

# Datenstrukturen und Algorithmen

## Schriftliche Prüfung, Frühling 2009

27. Mai 2009

**Name, Vorname:**

**Matrikelnummer:**

**Hinweise:**

1. Diese Prüfung beinhaltet 13 Fragen und gibt total 52 Punkte. Wir rechnen nicht damit, dass Sie alles lösen. Für die Bestnote brauchen Sie bei weitem nicht alle Punkte!
2. Die Prüfung dauert 105 Minuten.
3. Als Hilfsmittel sind zwei A4 Seiten handschriftlicher Notizen erlaubt. Elektronische Geräte jeder Art sind nicht gestattet.
4. Schreiben Sie lesbar. Wir bewerten nur, was wir klar lesen können.
5. Vergessen Sie nicht, Ihren Namen und Ihre Matrikelnummer auf Ihrer Abgabe zu vermerken.
6. Schreiben Sie Ihre Lösungen direkt auf die Aufgabenblätter.
7. Falls Sie Zusatzblätter verwenden, beginnen Sie für jede Aufgabe ein neues Blatt. Schreiben Sie Ihren Namen und Ihre Matrikelnummer auf alle Zusatzblätter!

---

Nur für Korrektur.

Aufgabe	1	2	3	4	5	6	7	8	9	10	11	12	13	$\Sigma$
Mögl. Pt.	3	3	8	6	2	3	3	4	2	3	3	4	8	52
Punkte														

# Garantiert in Prüfung

1. Ordnen Sie die folgenden Funktionen nach ihrer asymptotischen Wachstumsrate:

(a)  $2^n \rightarrow \Theta(2^n)$

(b)  $n^3 \rightarrow \Theta(n^3)$

(c)  $2^{\log_2 n} \rightarrow \Theta(n)$

(d)  $12n + 10 \log n \rightarrow \Theta(n)$

(e)  $5n \rightarrow \Theta(n)$

(f)  $n^2 + 100n \rightarrow \Theta(n^2)$

(g)  $n \log n \rightarrow \Theta(n \log n)$

(h)  $7n \log n + 2 \rightarrow \Theta(n \log n)$

(i)  $2^{16} \rightarrow \Theta(1)$

1. (i)

2. (c), (d), (e)

3. (g), (h)

4. (f)

5. (b)

6. (a)

3 Punkte

2. Vergleichen Sie je die beiden Zeitkomplexitäten:

(a)  $f(n) = \sqrt{n}, g(n) = n \log n$

(b)  $f(n) = n^4, g(n) = n^4 + 3n^2$

(c)  $f(n) = n^{1/3}, g(n) = \log n$

Welche der folgenden Gleichungen gilt je:  $f = O(g), f = \Omega(g), f = \Theta(g)$ ? **3 Punkte**

a)  $f = O(g)$ , d.h.  $f(n) \leq c \cdot g(n)$ ,  $n \geq n_0$

b)  $f = \Theta(g)$ , d.h.  $f(n) = \Omega(g(n))$  und  $f(n) = O(g(n))$

c)  $f = \Omega(g)$ , d.h.  $f(n) \geq c \cdot g(n)$ ,  $n \geq n_0$

← Auch in Prüfung

3. Gegeben sei die folgende Rekursionsgleichung:

$$T(n) = \begin{cases} 4T(n/2) + n/4 + 2 & n > 1, \\ 4 & n = 1 \end{cases}$$

- a) Berechnen Sie das asymptotische Wachstum von  $T(n)$  mit dem Mastertheorem. **1 Punkt**
- b) Skizzieren Sie einen Rekursionsbaum für  $T(n)$ . Nehmen Sie an, dass  $n$  eine Zweierpotenz ist und geben Sie die Höhe des Baumes an. **3 Punkte**
- c) Leiten Sie eine explizite Formel für  $T(n)$  her, indem Sie die Kosten des Rekursionsbaumes aufsummieren. Benutzen Sie dann die Tatsache, dass

$$\sum_{i=0}^k q^i = \frac{q^{k+1} - 1}{q - 1},$$

um die Summen aufzulösen und eine direkte Gleichung für  $T(n)$  zu erhalten.

