

Aufgabe 8.2 a)

Wikipedia Bedingung Nr.3 für einen R-B-Baum :

"Jeder Pfad von einem gegebenen Knoten zu seinen Blattknoten enthält die gleiche Anzahl an Schwarzen Knoten"

Auch: Skript S.403, Folgerung für R-B-Bäume :

"Die Anzahl der schwarzen Knoten ist auf jedem Pfad von der Wurzel zu einem Blatt gleich."

b)

Ist ein Knoten rot, so sind beide Kinder schwarz, daher sind im Extrem-Fall auf kürzestem Pfad nur schwarze Knoten vorhanden. Damit und durch Bedingung Nr.3 aus Aufgabe a), kann der längste Pfad maximal doppelt so lang sein wie der Kürzeste. Also $L_{max} = 2 \cdot L_{min}$.

c)

Höhe nach oben begrenzt durch $h \leq 2 \cdot \lceil \log_2(n+2) \rceil - 2$

Umformen ergibt ein Baum der Höhe h hat minimal $2^{(0.5h+1)} - 2$ Knoten

Aufgabe 8.2d

X - einzufügender Knoten

1. Finde richtige Einfüge Position für X und füge X als roten Knoten dort an, wie bei einem Binären Suchbaum.
2. Wenn X die Wurzen (root) ist : ändere Farbe von X zu Schwarz.
3. Falls Vater und Onkel NICHT SCHWARZ sind oder X nicht die Wurzel ist:
 - a) Wenn der Onkel ROT ist :
 - Ändere Farbe von Vater und Onkel zu SCHWARZ
 - Färbe Großeltern als ROT
 - Wiederhole Schritte 2. und 3. für Großvater.
 - b) Wenn der Onkel SCHWARZ ist (wie beim AVL-Baum), unterteile in 4 Fälle:
 - (Left-Left-Case) X links vom Vater und Vater Links vom Großvater
 - Rechtsrotation des Großvaters
 - Vater und Großvater wechseln(tauschen) die Farbe
 - (Right-Left-Case) X rechts vom Vater, Vater links vom Großvater
 - Linksrotation des Vaters
 - Left-Left-Case ausführen
 - (Right-Right-Case) X rechts vom Vater, Vater rechts vom Großvater
 - Linksrotation des Großvaters
 - Vater und Großvater tauschen die Farben
 - (Right-Left-Case) X links vom Vater, Vater rechts vom Opa
 - Rechtsrotation des Großvaters
 - Right-Right-Case ausführen

Aufg. 8.3 a)

1.

$$[8, 12]$$

2.

$$[8, 12]$$

$$[15, 23]$$

3.

$$[12, 18]$$

$$[8, 12]$$

$$[15, 23]$$

4.

$$[12, 18]$$

$$[8, 12]$$

$$[15, 23]$$

$$[8, 11]$$

5.

$$[12, 18]$$

$$[8, 12]$$

$$[15, 23]$$

$$[5, 11]$$

$$[25, 28]$$

6.

$$[12, 18]$$

$$[5, 11]$$

$$[15, 23]$$

$$[1, 30]$$

$$[8, 12]$$

$$[25, 28]$$

7.

$$[12, 18]$$

30

$$[15, 23]$$

28

$$[25, 28]$$

$$[1, 30]$$

30

$$[8, 12]$$

12

$$[6, 9]$$

9

Aufgabe 8.3 b)

Größeres(a, b) liefert das größere Element von beiden zurück

berechneMax(Knoten N)

Falls N ein BlattKnoten (Kind Links UND Kind Rechts beide NIL-Knoten)

N.Maximum <- Obere Intervall-Grenze von N;

Sonst

G <- Größeres(berechneMax(N.Kind Links), berechneMax(N.KindRechts))

N.Maximum <- Größeres (Obere Intervall-Grenze von N, G)

Gebe N.Maximum zurück

Als Code:

```
1.  Class Node{
2.      private Node lKind = null;
3.      private Node rKind = null;
4.      private int dataMax = null; //eingenes Maximum
5.      private int uInt = null; //obere und untere Intervallgrenzen
6.      private int oInt = null;
7.      //die Werte die ein Node haben sollte
8.  }
9.
10.     int postOrderMaximum(Node n)
11.     if(n.rKind == null && n.lKind == null){
12.         n.dataMax = n.oInt;
13.     }else{
14.         n.dataMax = max(n.oInt, max(postOrderMaximum(n.lKind),
postOrderMaximum(n.rKind)));
15.     }
16.     return n.dataMax;
```

Aufgabe 8.3 c)

Der Algorithmus ist Korrekt und müsste fehlerfrei laufen (wobei T.root != null sein sollte)

Falls ein Intervall existiert in dem X vorkommt, wird dieses immer über die Maxima der Knotenpunkte gefunden. Auch wenn es kein solches Intervall gibt, so wird vom letzten überprüften Blatt ein Verweis auf seinen rechten NIL-Knoten aufgerufen und somit korrekterweise NULL ausgegeben.

Die Laufzeit ist hierbei **O(log(n+1))** weil auch im Worst-Case-Szenario die WHILE-Schleife höchstens so oft wie die Höhe des Baums aufgerufen wird, sprich dem längsten Pfad der Wurzel bis zu einem Blatt. +1 Weil der NIL-Knoten des letztens Blatts auch noch einen Aufruf erzeugt.

Aufgabe 8.3 d

Es müssen nur die Maxima auf dem Suchpfad geändert werden, da sich auch nach dem Einfügen nichts an den Maxima der benachbarten Teilbäume ändert. Am effizientesten wäre es das Maxima dynamisch während des Rotierens zu berechnen, also quasi im Algorithmus aus Aufgabe 8.2d bei dem Teilschritt 3a. Weil ein simpler vergleich nach jeder Rotation schon ausreichen würde um wieder überall das Korrekte Maximum zu berechen. Da wir wie zuvor von unten nach oben vorgehen, ist die Reihenfolge von unten nach Oben, den Suchpfad entlang auch am schnellsten.