Technische Universität Darmstadt
Fachgebiet Theoretische Informatik

Prof. Johannes Buchmann
Fateme Shirazi
Johannes Braun
Dr. Albrecht Petzoldt

Grundlagen der

Informatik 2

SS 2014

TECHNISCHE UNIVERSITÄT DARMSTADT

Klausurlösung

Name: .... Matrikelnummer: ....

Studium: .... Diplom ☐ Bachelor ☐ Master ☐

Fachbereich: .... Fachsemester: ......

Prüfungssekretariat/Büro: .... None ☐

Semesterleistung erbracht? ☐ Wenn ja, wann? ....

Wiederholung? ☐ Versuch Nummer: ...... Jahr des letzten Versuchs: ....

Unterschrift: ....

Viel Erfolg!

Aufgabe	K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	Σ
Mögliche Punkte	10	10	4	7	9	10	10	10	10	5	5	10	100
Erreichte Punkte													

Allgemeines

- Halten Sie Ihren Studienausweis und einen Lichtbildausweis zur Kontrolle bereit.
- Füllen Sie das Deckblatt vollständig aus.
- Prüfen Sie, ob die Klausur 12 Aufgaben enthält.
- Kennzeichnen Sie alle Blätter mit Name und Matrikelnummer.
- Verwenden Sie für jede Aufgabe ein neues Blatt.
- Leerblätter werden von der Aufsicht gestellt. Verwenden Sie kein eigenes Papier.
- Geben Sie die verwendeten Formeln, Sachverhalte und Zwischenergebnisse an.

Bewertung

- Unleserlichkeit führt zu Punktabzug.
- Diese Klausur wird nur gewertet, falls die Semesterleistung ordnungsgemäß erbracht wurde.

Dauer der Klausur und zugelassene Hilfsmittel

- Ihnen stehen 120 Minuten zum Bearbeiten der Aufgaben zur Verfügung.
- Einzige zugelassene Hilfsmittel sind ein nicht programmierbarer Taschenrechner ohne Formelspeicher und ein beidseitig handschriftlich beschriebenes DIN-A4 Blatt (keine Kopien etc.).
- Bitte geben Sie andere elektronische Geräte (Handys, PDAs, Laptops, programmierbare Taschenrechner) der Klausuraufsicht zur Verwahrung.
- Studierende, deren Muttersprache nicht Deutsch ist, können zusätzlich ein zweisprachiges Wörterbuch verwenden.
- Die Klausuraufsicht überprüft die Hilfsmittel.

Prüfungstyp

- Sie sind in einem Diplomstudiengang und haben sich in keinem Prüfungssekretariat zur Prüfung angemeldet
  $\Longrightarrow$ Dies ist eine vorlesungsbegleitende Prüfung für Sie, die Sie in einer Diplomprüfung einbringen können.
- Sie sind in einem Bachelor/Master-Studiengang und haben sich bei TUCaN angemeldet
  $\Longrightarrow$ Dies ist eine Fachprüfung für Sie.
- Sie sind in einem Bachelor/Master-Studiengang und haben sich nicht bei TUCaN angemeldet
  $\Longrightarrow$ STOP: Sie dürfen an dieser Prüfung nicht teilnehmen!

K1 Verschiedenes (10 Punkte)

a) Vervollständigen Sie die folgende Definition:

$$f(n) = \Theta(g(n)) \Leftrightarrow \exists c_1, c_2 \in \mathbb{R}_{>0}, n_0 \in \mathbb{N},\ so\ dass\ \forall n \geq n_0 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\ und\ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_$$

b) Vervollständigen Sie den folgenden Text zu Rot-Schwarz-Bäumen.

- Die Wurzel ist ____________________.
- Jedes Blatt (NIL) ist ____________________.
- Ist ein Knoten ____________________, so sind seine beiden Kinder ____________________.
- Für jeden Knoten x gilt: Alle Pfade von x zu einem Blatt enthalten die gleiche Zahl ____________________ Knoten.
- Die Höhe h eines Rot-Schwarz-Baumes mit n Knoten kann durch ____________________ nach oben abgeschätzt werden.

c) Nennen Sie zwei NP-vollständige Probleme.