**3 Punkte**

- d) Beweisen Sie Ihre Gleichung für  $T(n)$  durch Induktion. **1 Punkt**

a)  $T(n) = \Theta(n^2)$



4. Geben Sie für folgende Algorithmen die asymptotische Laufzeit in Abhängigkeit von  $n$  an. Verwenden Sie die Theta-Notation. Begründen Sie Ihre Antwort kurz in Worten oder geben Sie eine Summenformel für die Laufzeit an.

a) 1  $i \leftarrow 1$   
2 **while**  $i < n/4$   
3     **do**  
4          $j \leftarrow 2i$   
5         **while**  $j < n$   
6         **do**  
7              $j \leftarrow j + 1$   
8          $i \leftarrow i + 1$

**2 Punkte**

b) 1  $i \leftarrow n$   
2 **while**  $i > 1$   
3     **do**  
4          $j \leftarrow i$   
5         **while**  $j < n$   
6         **do**  
7              $j \leftarrow 2j$   
8          $i \leftarrow i - 1$

**2 Punkte**

c) 1  $i \leftarrow 1$   
2 **while**  $i < n$   
3     **do**  
4          $j \leftarrow 0$   
5         **while**  $j \leq i$   
6         **do**  
7              $j \leftarrow j + 1$   
8          $i \leftarrow 2i$

Hinweis: Verwenden Sie die Summenformel  $\sum_{i=0}^k q^i = \frac{q^{k+1}-1}{q-1}$ . **2 Punkte**

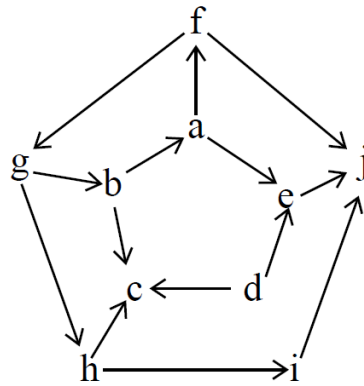


5. Der folgende gerichtete Graph wird mit Breitensuche traversiert. Die Suche startet beim Knoten *a*.

a) Geben Sie eine Reihenfolge an, in der die Knoten erreicht werden können.

**1 Punkt**

b) Skizzieren Sie den entsprechenden Breitensuchbaum. **1 Punkt**



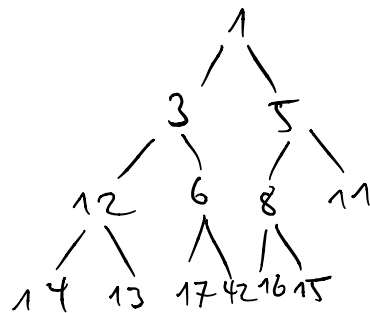


6. In der folgenden Tabelle ist ein Min-Heap in der üblichen Form gespeichert.

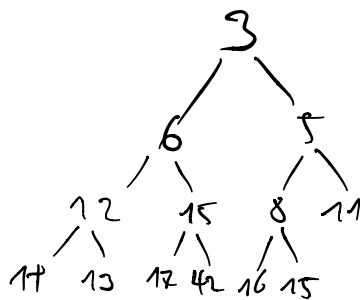
[1, 3, 5, 12, 6, 8, 11, 14, 13, 17, 42, 16, 15]

- Skizzieren Sie den Heap als binären Baum. **1 Punkt**
- Wie sieht die Tabelle aus, nachdem das kleinste Element entfernt wurde und die Heap-Bedingung wieder hergestellt wurde? **1 Punkt**
- Skizzieren Sie wieder den entsprechenden binären Baum, nachdem das Element entfernt wurde. **1 Punkt**

a)



b)



[3, 6, 5, 12, 15, 8, 11, 14, 13, 17, 42, 16, 15]

c)



7. Die Methode  $Quicksort(A, p, r)$  sortiere die Elemente von Index  $p$  bis  $r$  des Feldes  $A$  mittels des Quicksort Algorithmus. Illustrieren Sie den Ablauf von  $Quicksort(A, 1, 10)$  anhand des Feldes  $A = [3, 7, 10, 4, 8, 9, 11, 6, 12, 2]$ , indem Sie den Ablauf unten vervollständigen.

