Submission: 3. Dec. 2009, 4 p.m.

- The solutions can be submitted in English or German
- You are required to submit your own solution.
- You are allowed to discuss your solutions with each other. Nevertheless, you are required to write down the answers in your own words.

**Exercise 3.1 - Treaps**  
[Points: 1+1+1+1]

1. Insert the keys $f,g,h,e,b,a,c$ into an initially empty treap. The priorities of these keys are given in the table below.

<table>
<thead>
<tr>
<th>keys</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>e</th>
<th>f</th>
<th>g</th>
<th>h</th>
<th>i</th>
<th>j</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>prio</td>
<td>8</td>
<td>15</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>25</td>
<td>22</td>
<td>19</td>
<td>13</td>
</tr>
</tbody>
</table>

2. Delete $e$ from the treap.
3. Insert the keys $i,k,j$ into an other empty heap. Merge the treap resulting from 2. with the treap from this part.
4. Perform $\text{Spalte}(T,d,T_1,T_2)$ where $T$ is the treap obtained in part 3.

For each of the previous insert, delete, and split (Spalte) operations, illustrate the state of the treap before and after every rotation.

**Exercise 3.2 - Treaps**  
[Points: 3+3]

1. The left spine of a binary search tree $T$ is the path from the root to the node with the smallest key. In other words, the left spine is the longest path from the root that consists only of left edges. Symmetrically, the right spine of $T$ is the path from the root to the node with the maximum key. The length of a spine is the number of nodes it contains (including the root).

Consider a Treap $T$ immediately after inserting node $x$. Let $C$ be the length of the right spine of the left subtree of $x$. Let $D$ be the length of the left spine of the right subtree of $x$. Prove that the total number of rotations that were performed during the insertion of $x$ is equal to $C + D$.

2. Let $S = \{x_1, \ldots, x_n\}$ where $x_i < x_j$ if $i < j$ and let $T_S$ be a treap which contains the $x_i$’s as nodes. The priorities $\text{prio}(x_i)$ are chosen uniformly at random from $(0,1)$.

Prove that searching for a node $x \notin S$ in $T_S$ has an expected runtime of $H_m + H_{n-m}$, where $x_m < x < x_{m+1}$.

**Exercise 3.3 - Universal hashing**  
[Points: 5]

Let $U = \{0, \ldots, N-1\}$, where $N$ is a prime and $m = 4$. Let $a_i = 40 \cdot i$ and $b_i = 60 \cdot i$.

Now consider the following class of hash functions.

$$H = \{ h_i(k) = ((a_i \cdot k + b_i) \mod N - 1) \mod m \} \text{ for } i \in \{1, \ldots, N(N-1)\}$$

Is $H$ universal? Prove your answer. If $H$ is not universal, then modify $h_i$, $a_i$ and $b_i$, so that the resulting class becomes universal.
Exercise 3.4 - Perfect hashing

Let $U = \{0, \ldots, 30\}$ and $S = \{2, 4, 7, 11, 12, 18, 19, 21, 26, 28\}$.

1. Use the two-level scheme described in the lecture to build a perfect hash function with $k = 3, N = 31, n = |S| = 10$. For $i = 0, \ldots, n - 1$ determine the values $W_i, b_i, m_i, k_i$, and $h_{k_i}$.

2. For each element of $S$ quote the position in the hash table at which this element is stored.