Lösung.

a)

$$f(n) = \Theta(g(n)) \Leftrightarrow \exists c_1, c_2 \in \mathbb{R}_{>0}, n_0 \in \mathbb{N},\ so\ dass\ \ \forall n \geq n_0\ f(n) \leq c_1 \cdot g(n)\ und\ f(n) \geq c_2 \cdot g(n)$$

(2 Punkte)

b)

- Die Wurzel ist schwarz.
- Jedes Blatt (NIL) ist schwarz.
- Ist ein Knoten rot, so sind seine beiden Kinder schwarz.
- Für jeden Knoten x gilt: Alle Pfade von x zu einem Blatt enthalten die gleiche Zahl schwarzer Knoten.
- Die Höhe h eines Rot-Schwarz-Baumes mit n Knoten kann durch $h \leq 2\lg(n+1)$ nach oben abgeschätzt werden.

(je Lücke 1 Punkt; maximal 6 Punkte)

c)

Richtig (u.a.): Travelling Salesman, Hamiltonkreis (Rudratapfad), 3SAT, Circuit-SAT, längste Pfade, 3D Matching, Knapsack, Independent Set, Integer Linear Programming, balancierter Schnitt

Falsch: Eulerpfad, kürzeste Pfade, 2SAT, HORNSAT ...

(2 Punkte)

K2 Algorithmen (10 Punkte)

Betrachten Sie den folgenden Algorithmus zur Evaluierung eines Polynoms

$$P(x) = \sum_{k=0}^{n} a_k x^k.$$

Der Algorithmus erhält als Input einen Koeffizientenvektor $a = (a_0, \ldots, a_n)$ und eine Zahl x. Der Algorithmus gibt den Wert des Polynoms P an der Stelle x zurück.

```
HORNER(a, x)
1  y=0
2  i=n
3  while i ≥ 0
4     y= a_i + x · y
5     i = i − 1
6  return y
```

a) Beweisen Sie die Korrektheit von HORNER. Benutzen Sie dafür folgende Schleifeninvariante:
Am Anfang jeder Iteration der while-Schleife hat y den Wert

$$y = \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k.$$

b) Was ist die Laufzeit des Algorithmus (# arithmetische Operationen (Additionen + Multiplikationen)) in Abhängigkeit von n? (Groß-O-Notation). Begründen Sie Ihre Antwort.

Lösung.

a)
- Initialisierung ($i = n$): Zu Beginn des ersten Schleifendurchlaufs hat y den Wert 0. Die Summe

  $$\sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k$$

  ist für $i = n$ leer, also ist ihr Wert ebenfalls 0. (2 Punkte)
- Erhaltung: Zu Beginn des i-ten Schleifendurchgangs gilt nach der Induktionsvoraussetzung $y = \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k$. Nach dem i-ten Schleifendurchgang hat y den Wert (Zeile 4)

  $$y = a_i + x \cdot \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k = a_i + \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^{k+1} = a_i + \sum_{k=1}^{n-i} a_{k+i} x^k = \sum_{k=0}^{n-i} a_{k+i} x^k = \sum_{k=0}^{n-((i-1)+1)} a_{k+(i-1)+1} x^k.$$

  Damit ist die Schleifeninvariante auch vor Beginn des nächsten Schleifendurchlaufs erfüllt. (4 Punkte)
- Beendigung: Die Schleife terminiert mit $i = -1$. Zu diesem Zeitpunkt gilt

  $$y = \sum_{k=0}^{n-(i+1)} a_{k+i+1} x^k = \sum_{k=0}^{n} a_k x^k = P(x).$$

  Damit wird in Zeile 6 der Wert des Polynoms P an der Stelle x zurückgegeben. (2 Punkte)

b) Die Laufzeit jedes Schleifendurchlaufs ist konstant (1 Addition + 1 Multiplikation). Da die Schleife $O(n)$ mal durchlaufen wird, ist auch die Gesamtlaufzeit $O(n)$. (2 Punkte)

K3 Rekurrenzgleichungen (4 Punkte)

