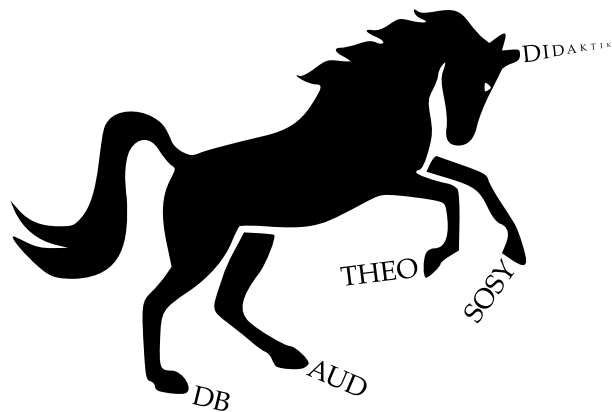


Die komplette Sammlung

Alle Aufgaben



Die Bschlangaul-Sammlung

Hermine Bschlangaul and Friends

Inhaltsverzeichnis

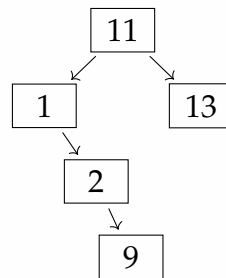
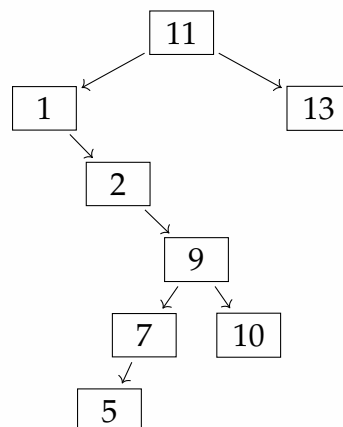
46115 / 2014 / Frühjahr / Thema 1 / Aufgabe 7	3
46116 / 2018 / Herbst / Thema 2 / Teilaufgabe 2 / Aufgabe 3	6
66115 / 2013 / Herbst / Thema 2 / Aufgabe 8	11
66115 / 2014 / Frühjahr / Thema 1 / Aufgabe 3	13
66115 / 2017 / Herbst / Thema 2 / Aufgabe 8	17
66115 / 2019 / Herbst / Thema 2 / Aufgabe 7	22
66116 / 2016 / Herbst / Thema 2 / Teilaufgabe 2 / Aufgabe 1	24
66116 / 2017 / Frühjahr / Thema 1 / Teilaufgabe 1 / Aufgabe 2	26

46115 / 2014 / Frühjahr / Thema 1 / Aufgabe 7

Fügen Sie nacheinander die Zahlen 11, 1, 2, 13, 9, 10, 7, 5

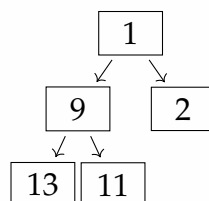
- (a) in einen leeren binären Suchbaum und zeichnen Sie den Suchbaum jeweils nach dem Einfügen von „9“ und „5“

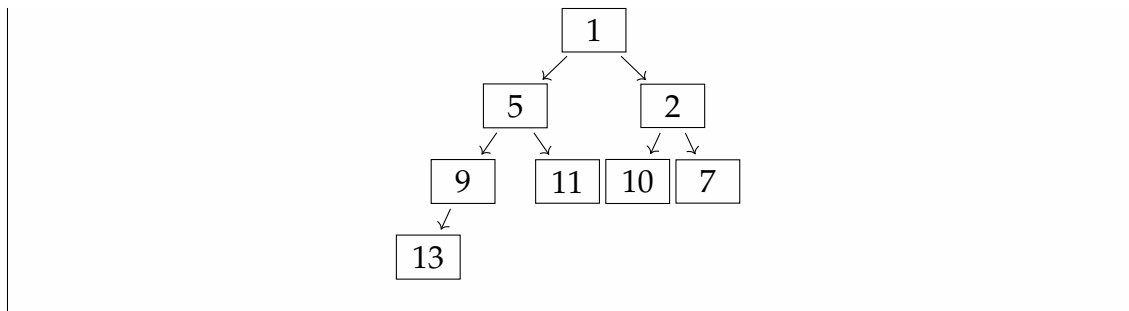
Lösungsvorschlag

Nach dem Einfügen von „9“:*Nach dem Einfügen von „5“:*

- (b) in einen leeren Min-Heap ein, der bzgl. „ \leq “ angeordnet ist und geben Sie den Heap nach „9“ und nach „5“ an

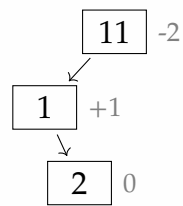
Lösungsvorschlag

Nach dem Einfügen von „9“:*Nach dem Einfügen von „5“:*

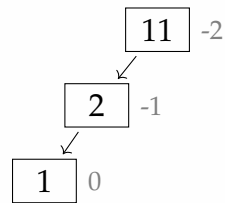


- (c) in einen leeren AVL-Baum ein! Geben Sie den AVL Baum nach „2“ und „5“ an und beschreiben Sie die ggf. notwendigen Rotationen beim Einfügen dieser beiden Elemente!

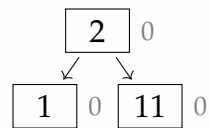
Nach dem Einfügen von „2“:



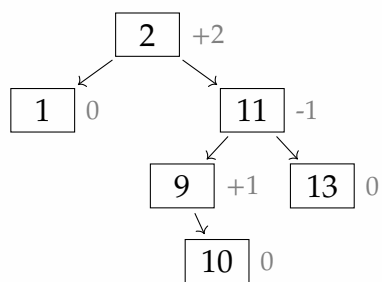
Nach der Linksrotation:



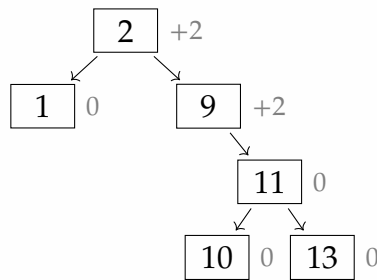
Nach der Rechtsrotation:



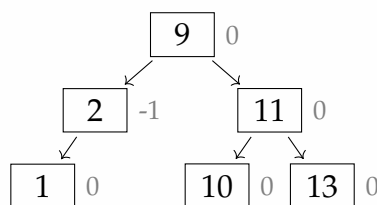
Nach dem Einfügen von „10“:



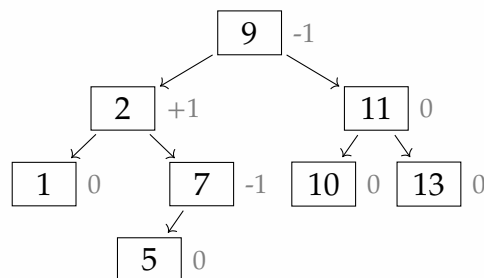
Nach der Rechtsrotation:



Nach der Linksrotation:



Nach dem Einfügen von „5“:



46116 / 2018 / Herbst / Thema 2 / Teilaufgabe 2 / Aufgabe 3

Gegeben sei das folgende Datenbank-Schema, das für die Speicherung der Daten einer Schule entworfen wurde, zusammen mit einem Teil seiner Ausprägung. Die Primärschlüssel-Attribute sind jeweils unterstrichen.

