the equation is true for the empty tree.
Then show that for any non-empty tree t, the equation is true if we assume that it is true for the two subtrees t_l and t_r.

Hint: Notice that the function $depth(p)$ has different values depending on which tree you are looking at! You may want to make this clear by using different function names (e.g. $depth_t$, $depth_{t_l}$ and $depth_{t_r}$) for the different trees.