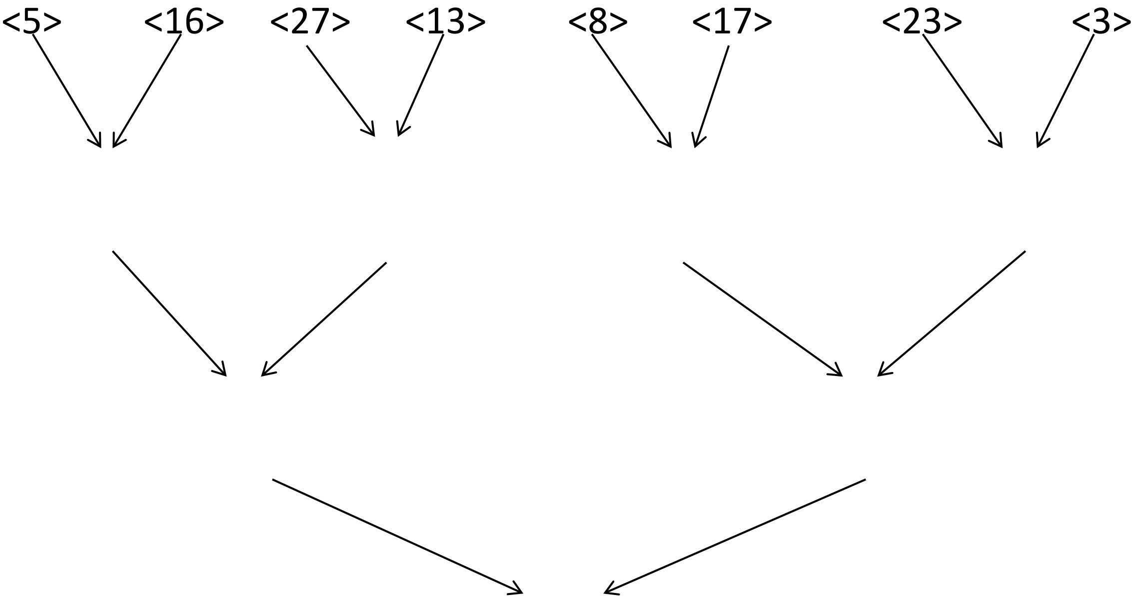
Lösen Sie folgende Rekurrenzgleichung mit Hilfe des Mastertheorems.

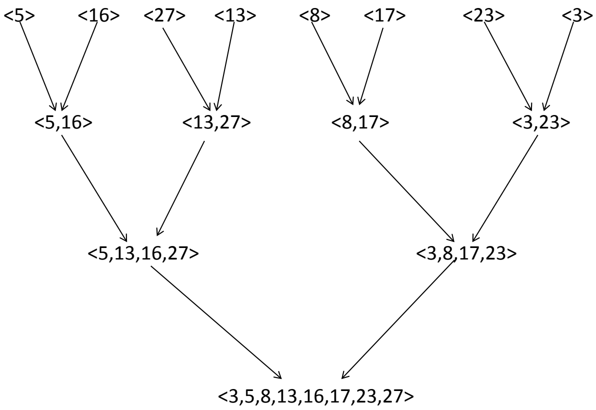
$$T(n) = 2T(2n/3) + n \log n$$

Lösung. In der Notation des Mastertheorems haben wir $a = 2$, $b = \frac{3}{2}$ und $f(n) = n \log n$. Damit gilt $f(n) = \Theta(n \log n) = O(n^{\log_b a - \epsilon})$ für $\epsilon = \log_{3/2} 2 - 1 \approx 0,7$ (2 Punkte). Damit können wir Fall 1 des Mastertheorems anwenden und erhalten $T(n) = \Theta(n^{\log_b a}) = \Theta(n^{1,709\ldots})$ (2 Punkte).

K4 Sortieren (7 Punkte)

Demonstrieren Sie die Arbeitsweise von Merge-Sort anhand des Arrays $A = \langle 5, 16, 27, 13, 8, 17, 23, 3\rangle$ unter Verwendung der Zeichnung. Tragen Sie unter die Pfeile jeweils das Array ein, das durch Anwendung von Merge entsteht.