Die Relation *Schüler* enthält allgemeine Daten zu den Schülerinnen und Schülern. Schülerinnen und Schüler nehmen an Prüfungen in verschiedenen Unterrichtsfächern teil und erhalten dadurch Noten. Diese werden in der Relation *Noten* abgespeichert. Prüfungen haben ein unterschiedliches Gewicht. Beispielsweise hat ein mündliches Ausfragen oder eine Extemporale das Gewicht 1, während eine Schulaufgabe das Gewicht 2 hat.

Schüler:

CREATE TABLE
INSERT

<u>SchülerID</u>	Vorname	Nachname	Klasse
1	Laura	Müller	4A
2	Linus	Schmidt	4A
3	Jonas	Schneider	4A
4	Liam	Fischer	4B
5	Tim	Weber	4B
6	Lea	Becker	4B
7	Emilia	Klein	4C
8	Julia	Wolf	4C

Noten:

<u>SchülerID</u> [Schüler]	Schulfach	Note	Gewicht	Datum
1	Mathematik	3	2	23.09.2017
1	Mathematik	1	1	03.10.2017
1	Mathematik	2	2	15.10.2017
1	Mathematik	4	1	11.11.2017

- (a) Geben Sie die SQL-Befehle an, die notwendig sind, um die oben dargestellten Tabellen in einer SQL-Datenbank anzulegen.

Lösungsvorschlag

```

CREATE TABLE IF NOT EXISTS Schüler (
  SchülerID INTEGER PRIMARY KEY NOT NULL,
  Vorname VARCHAR(20),
  Nachname VARCHAR(20),
  Klasse VARCHAR(5)
);

CREATE TABLE IF NOT EXISTS Noten (
  SchülerID INTEGER NOT NULL,
  Schulfach VARCHAR(20),
  Note INTEGER,
  Gewicht INTEGER,
  Datum DATE,
  PRIMARY KEY (SchülerID, Schulfach, Datum),
  FOREIGN KEY (SchülerID) REFERENCES Schüler(SchülerID)
);

```

- (b) Entscheiden Sie jeweils, ob folgende Einfügeoperationen vom gegebenen Datenbanksystem (mit der angegebenen Ausprägungen) erfolgreich verarbeitet werden können und begründen Sie Ihre Antwort kurz.

```
INSERT INTO Schüler
(SchülerID, Vorname, Nachname, Klasse)
VALUES
(6, 'Johannes', 'Schmied', '4C');
```

Lösungsvorschlag

Nein. Ein/e Schüler/in mit der ID 6 existiert bereits. Primärschlüssel müssen eindeutig sein.

```
INSERT INTO Noten VALUES (6, 'Chemie', 1, 2, '1.4.2020');
```

Lösungsvorschlag

Nein. Ein *Datum* ist zwingend notwendig. Da *Datum* im Primärschlüssel enthalten ist, darf es nicht NULL sein. Es gibt auch keine/n Schüler/in mit der ID 9. Der/die Schüler/in müsste vorher angelegt werden, da die Spalte *SchülerID* von der Tabelle *Noten* auf den Fremdschlüssel *SchülerId* aus der Schülertabelle verweist.

- (c) Geben Sie die Befehle für die folgenden Aktionen in SQL an. Beachten Sie dabei, dass die Befehle auch noch bei Änderungen des oben gegebenen Datenbankzustandes korrekte Ergebnisse zurückliefern müssen.

- Die Schule möchte verhindern, dass in die Datenbank mehrere Kinder mit dem selben Vornamen in die gleiche Klasse kommen. Dies soll bereits auf Datenbankebene verhindert werden. Dabei sollen die Primärschlüssel nicht verändert werden. Geben Sie den Befehl an, der diese Änderung durchführt.

Lösungsvorschlag

```
ALTER TABLE Schüler
ADD CONSTRAINT eindeutiger_Vorname UNIQUE (Vorname, Klasse);
```

- Der Schüler *Tim Weber* (SchülerID: 5) wechselt die Klasse. Geben Sie den SQL-Befehl an, der den genannten Schüler in die Klasse „4C“ überführt.

Lösungsvorschlag

```
UPDATE Schüler
SET Klasse = '4C'
WHERE
  Vorname = 'Tim' AND
  Nachname = 'Weber' AND
  SchülerID = 5;
```

- Die Schülerin *Laura Müller* (SchülerID: 1) zieht um und wechselt die Schule. Löschen Sie die Schülerin aus der Datenbank. Nennen Sie einen möglichen Effekt, welcher bei der Verwendung von Primär- und Fremdschlüsseln auftreten kann.

Alle Noten von *Laura Müller* werden gelöscht, falls **ON DELETE CASCADE** gesetzt ist. Oder es müssen erst alle Fremdschlüsselverweise auf diese *SchülerID* in der Tabelle *Noten* gelöscht werden

```
DELETE FROM Noten
WHERE SchülerID = 1;
```

- Erstellen Sie eine View „DurchschnittsNoten“, die die folgenden Spalten beinhaltet: *Klasse*, *Schulfach*, *Durchschnittsnote*

Hinweis: Beachten Sie die Gewichte der Noten.

```
CREATE VIEW DurchschnittsNoten AS (
  (SELECT s.Klasse, n.Schulfach, (SUM(n.Note * n.Gewicht) /
    ↳ SUM(n.Gewicht)) AS Durchschnittsnote
  FROM Noten n, Schüler s
  WHERE s.SchülerID = n.SchülerID
  GROUP BY s.Klasse, n.Schulfach)
);
```

```
SELECT * FROM DurchschnittsNoten;
```

klasse	schulfach	durchschnittsnote
4A	Mathematik	2

(1 row)

- Geben Sie den Befehl an, der die komplette Tabelle „Noten“ löscht.

```
DROP TABLE Noten;
```

- (d) Formulieren Sie die folgenden Anfragen in SQL. Beachten Sie dabei, dass sie SQL-Befehle auch noch bei Änderungen der Ausprägung die korrekten Anfrageergebnisse zurückgeben sollen.

- Gesucht ist die durchschnittliche Note, die im Fach Mathematik vergeben wird.

Hinweis: Das Gewicht ist bei dieser Anfrage nicht relevant

```
SELECT AVG(Note)
FROM Noten
WHERE Schulfach = 'Mathematik';
```

- Berechnen Sie die Anzahl der Schüler, die im Fach Mathematik am 23.09.2017 eine Schulaufgabe (öGewicht=2) geschrieben haben.

