

Algorithmen und Datenstrukturen

Aufgabe 1 (AGS 6.1.10)

Wenden Sie den Quicksort-Algorithmus auf die Folge 4, 7, 6, 2, 9 an. Die Zahlen sollen aufsteigend sortiert werden. Dokumentieren Sie den Rechenablauf, indem Sie

- das Pivot-Element jeder Teilfolge kennzeichnen und
- die Teilfolgen und Stellung der Indizes i, j jeweils
 - unmittelbar vor und nach jedem Tausch von Elementen, sowie
 - unmittelbar vor und nach jedem rekursiven Aufruf angeben.

Aufgabe 2 (AGS 6.2.12)

Wenden Sie den Heapsort-Algorithmus auf die Folge 2, 0, 9, 3, 5, 8, 4, 1, 6, 7 an.

Dokumentieren Sie dazu in der Phase 1 das schrittweise Herstellen der Heap-Eigenschaft und dabei insbesondere die Veränderungen durch die Funktion „sinkenlassen“. Das Sinkenlassen in mehreren *unabhängigen* Teilbäumen darf in einem Schritt protokolliert werden.

In der Phase 2 brauchen Sie nur zwei Sortierschritte auszuführen. Ein Sortierschritt besteht aus einem Tausch- und einem Sinkenlassen-Schritt, die jeweils einzeln zu notieren sind.

Zusatzaufgabe 1 (AGS 6.1.1 ★)

Wenden Sie den Quicksort-Algorithmus auf die Folge 9, 5, 4, 2, 3, 8, 1 an.

Zusatzaufgabe 2 (AGS 6.2.18 ★)

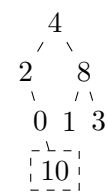
Wenden Sie den Heapsort-Algorithmus auf die Folge 4, 10, 11, 3, 8, 13, 28, 29, 19, 21 an. In Phase 2 brauchen Sie nur zwei Sortierschritte auszuführen.

Zusatzaufgabe 3 (AGS 3.2.44 a,b ★)

Gegeben sei die folgende Typdefinition für binäre Bäume:

```
typedef struct node *tree;
struct node { int key; tree left, right; };
```

- (a) Die *Prominenz* eines Binärbaumes t ist die Anzahl der Knoten auf dem Pfad von der Wurzel von t bis zum Blatt in t , das am weitesten links steht (in der Abbildung rechts ist dieses Blatt markiert). Schreiben Sie eine Funktion `int prom(tree t)`, welche die Prominenz des übergebenen Baumes t berechnet. Für $t == \text{NULL}$ soll `prom(t) == 0` gelten. Für den Baum in der rechten Abbildung soll die Funktion 4 zurückgeben.



- (b) Ein binärer Baum t heißt *Rechtsbaum*, falls für jeden Knoten v von t die Prominenz des rechten Teilbaums von v (`right`) größer oder gleich der Prominenz des linken Teilbaums von v (`left`) ist. Ein Baum, der keine Knoten besitzt, ist also trivialerweise ein Rechtsbaum. Schreiben Sie eine Funktion `int isRightist(tree t)`, welche für einen Baum t als Eingabe den Wert 1 ausgibt, falls t ein Rechtsbaum ist, ansonsten den Wert 0.