Geben Sie jeden rekursiven Aufruf von  $Quicksort$  mit seinen Parametern an, und schreiben Sie jeweils das Feld  $A$  auf, nachdem das Feld mit Hilfe des Pivotelements zerlegt wurde. Nehmen Sie an, in jedem Schritt werde das Element an der Stelle  $r$  als Pivot-Element verwendet. **3 Punkte**

$Quicksort(A, 1, 10)$

Feld  $A$  nach Zerlegung mit Pivot  $A[10]$ :

$Quicksort(A, \dots, \dots)$

Feld  $A$  nach Zerlegung mit Pivot  $A[\dots]$ :

...

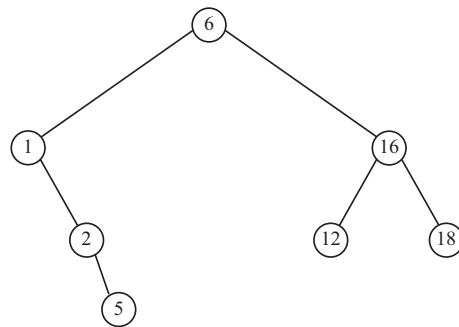
$A = [3, 7, 10, 4, 8, 9, 11, 6, 12, 2]$   
 $Quicksort(A, 1, 10) \text{ Pivot}[10]$

$[2, 7, 10, 4, 8, 9, 11, 6, 12, 3]$   $q = A[1]$

$Quicksort(A, 2, 10) \text{ Pivot}[10]$

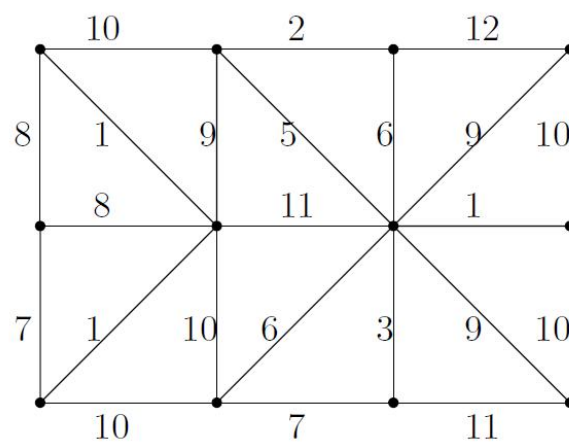
$[2, 3, 10, 4, 8, 9, 11, 6, 12, 7]$   
 $Quicksort(A, 1, 1)$

8. Gegeben ist der binäre Suchbaum in der Figur unten.



- a) Geben Sie die Reihenfolge der Knoten in der Inorder und der Preorder-Traversierung.  
**2 Punkte**
- c) Zeichnen Sie den binären Suchbaum, dessen Postorder-Traversierung die Folge 1, 4, 5, 3, 2, 6, 9, 11, 10, 8, 7, 13, 12 ergibt. Hinweis: Überlegen Sie sich zuerst, welches die Wurzel des Baumes ist. **2 Punkte**

9. Markieren Sie im untenstehenden gewichteten Graphen die Kanten eines minimalen Spannbaums. Geben Sie den Namen des Algorithmus an, den Sie verwenden, und schreiben Sie die Reihenfolge auf, in der Sie die Kanten zum Spannbaum hinzufügen.
- 2 Punkte**



10. Gegeben sei folgende Zeichenkette bestehend aus den Zeichen  $a, b, c, d, e, f, g$ :

f b a g a a g c d d e a g g f f g

Konstruieren Sie einen Huffman-Code für diese Zeichenkette. Stellen Sie den binären Codierungsbaum dar, und geben Sie für jedes Zeichen den Binärcode. Sie müssen nicht den gesamten Code für die Zeichenkette aufschreiben. **3 Punkte**

