Marburg University

Department of Computer Science
Prof. Dr. Sebastian Wild

Date: 2025-10-31
Version: 2025-10-31 08:46

Exercise Sheet 3 for Effiziente Algorithmen (Winter 2025/26)

Hand In: *Until 2025-11-07 18:00, on ILIAS.*

Problem 1

20 + 20 + 20 points

a) Given a set of keys $\{2, 5, 6, 11, 17, 18, 22\}$, draw four valid binary search trees of heights 2,3,5, und 6, respectively.

b) In an *extended binary search tree*, the number of children of a node is always 0 or 2, so unary nodes allowed. Prove using (structural) induction: The number of nodes (internal and leaves) in an extended binary search tree is odd.

Problem 2

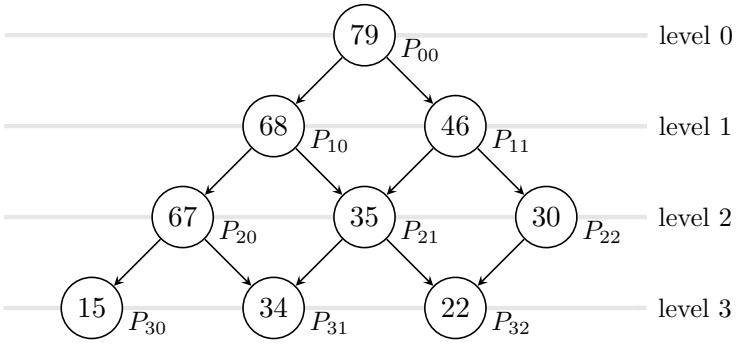
30 points

Design a data structure that supports the following Operations in *constant time* and using constant words of memory per stored element: Push, Pop, GetMin. Justify why your idea achieves the claimed complexities.

Problem 3

10 + 20 + 20 + 30 points

A *pyramid* is a data structure that can be made into an implementation of the ADT priority queue. An example with 9 nodes is shown below:



A pyramid has the following properties:

(i) **Structure:** A pyramid consists of $\ell \geq 0$ *levels*. The ith level, for $0 \leq i < \ell$, contains at most $i + 1$ nodes, which we refer to as $P_{i,0}, \ldots, P_{i,i}$.

Apart from the last level, every level is *full*. The last level must be "flush-left", i.e., all nodes are far left in the level as possible.

(ii) **Order:** Every node $P_{i,j}$ has at most two *children:* $P_{i+1,j}$ and $P_{i+1,j+1}$ (if these exist).

The priority of any node is at least as large as the priority of any of its children.

You can assume that our pyramids only handle pairwise different priorities.

a) Show that a pyramid with n nodes has height $\Theta(\sqrt{n})$.

b) Assuming you can access the attributes *key*, *leftChild*, *rightChild*, *leftParent*, and *rightParent* of a node in $O(1)$ time, give a pseudocode implementation of *deleteMax* for pyramids.

The running time of your algorithm must be linear in the height of the pyramid.

Hint: Revisit deleteMax in Heaps.

c) Develop a pseudocode implementation for *insert* in pyramids. The running time of your algorithm must be linear in the height of the pyramid.

d) Operation *contains*(x) takes a priority x and determines whether or not a priority queue contains an element of priority x.

In a binary heap, contains needs to search the entire heap in the worst case. In pyramids we can implement *contains* must more efficiently:

Describe an implementation for *contains* in pyramids with a running time in $O(\sqrt{n} \log n)$. You can assume a function $\texttt{get}(i, j) = P_{i,j}$, returning any node by coordinates in constant time.