Universität Leipzig
Institut für Informatik
Abteilung Datenbanken
Prof. Dr. E. Rahm
T. Böhme, Dr. P. Richter

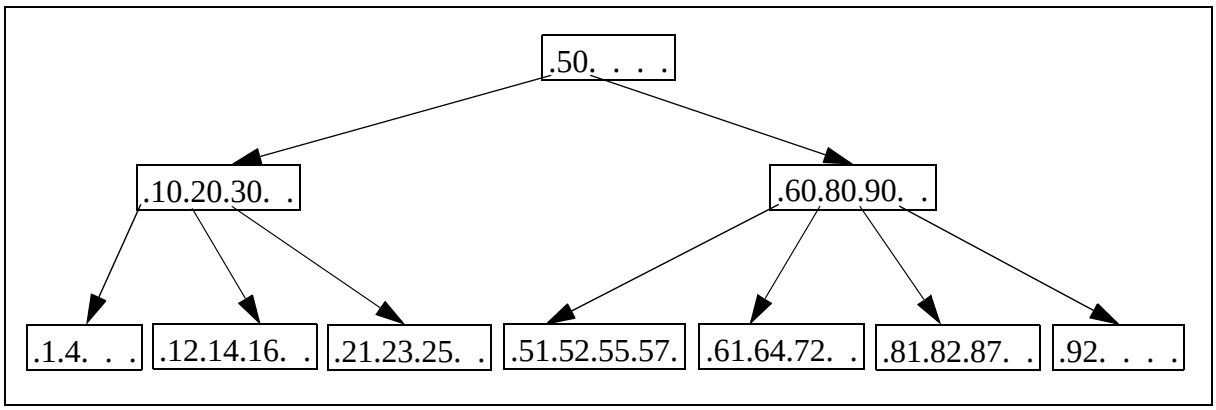
Ausgabe: 22.04.2002
Abgabe: 06.05.2002

Algorithmen und Datenstrukturen 2
SS 2002 - Übungsblatt 2

Hinweis: Alle *Listings* müssen in der ersten Zeile nach der Klassendefinition den *Namen des Programmautors* als Kommentar enthalten. Ohne diesen Namen werden die Listings **nicht gewertet**. Ebenfalls führt das Fehlen eines geforderten Ausdrucks der Programmausführung zur Nichtbewertung der Aufgabe.

1. Aufgabe (Einfügen und Löschen in B-Bäumen) (7 Punkte)

Gegeben sei der folgende 5-Wege-Suchbaum:



a) Diskutieren Sie, ob es sich bei dem gegebenen Baum um einen B-Baum mit k=2 handelt (begründen). Ergänzen Sie ihn (mit Knoten/Werten) gegebenenfalls so, daß ein B-Baum entsteht.

b) Fügen Sie in den gegebenenfalls erweiterten Baum in der gegebenen Reihenfolge die Schlüssel 84, 85 und 56 ein.

c) Löschen Sie danach in der gegebenen Reihenfolge die Schlüssel 23, 4 und 10.

Hinweis: bei b) und c) soll nach jeder Operation das Ergebnis in dem Teilbaum dargestellt werden, wo Änderungen erfolgt sind.

2. Aufgabe (Einfügen und Löschen in B*-Bäumen) (5 Punkte)

Gegeben sei die Folge 7, 5, 23, 19, 3, 2, 31, 17, 12

a) Erzeugen Sie mit der gegebenen Folge als Schlüsselwert einen B*-Baum mit k=k*=2. Stellen Sie diesen Baum dar.

b) Fügen Sie in diesen Baum die Schlüssel 9 und 11 ein. Löschen Sie danach den Schlüssel 17. Stellen Sie jeweils die geänderten Teilbäume dar.

3. Aufgabe (Höhe von B*-Bäumen) (6 Punkte)

a) Leiten Sie die auf der Vorlesungsfolie 1-31 angegebene Formel für die Höhe eines B*-Baumes her.

b) Für den Aufbau eines B*-Baumes seien folgende Parameter gegeben:
- Seitengröße L=8192 B
- $l_M=l_P=l_K=4$ B
- eingebettete Speicherung: $l_D=116$ B

Bestimmen Sie die minimale und maximale Höhe des Baumes für 1 Million und 1 Milliarde Datensätze.

4. Aufgabe (Einfügen in B-Bäume) (10 Punkte)

Implementieren Sie in Java das Einfügen von Schlüssel-Wert-Paaren in einen B-Baum. Auf der Web-Übungsseite haben wir Ihnen dafür die Klasse *BTree* bereitgestellt, die einen B-Baum teilweise implementiert. Die darin als Rumpf vorgegebene Methode *public void insert (Orderable key, Object obj)* ist von Ihnen zu implementieren. Sie soll einen neuen Schlüsselwert (key) mit dazugehörigem Datenobjekt (obj) in den Baum einfügen. Die Baumknoten werden durch die Klasse *BNode* repräsentiert. Der Einfügealgorithmus soll gemäß dem in der Vorlesung (Folie 1-21 ff) gezeigten Verfahren arbeiten. Soll ein Schlüssel eingefügt werden, der schon im Baum existiert, so wird nur das Datenobjekt aktualisiert. Die Einfügemethode muss die Baumhöhe, die in der Variablen *height* gespeichert ist, bei Bedarf anpassen.

Hinweise:
- Die Arrays für Schlüssel, Daten und Zeiger in der Klasse *BNode* sind eine Stelle größer als normalerweise notwendig definiert. Dies kann für eine vereinfachte Überlaufbehandlung beim Einfügen genutzt werden (es kann in jedem Fall eingefügt und danach der eventuell notwendige Split-Vorgang durchgeführt werden).
- Für das Einfügen bietet sich die Definition einer rekursiven Methode an. Mittels einer Hilfsklasse, die bestimmte Daten kapselt, kann ein komplexer Rückgabewert (z.B. Split-Informationen: linker Knoten, mittlerer Schlüssel, rechter Knoten) für diese Methode realisiert werden.

Überprüfen Sie die Implementierung mit dem ebenfalls bereitgestellten Programm *Ueb2Test*, welches mittels vorgegebenen Werten und Ihrer Einfügeimplementierung einen B-Baum aufbaut, einen Durchlauf in symmetrischer Ordnung sowie die Baumstruktur und die Baumhöhe ausgibt. Für das Übersetzen und Ausführen des Programms benötigen Sie die auf der Web-Übungsseite bereitgestellten Dateien *BNode.java*, *BTree.java*, *Orderable.java*, *OrderableInt.java* und *Ueb2Test.java*.

Abgabe:
1. dokumentiertes Listing des Quellprogramms (*BTree.java*, eventuelle Hilfsklassen)
2. Ausgabe eines Programmlaufs von *Ueb2Test*