Lösungsvorschlag

```
SELECT COUNT(*) AS Anzahl_Schüler
FROM Noten
WHERE Datum = '23.09.2017' AND Gewicht = 2 AND Schulfach = 'Mathematik';
```

anzahl_schüler
1

(1 row)

- Geben Sie die *SchülerID* aller Schüler zurück, die im Fach Mathematik mindestens drei mal die Schulnote 6 geschrieben haben.

Lösungsvorschlag

```
SELECT SchülerID
FROM Noten
WHERE Schulfach = 'Mathematik' AND Note = 6
GROUP BY SchülerID
HAVING COUNT(*) >= 3;
```

schülerid

(0 rows)

- Gesucht ist der Notendurchschnitt bezüglich jedes Fachs der Klasse „4A“.

Lösungsvorschlag

```
SELECT n.Schulfach, AVG(n.Note)
FROM Schüler s, Noten n
WHERE s.SchülerID = n.SchülerID AND s.Klasse = '4A'
GROUP BY n.Schulfach;
```

schulfach	avg
Mathematik	2.5000000000000000

(1 row)

- (e) Geben Sie jeweils an, welchen Ergebniswert die folgenden SQL-Befehle für die gegebene Ausprägung zurückliefern.

```
SELECT COUNT(DISTINCT Klasse)
FROM
Schüler NATURAL JOIN Noten;
```

Lösungsvorschlag

count
1

(1 row)

4A von Laura Müller. Ohne **DISTINCT** wäre das Ergebnis 4.

```
SELECT COUNT(ALL Klasse)
FROM
Noten, Schüler;
```

```
count
-----
      32
(1 row)
```

Es entsteht das Kreuzprodukt ($8 \cdot 4 = 32$).

```
SELECT COUNT(Note)
FROM
Schüler NATURAL LEFT OUTER JOIN Noten;
```

```
SELECT * FROM Schüler NATURAL LEFT OUTER JOIN Noten;
```

ergibt:

schülerid	vorname	nachname	klasse	schulfach	note	gewicht	datum
1	Laura	Müller	4A	Mathematik	3	2	2017-09-23
1	Laura	Müller	4A	Mathematik	1	1	2017-10-03
1	Laura	Müller	4A	Mathematik	2	2	2017-10-15
1	Laura	Müller	4A	Mathematik	4	1	2017-11-11
2	Linus	Schmidt	4A				
5	Tim	Weber	4B				
8	Julia	Wolf	4C				
6	Lea	Becker	4B				
4	Liam	Fischer	4B				
3	Jonas	Schneider	4A				
7	Emilia	Klein	4C				

(11 rows)

```
count
-----
      4
(1 row)
```

COUNT zählt die **NULL**-Werte nicht mit. Die *Laura Müller* hat 4 Noten.

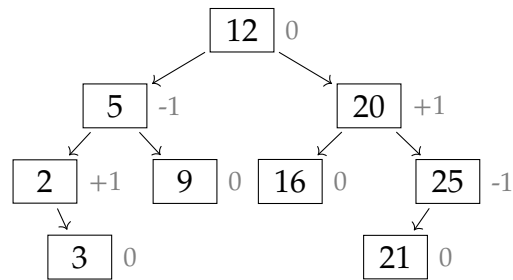
```
SELECT COUNT(*)
FROM
Schüler NATURAL LEFT OUTER JOIN Noten;
```

```
count
-----
     11
(1 row)
```

Siehe Zwischenergebnistabelle in der obenstehenden Antwort. Alle Schüler und die Laura 4-mal, weil sie 4 Noten hat.

66115 / 2013 / Herbst / Thema 2 / Aufgabe 8

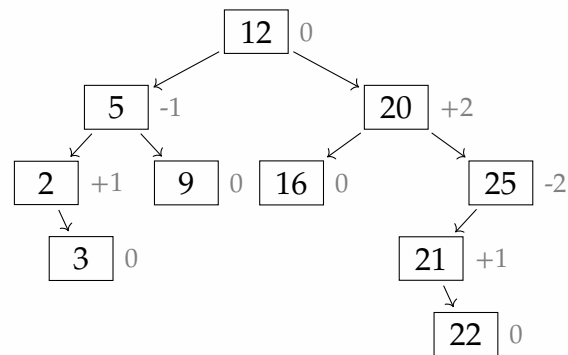
Gegeben sei der folgende AVL-Baum T . Führen Sie auf T folgende Operationen durch.



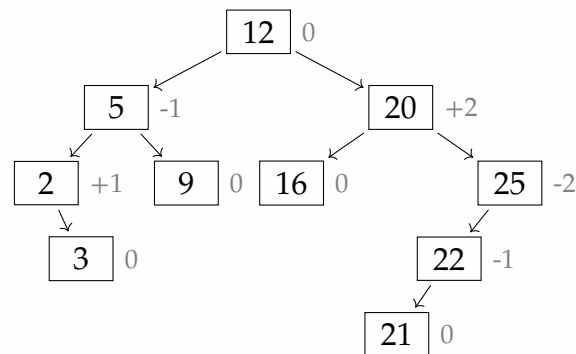
- (a) Fügen Sie den Wert 22 in T ein. Balancieren Sie falls nötig und geben Sie den entstandenen Baum (als Zeichnung) an.

Lösungsvorschlag

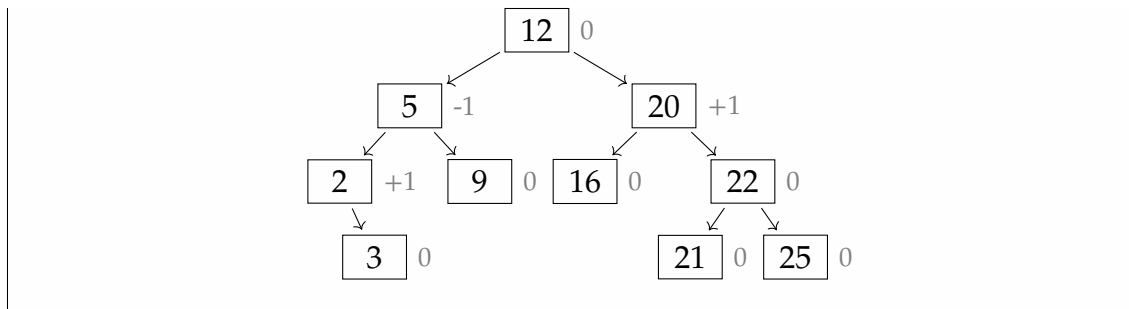
Nach dem Einfügen von „22“:



Nach der Linksrotation:



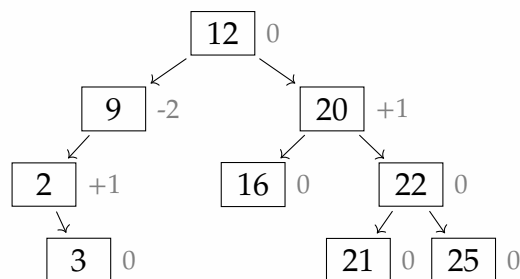
Nach der Rechtsrotation:



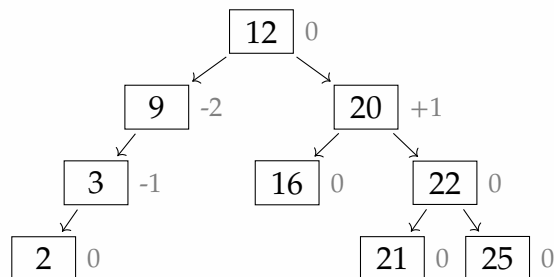
- (b) Löschen Sie danach die 5. Balancieren Sie T falls nötig und geben Sie den entstandenen Baum (als Zeichnung) an.

