Homework Assignment 6

Due Date: March 28, 2021, 23:59

Note. Please note that this semester all assignments are group assignments. Further note that for the grading we will apply a "10%" rule, i.e. the maximum number of points for this assignments is 55, but 50 will be counted as 100%. Points that exceed 50 will be stored in a separate counter and used later for compensation of lost points in other assignments or (if not used up this way) the final exam.

Exercise 1.

(i) Implement a variant of linear probing without using __Placeholder objects. Instead use the alternative *delete* operation handled in the lecture to move another element into the freed space.

(ii) Implement a variant of linear probing, where the size of the hash table is not m, but $m+m'$. In the search procedure in *insert*, *find* and *delete* increment the search index not considering the residue modulo m.

total points: 13

Exercise 2. The *delete* operation on binary search trees presented in the lectures merges the two successor trees, when an element labelling a non-leaf vertex is deleted. One might as well re-insert all the elements.

(i) Implement this alternative for the implementation of *delete*.

(ii) Discuss, why this idea should not be not recommended.

total points: 11

Exercise 3. Sketch interative implementations of *insert* and *delete* operations on AVL trees, and discuss the differences to the provided recursive implementations.

total points: 12

Exercise 4. Define that an AVL tree satisfies the *median property* iff for all vertices v the label $\ell(v)$ is an element with minimal distance to m, where m is the median of all elements in the AVL subtree rooted at v.

(i) Show that if an AVL tree satisfies the median property, then it is perfectly balanced.

(ii) Define and implement a modified *insert* operation that guarantees that the resulting AVL tree satisfies the median property.

(iii) Define and implement a modified *delete* operation that guarantees that the resulting AVL tree satisfies the median property.

(iv) Determine the worst case complexity for these modified *insert* and *delete* operations.

total points: 19