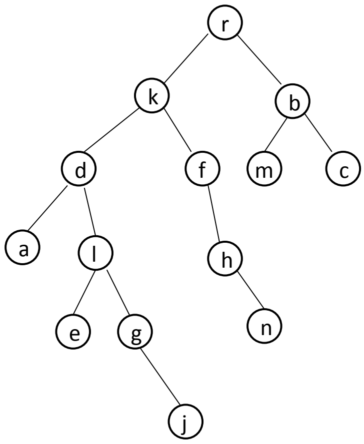




Lösung.
(jeweils 1 Punkt; insgesamt 7 Punkte)

K5 Bäume	(9 Punkte)

Betrachten Sie den untenstehenden binären Baum T mit Wurzel r.



a) Geben Sie die Höhe des Baums sowie die des Knotens f an.

b) Geben Sie das linke Kind des Knotens r sowie das rechte Kind des Knotens l an.

c) Geben Sie den Grad der Knoten f und j an.

d) Wie heißt das Elter der Knoten g bzw. h?

e) Geben Sie alle Blätter an, die im Teilbaum mit Wurzel d enthalten sind.

f) Geben Sie das Geschwister des Knotens m an.

g) Wie viele Knoten müssen zum Baum hinzugefügt werden, damit ein vollständiger binärer Baum entsteht (die Höhe bleibt gleich)?

Lösung.

a) Die Höhe des Baums ist die der Wurzel r. Diese ist 5. Die Höhe von f ist 2. (jeweils 1/2 Punkt; insgesamt 1 Punkt)

b) Das linke Kind von r ist k, das rechte Kind von l ist g (jeweils 1/2 Punkt; insgesamt 1 Punkt)

c) Der Grad von f ist 1, der von j ist 0. (je 1/2 Punkt; insgesamt 1 Punkt).

d) Das Elter von g ist l, das von h ist f. (je 1/2 Punkt; insgesamt 1 Punkt)

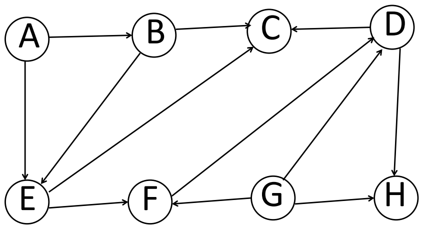
e) Der Teilbaum mit Wurzel d enthält 3 Blätter: a, e und j. (je 1/2 Punkt; insgesamt 1,5 Punkte)

f) Das Geschwister von m ist c. (1/2 Punkt)

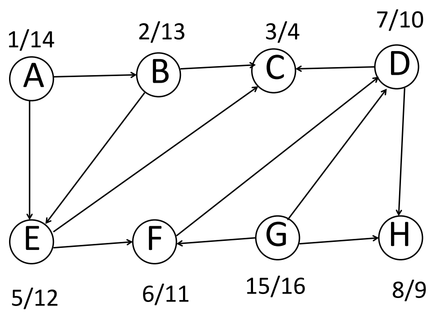
g) Ein vollständiger binärer Baum der Höhe 5 hat $2^{5+1} - 1 = 63$ Knoten. Da der obige Baum 14 Knoten enthält, müssen demnach noch 49 hinzugefügt werden. (3 Punkte)

K6 Tiefensuche und topologische Sortierung (10 Punkte)

Führen Sie auf untenstehendem Graphen eine Tiefensuche durch. Beginnen Sie beim Knoten A und betrachten Sie die Knoten in alphabetischer Reihenfolge. Schreiben Sie über jeden Knoten die jeweilige discovery und finishing time. Benutzen Sie Ihr Ergebnis, um eine topologische Sortierung des Graphen anzugeben.



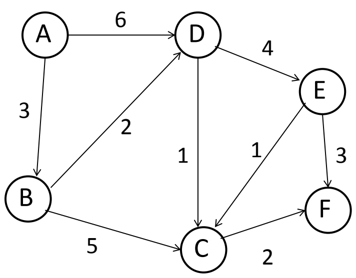
Lösung.



(je richtige Zahl 1/2 Punkt; insgesamt 8 Punkte)
Die topologische Sortierung lautet G, A, B, E, F, D, H, C (2 Punkte)

K7 Dijkstra (10 Punkte)

Benutzen Sie Dijkstra's Algorithmus, um im untenstehenden Graphen die kürzesten Pfade von A zu den übrigen Knoten zu ermitteln.
Der Algorithmus durchläuft nacheinander sämtliche Knoten des Graphen und fügt diese nach Bearbeitung zur Menge S hinzu. Ein Schritt entspricht dabei der Bearbeitung eines Knotens und der Aktualisierung der Menge S. Tragen Sie nach jedem Schritt für jeden Knoten den aktuellen Abstand d, seinen Vorgänger π und den aktuellen Zustand der Menge S in untenstehende Tabelle ein.



