Algorithmen und Datenstrukturen

9. Übungsblatt

Zeitraum: 15. – 19. Dezember 2014

Übung 1 (AGS 3.2.19)
Gegeben seien die folgenden Vereinbarungen für ternäre Bäume:

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

Implementieren Sie eine rekursive Funktion `count`, die für einen beliebigen Baum des oben beschriebenen Typs die Anzahl derjenigen Knoten zählt, die entweder genau zwei Nachfolgerknoten besitzen oder gar keinen.

Übung 2 (AGS 6.1.6)
Gegeben sei die Folge 8, 5, 4, 7, 3. Wenden Sie auf diese Folge den Quicksort-Algorithmus an!

Übung 3 (AGS 6.4.2)
Wenden Sie auf die Folge 32, 6, 21, 5, 23, 11, 17, 8, 3, 12, 10 den Heapsort-Algorithmus an. Dokumentieren Sie beide Phasen des Algorithmus, insbesondere in der ersten Phase das schrittweise Herstellen der Heap-Eigenschaft, sowie in der zweiten Phase jeden Doppelschritt (bestehend aus Austauschen und Sinkenlassen).

Übung 4 (AGS 7.3)
Geben Sie zu dem Pattern `a b a a b a b b` die mit Hilfe des Knuth-Morris-Pratt-Algorithmus berechnete Verschiebetabelle an.

Zusatzaufgabe 1 (AGS 7.1 ⋆)
Geben Sie zu den Pattern
(a) `a b a a b a a a b`
(b) `a a a b a a a a`
die jeweils mit Hilfe des KMP-Algorithmus (Knuth-Morris-Pratt) berechnete Verschiebetabelle an.

Zusatzaufgabe 2 (AGS 6.4.1 ⋆)
Wenden Sie auf die Folge: 13, 6, 25, 4, 23, 11, 18, 9, 3, 19, 10, 7, 2 den Heapsort-Algorithmus an.

Zusatzaufgabe 3 (AGS 6.1.5 ⋆)
Gegeben sei die Folge: 6, 10, 4, 3, 7. Wenden Sie auf diese Folge den Quicksort-Algorithmus an!