Lösungsvorschlag

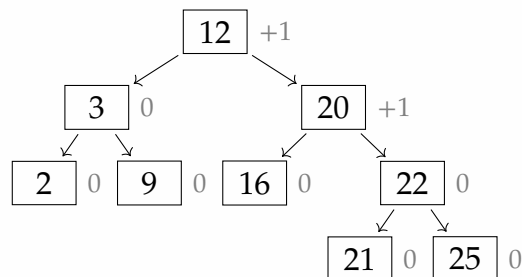
Nach dem Löschen von „5“:



Nach der Linksrotation:



Nach der Rechtsrotation:



66115 / 2014 / Frühjahr / Thema 1 / Aufgabe 3

- (a) Fügen Sie die Zahlen (7, 6, 2, 1, 5, 3, 8, 4) in dieser Reihenfolge in einen anfangs leeren AVL Baum ein. Stellen Sie die AVL Eigenschaft ggf. nach jedem Einfügen

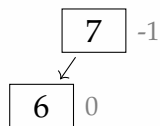
mit geeigneten Rotationen wieder her. Zeichnen Sie den AVL Baum einmal vor und einmal nach jeder einzelnen Rotation.

Lösungsvorschlag

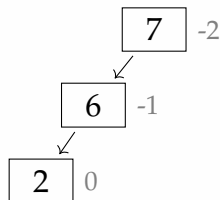
Nach dem Einfügen von „7“:



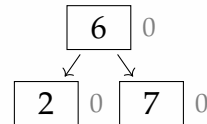
Nach dem Einfügen von „6“:



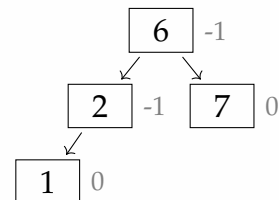
Nach dem Einfügen von „2“:



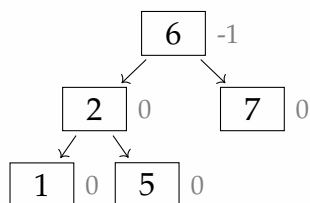
Nach der Rechtsrotation:



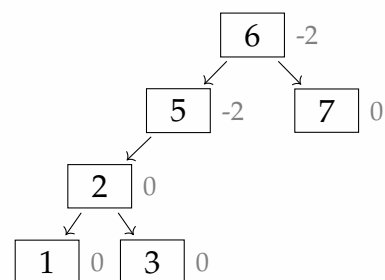
Nach dem Einfügen von „1“:



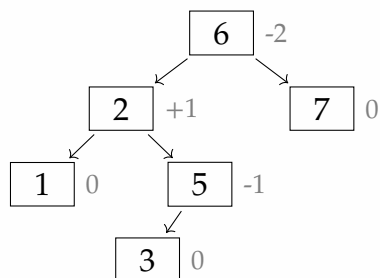
Nach dem Einfügen von „5“:



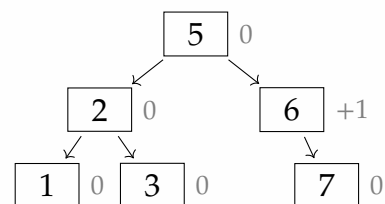
Nach der Linksrotation:



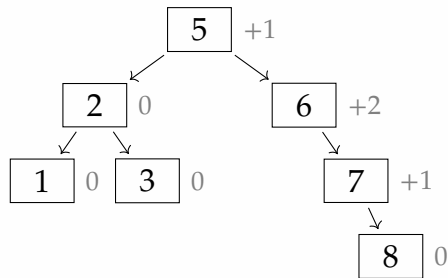
Nach dem Einfügen von „3“:



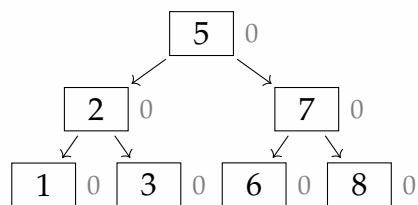
Nach der Rechtsrotation:



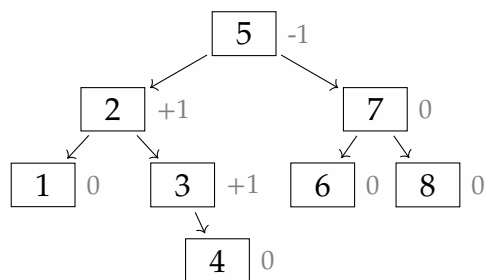
Nach dem Einfügen von „8“:



Nach der Linksrotation:

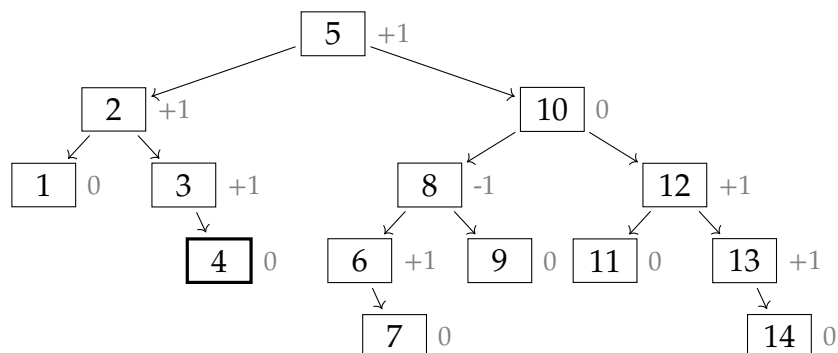


Nach dem Einfügen von „4“:

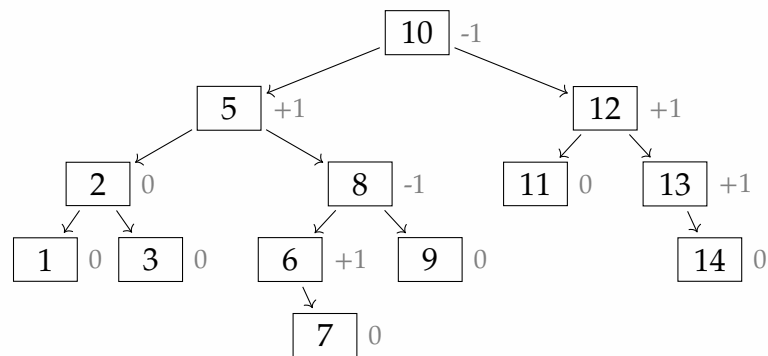


- (b) Entfernen Sie den jeweils markierten Knoten aus den folgenden AVL-Bäumen. Stellen Sie die AVL-Eigenschaft ggf. durch geeignete Rotationen wieder her. Zeichnen Sie nur den resultierenden Baum.