d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$	$\pi(E)$	$\pi(F)$	S
0	∞	∞	∞	∞	∞	nil	nil	nil	nil	nil	nil	$\emptyset$

Lösung.

d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	$\pi(A)$	$\pi(B)$	$\pi(C)$	$\pi(D)$	$\pi(E)$	$\pi(F)$	S
0	∞	∞	∞	∞	∞	nil	nil	nil	nil	nil	nil	$\emptyset$
	3		6				A		A			$\{A\}$
		8	5					B	B			$\{A, B\}$
		6		9				D		D		$\{A, B, D\}$
					8						C	$\{A, B, D, C\}$
												$\{A, B, D, C, F\}$
												$\{A, B, D, C, F, E\}$

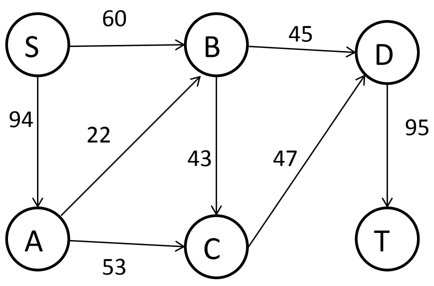
(jeweils 1/2 Punkt; insgesamt 10 Punkte)

K8 Maximale Flüsse (10 Punkte)

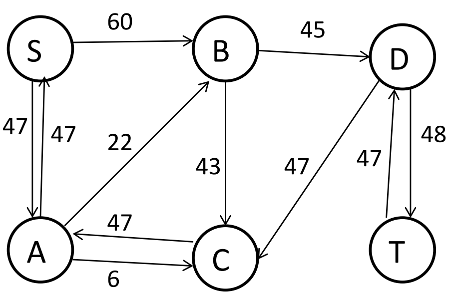
Bestimmen Sie mit dem Algorithmus von Ford-Fulkerson den maximalen Fluss in folgendem Flussnetzwerk mit Quelle S und Senke T. Der Algorithmus fügt in jedem Schritt einen ergänzenden Pfad zum aktuellen Flussnetzwerk hinzu. Dies wird so lange wiederholt, bis es keinen ergänzenden Pfad mehr gibt.

Geben Sie nach jedem Schritt den gerade hinzugefügten ergänzenden Pfad, seine Kapazität und den aktuellen Gesamtfluss an. Skizzieren Sie außerdem das entstehende Restnetzwerk. Fügen Sie dabei stets den ergänzenden Pfad mit maximaler Kapazität hinzu.

Beweisen Sie die Maximalität des gefundenen Flusses, indem Sie einen minimalen Cut in das in der Aufgabenstellung angegebene Netzwerk einzeichnen.

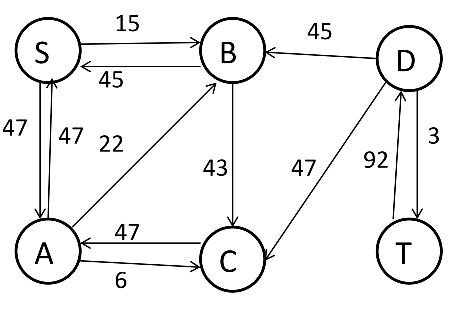


Lösung. Pfad S –> A –> C –> D –> T mit Kapazität 47 ⇒ Gesamtfluss 47 (1 Punkt)



(3 Punkte)

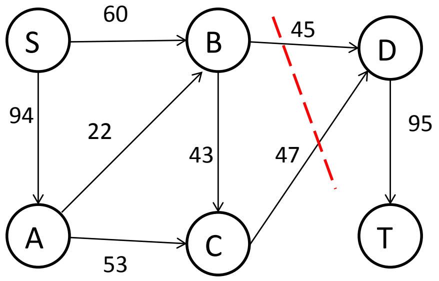
Pfad S –> B –> D –> T mit Kapazität 45 ⇒ Gesamtfluss 92 (1 Punkt)



(3 Punkte)

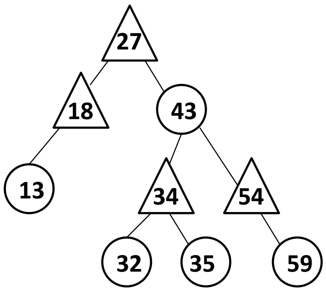
Da in obigem Restnetzwerk kein weiterer Fluss von S nach T mehr möglich ist, ist der maximale Gesamtfluss 92. (1 Punkt)

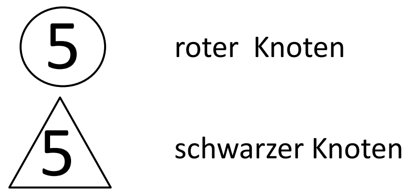
Minimaler Cut:



(1 Punkt)

Gegeben sei der folgende Rot-Schwarz-Baum.

