Marburg University

Department of Computer Science
Prof. Dr. Sebastian Wild

Date: 2025-10-31
Version: 2025-10-31 08:41

Übungsblatt 3 zur Vorlesung Effiziente Algorithmen (Winter 2025/26)

Abgabe: *Bis 2025-11-07 18:00, on ILIAS.*

1. Aufgabe

20 + 20 + 20 Punkte

a) Gegeben sei die Menge von Schlüsseln $\{2, 5, 6, 11, 17, 18, 22\}$. Zeichnen Sie je einen gültigen binären Suchbäum mit Höhe 2, 3, 5 bzw. 6.

b) Bei einem *erweiterten Binärbaum* ist die Anzahl von Kindern entweder 0 oder 2, d. h. es darf keine unären Knoten geben. Zeigen Sie durch (strukturelle) Induktion: Die Anzahl Knoten (Blättere und innere Knoten) in einem erweiterten Binärbaum ist ungerade.

2. Aufgabe

30 Punkte

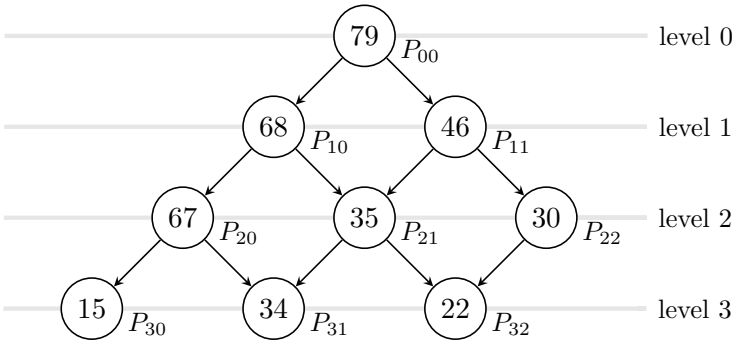
Entwerfen Sie eine Datenstruktur, welche die folgenden Operationen in konstanter Zeit und mit konstantem Speicherbedarf pro Element unterstützt: Push, Pop, GetMin.

Begründen Sie, warum Ihre Lösung die behaupteten Komplexitäten einhält.

3. Aufgabe

10 + 20 + 20 + 30 Punkte

Eine *Pyramide* ist eine Datenstruktur, welche die Basis für eine Implementierung des abstrakten Datentyps Priority Queue bilden kann. Im Folgenden ist ein Beispiel für eine Pyramide mit 9 Knoten gezeigt:



Eine Pyramide hat die folgenden beiden Eigenschaften:

(i) **Struktur:** Eine Pyramide besteht aus $\ell \geq 0$ Levels. Das i-te Level, für $0 \leq i < \ell$, enthält höchstens $i + 1$ Einträge, welche als $P_{i,0}, \ldots, P_{i,i}$ bezeichnet werden.

Bis auf potentiell das letzte Level ist jedes Level vollständig gefüllt. Das letzte Level ist "linksbündig", d. h. Knoten sind von links nach rechts befüllt.

(ii) **Ordnung:** Jeder Knoten $P_{i,j}$ hat höchstens zwei *Kinder:* $P_{i+1,j}$ und $P_{i+1,j+1}$ (falls diese Knoten existieren). Die Priorität eines Knotens ist immer mindestens so groß wie die Priorität seiner Kinder (falls existent).

Sie dürfen annehmen, dass alle Elemente einer Pyramide paarweise verschieden sind.

a) Zeigen Sie, dass eine Pyramide mit n Knoten die Höhe $\Theta(\sqrt{n})$ hat.

b) Angenommen sie können auf die Attribute *key*, *leftChild*, *rightChild*, *leftParent*, and *rightParent* eines Knotens in $O(1)$ Laufzeit zugreifen.

Geben Sie eine Implementierung in Pseudocode für die *deleteMax*-Operation in Pyramiden an. Die Laufzeit Ihres Algorithmus muss linear in der Höhe der Pyramide sein.

Tipp: Orientieren Sie sich an der deleteMax-Operation von Heaps.

c) Geben Sie eine Implementierung in Pseudocode für die *insert*-Operation in Pyramiden an. Die Laufzeit Ihres Algorithmus muss linear in der Höhe der Pyramide sein.

d) Die Operation *contains*(x) nimmt eine Priorität x entgegen, und gibt zurück, ob eine Priority Queue einen Schlüssel mit der Priorität x enthält.

In einem binären Heap muss im Worst Case der gesamte Heap durchsucht werden. In Pyramidem lässt sich die *contains*-Operation effizienter implementieren.

Beschreiben Sie eine Implementierung für *contains* in Pyramidem mit einer Laufzeit von $O(\sqrt{n} \log n)$. Für Ihre Implementierung dürfen Sie annehmen, dass es eine Funktion $\texttt{get}(i, j) = P_{i,j}$ gibt, welche einen Knoten mit den gegebenen Koordinaten in $O(1)$ Laufzeit zurückgibt.