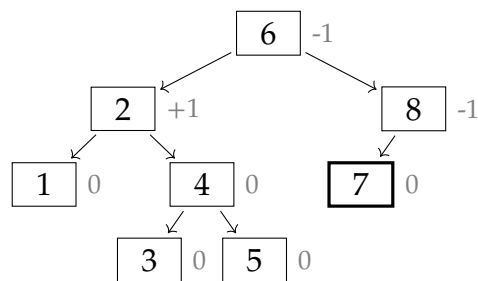
(i) Baum 1:



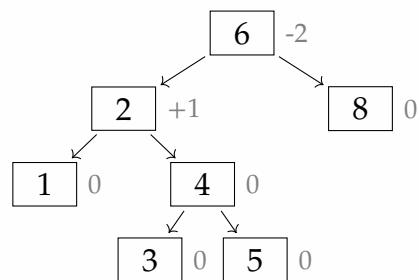
Nach dem Löschen von „4“:



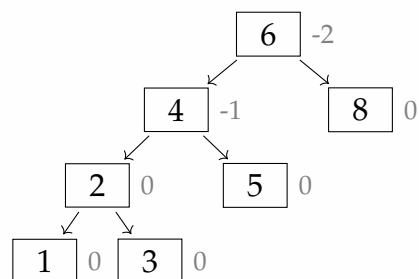
(ii) Baum 2:



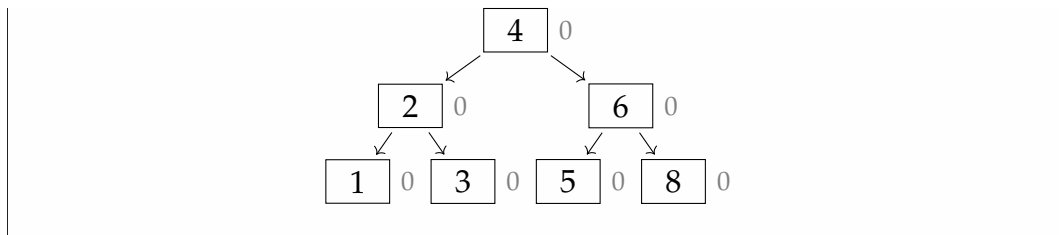
Nach dem Löschen von „7“:



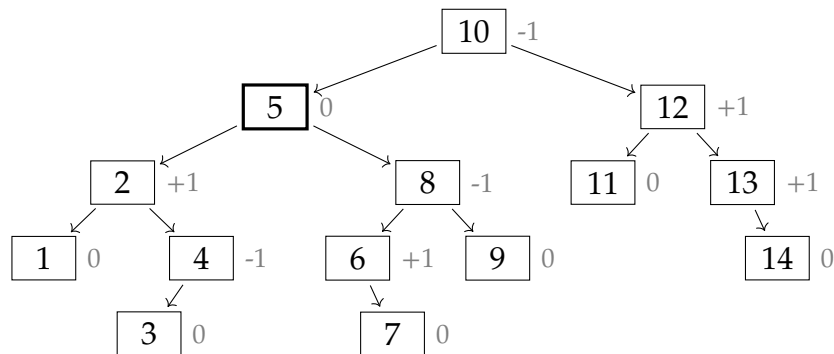
Nach der Linksrotation:



Nach der Rechtsrotation:

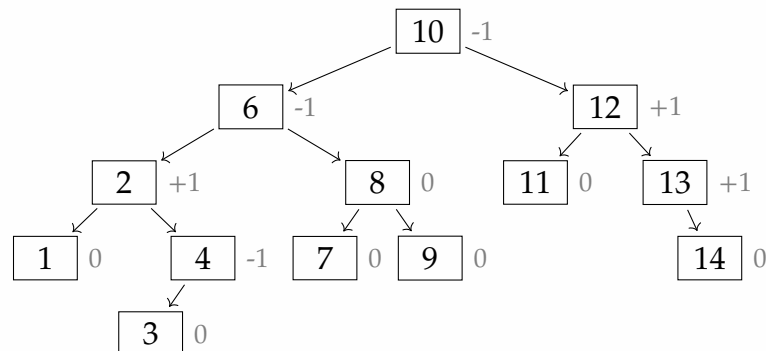


(iii) Baum 3:



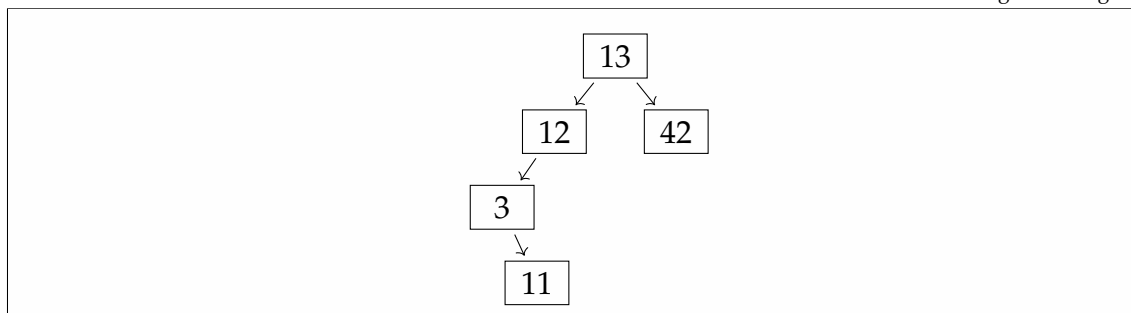
Lösungsvorschlag

Nach dem Löschen von „5“:

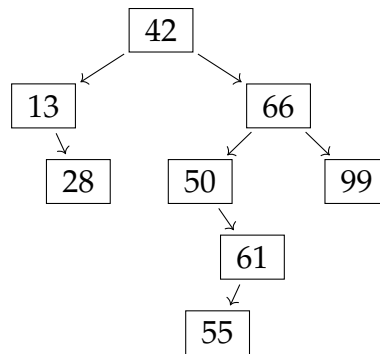
**66115 / 2017 / Herbst / Thema 2 / Aufgabe 8**

- (a) Fügen Sie die Zahlen 13, 12, 42, 3, 11 in der gegebenen Reihenfolge in einen zunächst leeren binären Suchbaum mit aufsteigender Sortierung ein. Stellen Sie nur das Endergebnis dar.