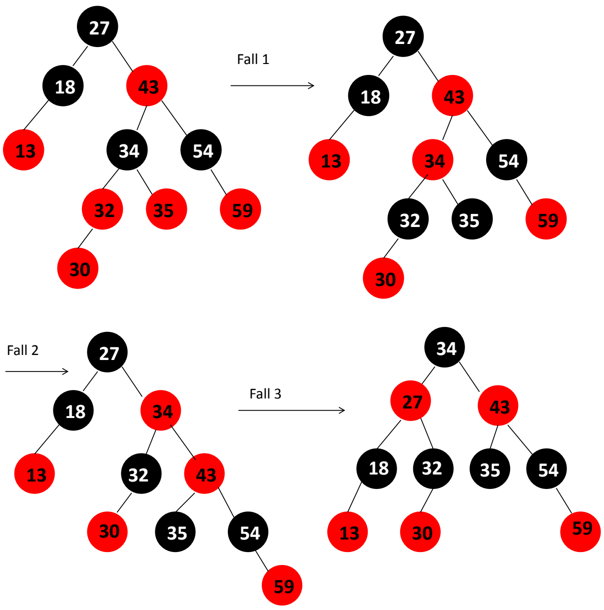




Führen Sie die Operation Rb-Insert(30) durch. Skizzieren Sie den unmittelbar nach dem Einfügen entstehenden Baum und den durch Rb-Insert-Fixup entstehenden Rot-Schwarz-Baum (inklusive aller Zwischenschritte). Geben Sie dabei an, welche der in der Vorlesung behandelten Fälle von RB-Insert-Fixup aufgerufen werden.

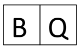
Stellen Sie dabei rote und schwarze Knoten wie in der obigen Abbildung gezeigt dar.

Lösung. Insert(30):

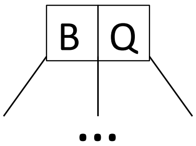


(1 Punkt fürs Einfügen (Bild 1); danach für jede Umformung + Angabe des jeweiligen Falles 3 Punkte; für jeden Fehler (falsche Farbe, falscher Schlüssel etc) ein Punkt Abzug)

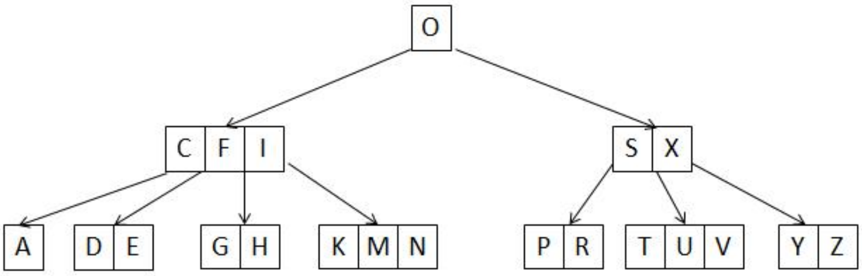
Führen Sie auf untenstehendem B-Baum mit Minimalgrad $t = 2$ die Operation Insert(J) durch.

Skizzieren Sie den entstehenden B-Baum nach dem Einfügevorgang. Dabei können Sie einen Teilbaum mit Wurzel 

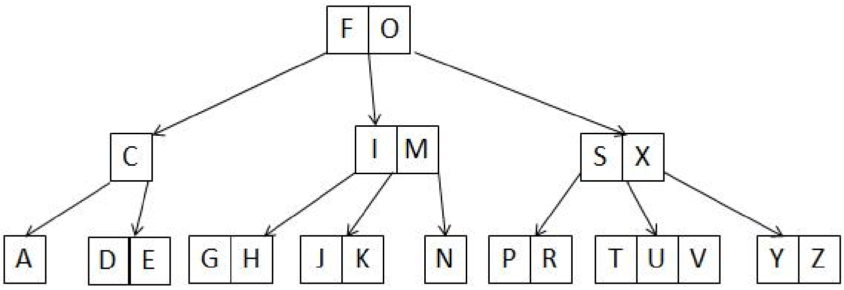
, der sich nicht verändert hat, wie folgt darstellen.



