

# Theoretische Informatik 1

## 9. Übung

Schreiben Sie Ihren Namen in den Dateinamen der von Ihnen abgegebenen Datei.

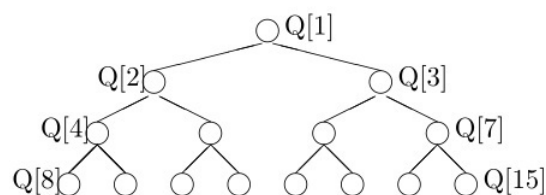
**Abgabe:** Lösen Sie entweder Aufgabe 1 oder Aufgabe 2. Ihre Lösung senden Sie bitte bis zum Donnerstag, dem 14.12.23, 7:30 Uhr, per E-Mail an [knut.odermann@informatik.tu-chemnitz.de](mailto:knut.odermann@informatik.tu-chemnitz.de), am besten als pdf-Datei (idealerweise mit einem Textsatzsystem wie LaTeX erstellt) und gut lesbar. Nicht akzeptiert werden Scans bzw. Photos von Quellen in Formaten größer als DIN-A4. Vermerken Sie auf Ihrer Abgabe Ihren Namen, Ihre Matrikelnummer und Ihren Studiengang.

### Aufgabe 1 [10 Punkte]

Zeigen Sie, dass

- (a) der folgende Algorithmus die Laufzeit  $O(n)$  hat und
- (b) der durch den Algorithmus erzeugte „Heap“ tatsächlich alle Bedingungen des Heaps erfüllt.

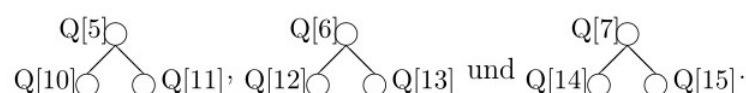
Gegeben sei ein Feld  $Q[1..n]$  von Elementen. Wir wollen aus  $Q$  einen *Heap* aufbauen. Wir nehmen  $n = 2^k - 1$  an und betrachten  $Q$  als Baum (z. B.  $k = 4$ ):



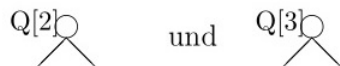
Der Heap wird nun *bottom-up* aufgebaut. Das heißt, zuerst wird in dem Teil



die Heap-Eigenschaft hergestellt, indem  $Q[4]$  an die richtige Stelle sickert. Genauso verfahren wir mit den Teilen



Danach lassen wir  $Q[2]$  bzw.  $Q[3]$  an die richtige Stelle sickern, um in den Teilbäumen



die Heapeigenschaft herzustellen. Schließlich entsteht durch Sicken von  $Q[1]$  ein korrekter Heap.

*Hinweis:* Überprüfen Sie, wie oft ein Knoten aus Ebene  $i$  maximal sickern kann, bis er an der richtigen Stelle steht. Summieren Sie diesen Wert über alle Knoten. Die entstehende Summenformel kann auf eine geometrische Reihe zurückgeführt werden, wenn man die Ungleichung  $i < 1, 5^i$  benutzt.

### Aufgabe 2 [10 Punkte]

Sortieren Sie mit dem Heapsort-Algorithmus die Zahlenfolge (6, 2, 9, 1, 3, 10, 8). Geben Sie alle Zwischenschritte an.

### Aufgabe 3

Beim Löschen der Wurzel im Heap muss erst der Wurzelknoten mit dem letzten Element getauscht werden und dann muss das (jetzt oben stehende) letzte Element wieder heruntersickern.

Überlegen Sie sich, warum man nicht direkt die Wurzel löschen kann und die entstehenden Leerstellen durch „Hochrücken“ von Kindknoten füllen kann.

### Aufgabe 4 [Zusatzaufgabe]

Implementieren Sie den Heap mit Knotenindex aus der Vorlesung, wie er für Dijkstra gebraucht wird.

Es werden also drei Arrays mit einer Länge/Kapazität von  $|V|$  gebraucht. Ein Array für den Heap, ein Array für die Schlüsselwerte und ein Array für den Index.

Bei dem Dijkstra-Algorithmus muss jeweils nur der Wurzelknoten gelöscht werden. Überlegen Sie sich trotzdem, wie Sie auch andere Knoten im Heap löschen können