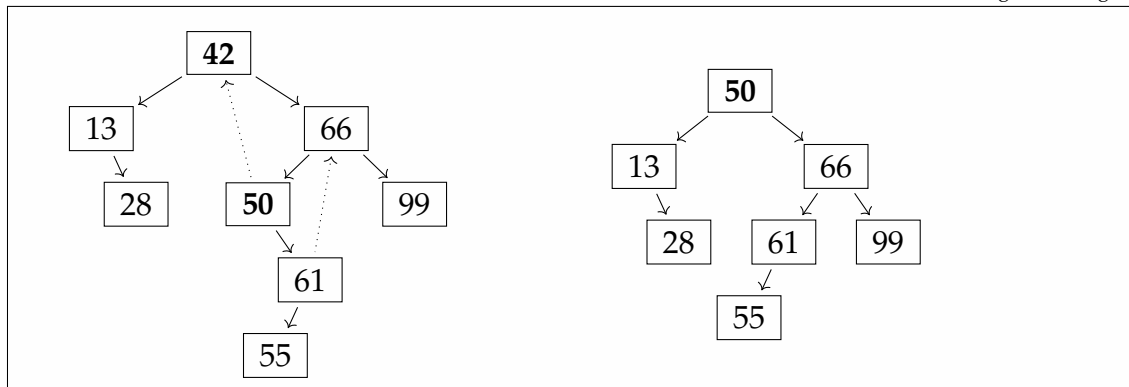
Lösungsvorschlag



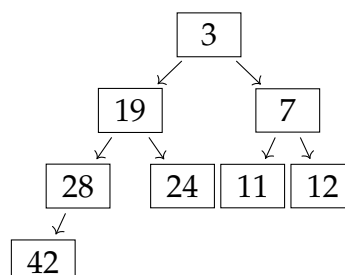
- (b) Löschen Sie den Wurzelknoten mit Wert 42 aus dem folgenden *binären* Suchbaum mit aufsteigender Sortierung und ersetzen Sie ihn dabei durch einen geeigneten Wert aus dem *rechten* Teilbaum. Lassen Sie möglichst viele Teilbäume unverändert und erhalten Sie die Suchbaumeigenschaft.



Lösungsvorschlag

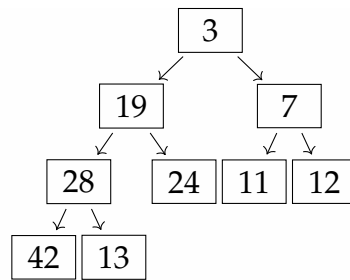


- (c) Fügen Sie einen neuen Knoten mit dem Wert 13 in die folgende Min-Halde ein und stellen Sie anschließend die Halde-Eigenschaft vom neuen Blatt aus beginnend wieder her, wobei möglichst viele Knoten der Halde unverändert bleiben und die Halde zu jedem Zeitpunkt links-vollständig sein soll. Geben Sie nur das Endergebnis an.

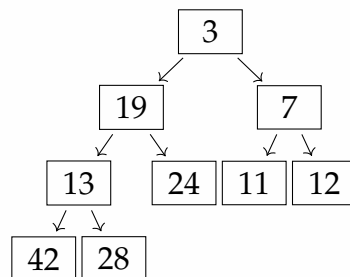


Lösungsvorschlag

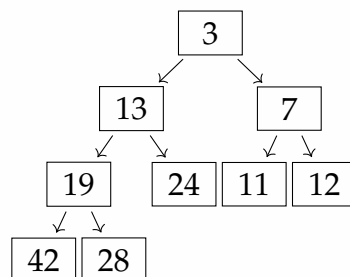
Nach dem Einfügen von „13“:



Nach dem Vertauschen von „13“ und „28“:



Nach dem Vertauschen von „13“ und „19“:

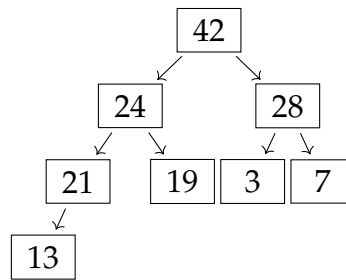


- (d) Geben Sie für die ursprüngliche Min-Halde aus Teilaufgabe c) (ööhne den neu eingefügten Knoten mit dem Wert 13) die Feld-Einbettung (Array-Darstellung) an.

Lösungsvorschlag

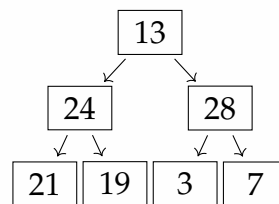
0	1	2	3	4	5	6	7
3	19	7	28	24	11	12	42

- (e) Löschen Sie den Wurzelknoten mit Wert 42 aus der folgenden Max-Halde und stellen Sie anschließend die Halden-Eigenschaft ausgehend von einer neuen Wurzel wieder her, wobei möglichst viele Knoten der Halde unverändert bleiben und die Halde zu jedem Zeitpunkt links-vollständig sein soll. Geben Sie nur das Endergebnis an.

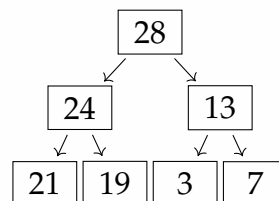


Lösungsvorschlag

Nach dem Ersetzen von „42“ mit „13“:

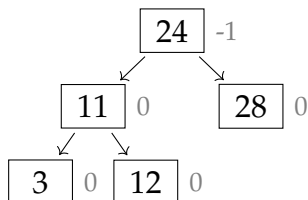


Nach dem Vertauschen von „13“ und „28“:



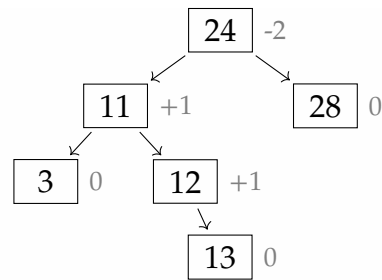
- (f) Fügen Sie in jeden der folgenden AVL-Bäume mit aufsteigender Sortierung jeweils einen neuen Knoten mit dem Wert 13 ein und führen Sie anschließend bei Bedarf die erforderliche(n) Rotation(en) durch. Zeichnen Sie den Baum vor und nach den Rotationen.

