Aufgaben zur Veranstaltung
Algorithmen und Datenstrukturen, SS 2021

H. Pflug, J. Dietel FH Aachen, Campus Jülich; IT Center, RWTH Aachen

Präsenzaufgaben 13 21./22.06.2021

Die Lösung der Aufgaben wird am Ende der Übung von Ihnen vorgestellt.

Aufgabe 1: Fragen

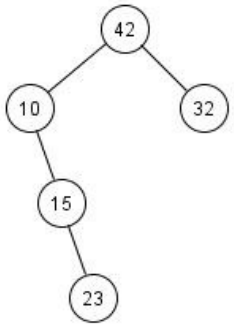
a) Welchen abstrakten Datentyp benötigt man beim Durchlauf eines Baums in Level-Order?

b) Welche Operationen können zum Einsatz kommen, wenn im B-Baum ein Element hinzugefügt wird?

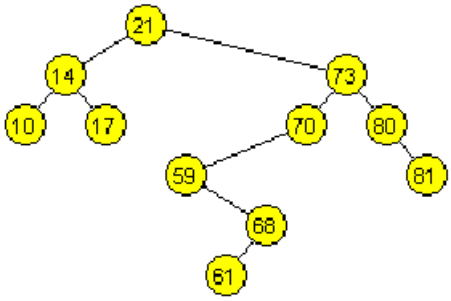
c) Welche Varianten zur Kollisionsbehandlung gibt es beim Hashing? Nennen Sie zwei Varianten.

d) Wie oft wird in einem AVL-Baum zum Löschen maximal rotiert?

e) Markieren Sie die Knoten im folgenden Baum, an denen die AVL-Eigenschaft verletzt ist.



f) Löschen Sie aus dem folgenden binären Suchbaum den Knoten 21.



Aufgabe 2: Gegeben sei das folgende Feld:

1	3	4	6	2	5	7

Sortieren Sie das Feld mit Heap-Sort.

a) Wenden Sie einen der in der Vorlesung vorgestellten Algorithmen an, um das Feld in einen Heap zu verwandeln. Geben Sie jeweils nach dem Einfügen einer Zahl den entstandenen Heap an, wahlweise als Feld (Tabellenvorlage) oder als Baum.

b) Sortieren Sie die Elemente durch Auflösung des Heaps. Geben Sie für jeden Zwischenschritt das aktuelle Feld an (einschließlich der bereits sortierten Elemente).
Tipp: Sie können sich zusätzlich als Hilfestellung auch jeweils den Heap in Baumform notieren.

Aufgabe 3: Gegeben sei die folgende Adjazenzliste (Gewichte jeweils in Klammern):

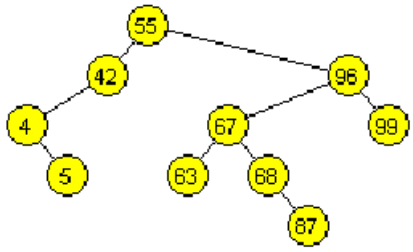
1: 2(7) 3(5) 4(2) 5(7) 6(8)
2: 1(8) 4(2) 6(9)
3: 1(9) 2(4) 5(1) 6(3)
4: 1(1) 2(1) 3(1) 5(3) 6(5)
5: 2(8) 3(4) 4(5) 6(1)
6: 2(5) 3(5) 4(2)

Bestimmen Sie die kürzesten Wege ausgehend von Knoten 1 nach dem Algorithmus von Dijkstra, indem Sie das folgende Schema ausfüllen.

		d						p				
V_i		2	3	4	5	6		2	3	4	5	6

Aufgabe 4:

Durchlaufen Sie den folgenden Baum in In-Order, Pre-Order und Post-Order und geben Sie jeweils die Reihenfolge der durchlaufenen Knoten an.



Aufgabe 5:

Binäre Suchbäume können unterschiedlich aufgebaut sein, auch wenn sie die gleichen Werte enthalten. Falls aber außer den Werten auch der Pre-Order-Durchlauf gegeben ist, ist ein binärer Suchbaum eindeutig.

Nehmen Sie die Klasse BinarySearchTree aus den vorigen Übungen und fügen Sie die folgenden Methoden und Konstruktoren hinzu:

```
//Erstellt den binaeren Suchbaum aus dem Preorder-Durchlauf
public BinarySearchTree(int[] preorderList)

//Gibt die Werte des Suchbaums in Preorder-Reihenfolge zurueck.
public ArrayList<Integer> getPreorderList()
```

Aufgabe 6:

Fügen Sie der Klasse TreeNode den copy-Konstruktor

```
public TreeNode(TreeNode orig)
```

hinzu. Der Copy-Konstruktor kopiert den Knoten orig mitsamt den darunter hängenden Teilbäumen.

Benutzen Sie nicht den Umweg über die Preorder-Darstellung, sondern kopieren Sie den Baum in einer einzigen (am besten rekursiven) Methode. Verwenden Sie Aufgabe 5, um Ihre Lösung zu testen.