Universität Stuttgart

Institut für Visualisierung
und Interaktive Systeme

Übungsblatt 4

Datenstrukturen und Algorithmen (SS 2018)

Abgabe: Mittwoch, 23.05.2018, 23:55 Uhr — Besprechung: ab Montag, 04.06.2018

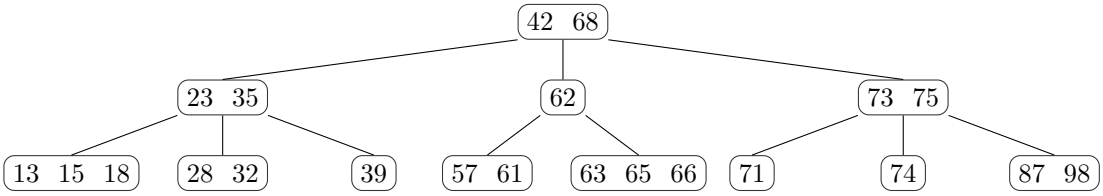
Bitte lösen Sie die Übungsaufgabe in **Gruppen von 3 Studenten** und wählen EINEN Studenten aus, welcher die Lösung im ILIAS als **PDF** als **Gruppenabgabe** (unter Angabe aller Gruppenmitglieder) einstellt. Bitte erstellen Sie dazu ein **Titelblatt**, welches die Namen der Studenten, die Matrikelnummern, und die E-Mail-Adressen enthält.

Die Aufgaben mit Implementierung sind mit Impl gekennzeichnet. Das entsprechende Eclipse-Projekt kann im ILIAS heruntergeladen werden. Bitte beachten Sie die Hinweise zu den Implementierungsaufgaben, die im ILIAS verfügbar sind.[1]

Dieses Übungsblatt beinhaltet 4 Aufgaben mit einer Gesamtzahl von 30 Punkten.

Aufgabe 1 2-3-4-Bäume, Rot-Schwarz-Bäume *[Punkte: 6]*

Gegeben sei der 2-3-4-Baum T_1:



(a) *(2 Punkte)* Fügen Sie in T_1 nacheinander die Werte *64* und *88* ein. Der daraus entstehende Baum sei T_2. Geben Sie Zwischenschritte an und erläutern Sie diese kurz.

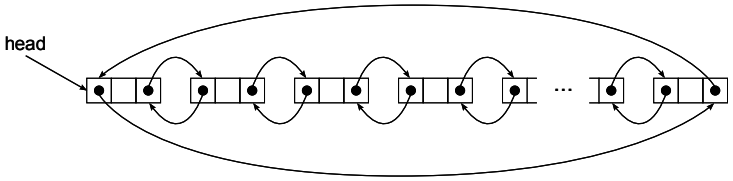
(b) *(2 Punkte)* Wandeln Sie T_2 in einen Rot-Schwarz-Baum um. Der daraus entstehende Baum sei T_3.

(c) *(2 Punkte)* Fügen Sie in T_3 den Wert *92* ein. Gehen Sie hierbei nach der Top-Down-Variante vor. Geben Sie Zwischenschritte an und erläutern Sie diese kurz.

Aufgabe 2 Impl Doppeltverkettete Listen *[Punkte: 8]*

(a) *(2 Punkte)* Implementieren Sie eine Knotenklasse `LinkedListNode`, die von einer doppelt verketteten Liste genutzt werden kann. Die Klasse muss das Interface `ILinkedListNode` implementieren und einen Standard-Konstruktor (d.h. Konstruktor ohne Parameter) enthalten.

(b) *(6 Punkte)* Implementieren Sie die Listenklasse `CircularLinkedList` für eine doppelt verkettete Liste, die intern Knoten vom Typ `ILinkedListNode` nutzt. Die Listenklasse muss das Interface `ICircularLinkedList` implementieren und einen Standard-Konstruktor (d.h. Konstruktor ohne Parameter) enthalten. Die Struktur der zu implementierenden Liste ist in der folgenden Abbildung dargestellt.



[1] `https://ilias3.uni-stuttgart.de/goto_Uni_Stuttgart_crs_1432415.html`

Beachten Sie, dass die Methode `public T get(int index)` des Interfaces `ICircularLinkedList` einen Parameter vom Typ `int` erwartet und das Element an der entsprechenden Position zurückliefert. Für Indizes, die größer sind als die Länge der Liste, wird die Indexposition in zirkulärer Weise durch Zurückkehren vom Listenende zum Listenanfang fortgesetzt. Für negative Indizes werden die Elemente ausgehend vom Ende der Liste in umgehkehrter Richtung zurückliefert (z.B. liefert -1 das letzte Element; -2 liefert das vorletzte Element, usw.).

Aufgabe 3 Impl Baum-Iteratoren *[Punkte: 12]*

Gegeben im Eclipse-Projekt sind die Schnittstellen `IBinaryTreeNode`, `IBinarySearchTree`, und `IBinarySearchTreeIterable` für einen Binärbaum inkl. Pre-, In-, Post- und Level-Order-Iteratoren. Die entsprechende Knotenklasse `BinaryTreeNode` ist vorgegeben und darf nicht modifiziert werden. Erweitern Sie die Klasse `BinarySearchTree`, damit sie die Schnittstelle `IBinarySearchTreeIterable` implementiert (ohne die bestehenden Methoden zu modifizieren).

Ergänzung: Die Methode `remove` der Iteratoren wird nicht unterstützt und sollte immer `UnsupportedOperationException` werfen.

Aufgabe 4 B-Bäume *[Punkte: 4]*

Fügen Sie in einen leeren B-Baum mit $m = 2$ der Reihe nach die Zahlen *21*, *60*, *25*, *70*, *63*, *66*, *69*, *33*, *39*, *51*, *34*, *42*, *50*, *72*, *80*, *82* und *73* ein. Geben Sie Zwischenschritte, in denen ein Split notwendig ist, an und erläutern Sie diese kurz.