(i) AVL-Baum A

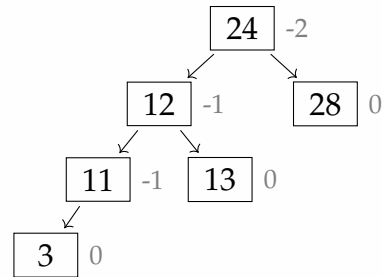


Lösungsvorschlag

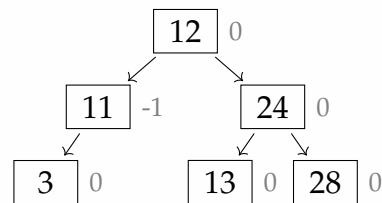
Nach dem Einfügen von „13“:



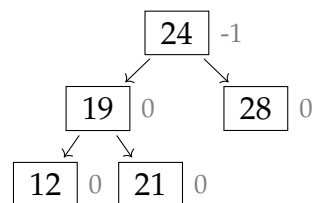
Nach der Linksrotation:



Nach der Rechtsrotation:

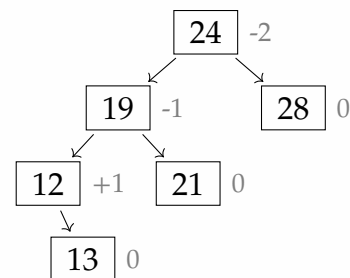


(ii) AVL-Baum B

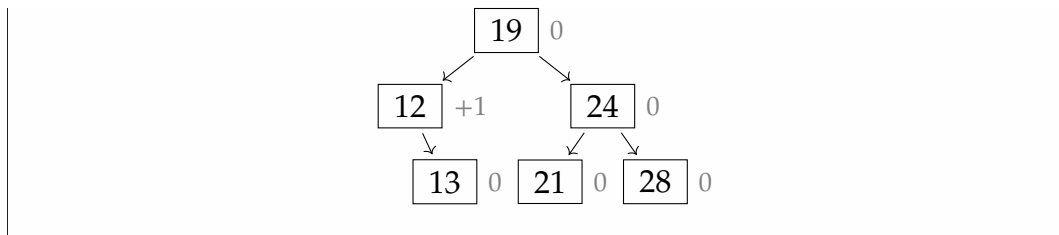


Lösungsvorschlag

Nach dem Einfügen von „13“:

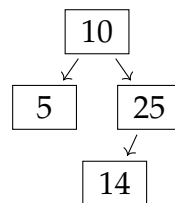


Nach der Rechtsrotation:



66115 / 2019 / Herbst / Thema 2 / Aufgabe 7

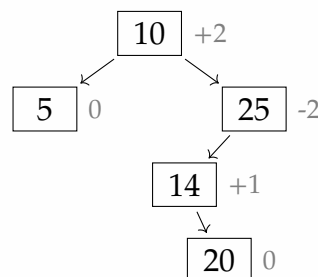
Fügen Sie (manuell) nacheinander die Zahlen 20, 31, 2, 17, 7 in folgenden AVL-Baum ein. Löschen Sie anschließend aus dem entstandenen Baum nacheinander 14 und 25.



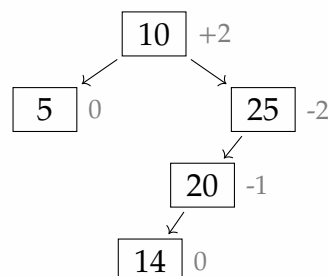
Zeichnen Sie jeweils direkt nach jeder einzelnen Operation zum Einfügen oder Löschen eines Knotens, sowie nach jeder elementaren Rotation den entstehenden Baum. Insbesondere sind evtl. anfallende Doppelrotationen in zwei Schritten darzustellen. Geben Sie zudem an jedem Knoten die Balancewerte an.

Lösungsvorschlag

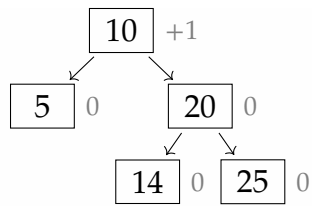
Nach dem Einfügen von „20“:



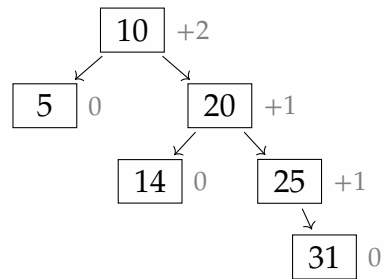
Nach der Linksrotation:



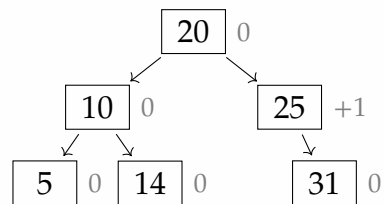
Nach der Rechtsrotation:



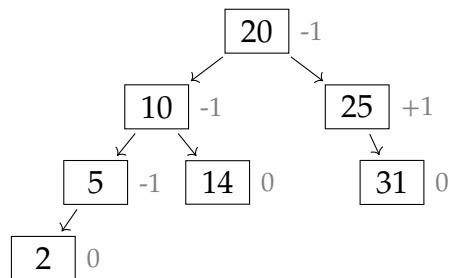
Nach dem Einfügen von „31“:



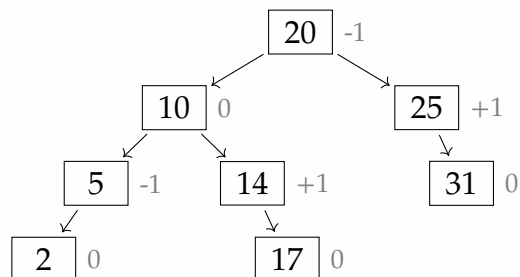
Nach der Linksrotation:



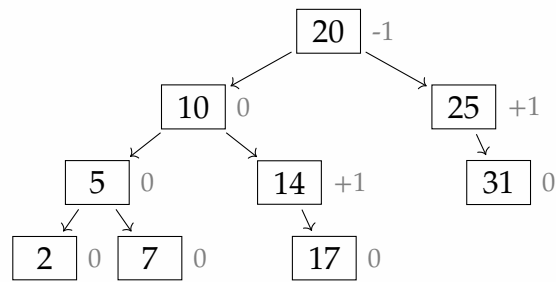
Nach dem Einfügen von „2“:



Nach dem Einfügen von „17“:

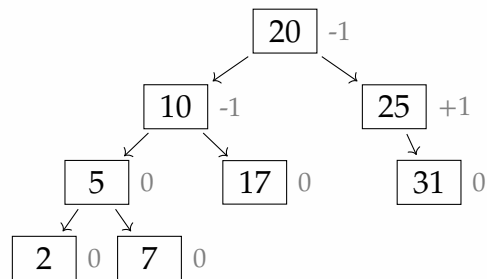


Nach dem Einfügen von „7“:

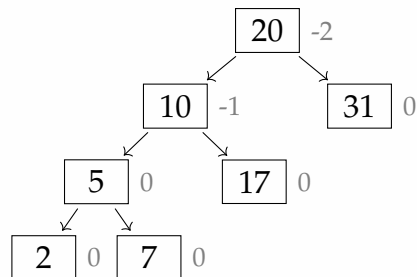


Löschen

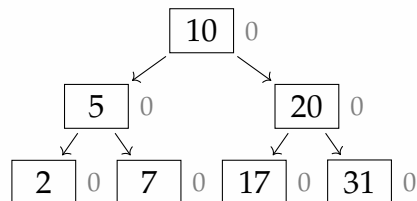
Nach dem Löschen von „14“:



Nach dem Löschen von „25“:



Nach der Rechtsrotation:



66116 / 2016 / Herbst / Thema 2 / Teilaufgabe 2 / Aufgabe 1

Ordnen Sie die folgenden Aussagen entsprechend ihres Wahrheitsgehaltes in einer Tabelle der folgenden Form an:

Kategorie	WAHR	FALSCH
X	X1, X3	X2
Y	Y2	Y1
...