Geben Sie an, ob und gegebenenfalls welche Knoten beim Einfügen gesplittet werden müssen.



Lösung. Einfügen von J: Knoten CFI und KMN müssen gesplittet werden. (2 Punkte)



(3 Punkte fürs Bild; für jeden Fehler (Schlüssel im falschen Knoten etc.) ein Punkt Abzug)

K11 Offene Adressierung (5 Punkte)

Betrachten Sie eine Hashtabelle der Länge $m = 5$. Die Hashtabelle soll offene Adressierung und lineares Sondieren verwenden. Benutzen Sie als Hashfunktion

$$h(k,i) = (\lfloor 5 \cdot (0.61 \cdot k \bmod 1) \rfloor + 3 \cdot i) \bmod 5.$$

Tragen Sie die Schlüssel 15, 9 und 19 in dieser Reihenfolge in untenstehende Hashtabelle ein. Geben Sie jeweils an, für welchen Wert von i das Eintragen erfolgreich ist.

0	
1	
2	
3	
4	

Lösung.
Zunächst wählen wir immer $i = 0$.

$h(15,0) = (\lfloor 5 \cdot (0,61 \cdot 15 \bmod 1) \rfloor + 0) \bmod 5 = 0$. Der Platz 0 ist frei und der Schlüssel 15 wird dort eingetragen.

$h(9,0) = (\lfloor 5 \cdot (0,61 \cdot 9 \bmod 1) \rfloor + 0) \bmod 5 = 2$. Der Platz 2 ist frei und der Schlüssel 9 wird dort eingetragen.

$h(19,0) = (\lfloor 5 \cdot (0,61 \cdot 19 \bmod 1) \rfloor + 0) \bmod 5 = 2$. Der Platz 2 ist belegt. Setze $i = 1$.
$h(19,1) = (\lfloor 5 \cdot (0,61 \cdot 19 \bmod 1) \rfloor + 3) \bmod 5 = 0$. Der Platz 0 ist belegt. Setze $i = 2$.
$h(19,2) = (\lfloor 5 \cdot (0,61 \cdot 19 \bmod 1) \rfloor + 3 \cdot 2) \bmod 5 = 3$. Der Platz 3 ist frei und der Schlüssel 19 wird dort eingetragen.

(je 1 Punkt pro Hashwert + Folgerung; insgesamt 5 Punkte)

0	15
1	
2	9
3	19
4	

K12 Fouriertransformation (10 Punkte)

Berechnen Sie die inverse diskrete Fouriertransformation des Vektors $y = (1,2,3,4)$.

Lösung. Da die Länge des Vektors y vier ist, benötigen wir die vierten Einheitswurzeln $\omega_4^0 = 1, \omega_4^1 = i, \omega_4^2 = -1, \omega_4^3 = -i$ (1 Punkt). Der Koeffizientenvektor a ergibt sich nach der inversen diskreten Fouriertransfomation als

$$a_k = \frac{1}{n} \sum_{j=0}^{n-1} y_j \omega_n^{-jk} \; (k = 0, \ldots, n-1). \;\; (2 \; Punkte)$$

Damit ergibt sich $a_0 = \frac{1}{4} \cdot \sum_{i=0}^{3} y_j = 2,5$. (1 Punkt)
$a_1 = \frac{1}{4} \cdot (1 + 2i^{-1} + 3i^{-2} + 4i^{-3}) = \frac{1}{4} \cdot (-2 + 2i) = -0,5 + 0,5i$. (2 Punkte)
$a_2 = \frac{1}{4} \cdot (1 + 2i^{-2} + 3i^{-4} + 4i^{-6}) = \frac{1}{4} \cdot (-2) = -0,5$. (2 Punkte)
$a_3 = \frac{1}{4} \cdot (1 + 2i^{-3} + 3i^{-6} + 4i^{-9}) = \frac{1}{4} \cdot (-2 - 2i) = -0,5 - 0,5i$. (2 Punkte)

Damit haben wir $a = (5/2, -1/2 + 1/2i, -1/2, -1/2 - 1/2i)$.