A Allgemein

- A1** Im Software Engineering geht es vor allem darum qualitativ hochwertige Software zu entwickeln.
- A2** Software Engineering ist gleichbedeutend mit Programmieren.

B Vorgehensmodelle

- B1** Die Erhebung und Analyse von Anforderungen sind nicht Teil des Software Engineerings.
- B2** Agile Methoden eignen sich besonders gut für die Entwicklung komplexer und sicherer Systeme in verteilten Entwicklerteams.
- B3** Das Spiralmodell ist ein Vorläufer sogenannter Agiler Methoden.

C Anforderungserhebung

- C1** Bei der Anforderungserhebung dürfen in keinem Fall mehrere Erhebungstechniken (z. B. Workshops, Modellierung) angewendet werden, weil sonst Widersprüche in Anforderungen zu Vorschein kommen könnten.
- C2** Ein Szenario beinhaltet eine Menge von Anwendungsfällen.
- C3** Nicht-funktionale Anforderungen sollten, wenn möglich, immer quantitativ spezifiziert werden.

D Architekturmuster

- D1** Schichtenarchitekturen sind besonders für Anwendungen geeignet, in denen Performance eine wichtige Rolle spielt.
- D2** Das Black Board Muster ist besonders für Anwendungen geeignet, in denen Performance eine wichtige Rolle spielt.
- D3** „Dependency Injection“ bezeichnet das Konzept, welches Abhängigkeiten zur Laufzeit reglementiert.

E UML

- E1** Sequenzdiagramme beschreiben Teile des Verhaltens eines Systems.
- E2** Zustandsübergangsdiagramme beschreiben das Verhalten eines Systems.
- E3** Komponentendiagramme beschreiben die Struktur eines Systems.

F Entwurfsmuster

- F1** Das MVC Pattern verursacht eine starke Abhängigkeit zwischen Datenmodell und Benutzeroberfläche.
- F2** Das Singleton Pattern stellt sicher, dass es zur Laufzeit von einer bestimmten Klasse höchstens ein Objekt gibt.
- F3** Im Kommando Entwurfsmuster (engl. „Command Pattern“) werden Befehle in einem sog. Kommando-Objekt gekapselt, um sie bei Bedarf rückgängig zu machen.

G Testen

- G1** Validation dient der Überprüfung von Laufzeitfehlern.
- G2** Testen ermöglicht sicherzustellen, dass ein Programm absolut fehlerfrei ist.
- G3** Verifikation dient der Überprüfung, ob ein System einer Spezifikation entspricht.

Lösungsvorschlag

Kategorie	WAHR	FALSCH
A	A1	A2
B	B3	B1, B2
C	C3	C1, C2
D	D3	D1, D2
E	E1, E2, E3	
F	F2, F3	F1
G	G3	G1 ^a , G2 ^b

^aValidierung: Prüfung der Eignung beziehungsweise der Wert einer Software bezogen auf ihren Einsatzzweck: „Wird das richtige Produkt entwickelt?“

^bEin Softwaretest prüft und bewertet Software auf Erfüllung der für ihren Einsatz definierten Anforderungen und misst ihre Qualität.

66116 / 2017 / Frühjahr / Thema 1 / Teilaufgabe 1 / Aufgabe 2

Konstruieren Sie einen B-Baum, dessen Knoten maximal 4 Einträge enthalten können, indem Sie der Reihe nach diese Suchschlüssel einfügen:

8, 10, 12, 20, 5, 30, 25, 11

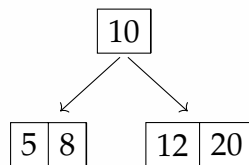
Anschließend löschen Sie den Eintrag mit dem Suchschlüssel 8.

Zeigen Sie jeweils graphisch den entstehenden Baum nach relevanten Zwischenschritten; insbesondere nach Einfügen der 5 sowie nach dem Einfügen der 11 und nach dem Löschen der 8.

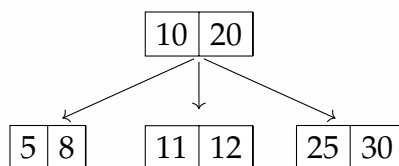
- Schlüsselwert 8 (einfaches Einfügen)
- Schlüsselwert 10 (einfaches Einfügen)
- Schlüsselwert 12 (einfaches Einfügen)
- Schlüsselwert 20 (einfaches Einfügen)

8	10	12	20
---	----	----	----

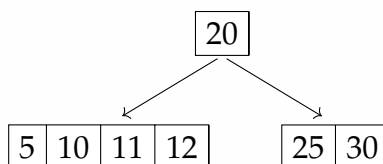
- Schlüsselwert 5 (Split)



- Schlüsselwert 30 (einfaches Einfügen)
- Schlüsselwert 25 (einfaches Einfügen)
- Schlüsselwert 11 (Split)



- Löschen des Schlüsselwerts 8 (Mischen/Verschmelzen)



Index

Agile Methoden, 25
ALTER TABLE, 8
AVL-Baum, 3, 11, 14, 17, 22

B-Baum, 26
Binärbaum, 3, 17
Blackboard-Muster, 25

CREATE TABLE, 7

DELETE, 8
DROP TABLE, 9

Einbringen von Abhängigkeiten (Dependency Injection), 25
Einzelstück (Singleton), 26
Entwurfsmuster, 25

GROUP BY, 9

Halde (Heap), 3, 17

INSERT, 7

Kommando (Command), 26
Komponentendiagramm, 25

Min-Heap, 3
Modell-Präsentation-Steuerung (Model-View-Controller), 26

Nicht-funktionale Anforderungen, 25

Schichtenarchitektur, 25
Sequenzdiagramm, 25
Software Engineering, 25
Spiralmodell, 25
SQL, 6

UPDATE, 8

Validation, 26
Verifikation, 26
VIEW, 9

Zustandsdiagramm Wissen, 25