$a:4 \quad b:1 \quad c:1 \quad d:2 \quad e:1 \quad f:3 \quad g:6$



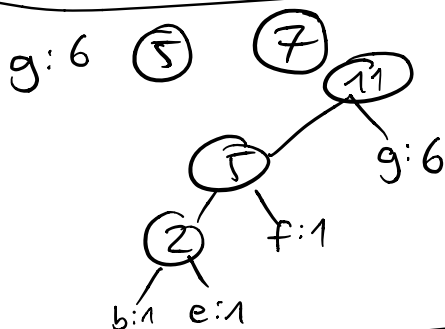
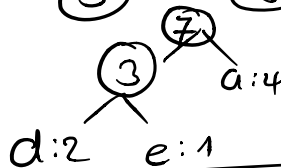
$a:4 \quad d:2 \quad e:1 \quad f:3 \quad g:6 \quad (2)$



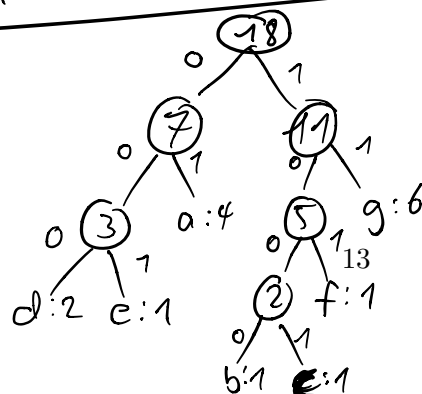
$a:4 \quad f:3 \quad g:6 \quad (2) \quad (3)$



$a:4 \quad g:6 \quad (3) \quad (5)$



$(7) \quad (11)$



$a = 01$      $b = 1000$      $c = 1001$      $d = 000$   
 $e = 001$      $f = 101$      $g = 11$

Vielleicht auch in Prüfung

11. Fügen Sie die Schlüssel 15, 19, 14, 7 in dieser Reihenfolge in die untenstehende Hashtabelle ein. Verwenden Sie doppeltes Hashing mit den Hilfshashfunktionen  $h_1(k) = (k \bmod 11)$  und  $h_2(k) = 1 + (k \bmod 9)$ . **3 Punkte**


11	45	24	47	04	38					
0	1	2	3	4	5	6	7	8	9	10

12. Gegeben seien zwei *zyklische, doppelt verkettete* Listen  $a$  und  $b$ . Nehmen Sie an, die Listen hätten je ein Wächterelement  $nil[a]$  und  $nil[b]$ . Geben Sie Pseudocode für eine Funktion  $concatenate(nil[a], nil[b])$ , welche der Liste  $a$  alle Elemente der Liste  $b$  anhängt. Nach Aufruf von  $concatenate$  besteht also die Liste  $a$  aus den Elementen von  $a$  gefolgt von den Elementen von  $b$ . Das Wächterelement  $nil[b]$  soll natürlich nicht in  $a$  eingefügt werden. Behandeln Sie die Fälle korrekt, wo  $a$  und/oder  $b$  keine Elemente ausser dem Wächter enthalten. **4 Punkte**



13. Ein Taxifahrer in New York hat sich in ein gefährliches Quartier verfahren. Er befindet sich in der linken oberen Ecke des Strassengitters, das unten dargestellt ist. Das Ziel des Taxifahrers ist es, in die untere rechte Ecke zu gelangen, wo er das gefährliche Quartier verlässt. Jeder Strassenabschnitt hat ein gewisses Risiko für einen Überfall, das mit ganzen Zahlen angegeben ist. Alle Strassen sind Einbahnstrassen, welche von links nach rechts oder von oben nach unten verlaufen. Gesucht ist ein Weg, der das Gesamtrisiko für einen Überfall, also die Summe der Risiken aller passierten Strassenabschnitte, minimiert.

Nehmen Sie an, die Risiken seien in einem zweidimensionalen Feld  $r$  gespeichert. Das heisst  $r[i, j]$  ist das Risiko für den Strassenabschnitt in Zeile  $i$  und Spalte  $j$  des Strassengitters. Die Startposition ist also  $i = 1, j = 1$ , und die Zielposition  $i = 5, j = 7$ .

	9	1	2	3	5	7
8	3	3	11	2	9	8
12	9	4	9	2	14	1
7	10	8	16	3	5	9
1	2	2	12	15	8	

- Erstellen Sie ein rekursives Programm in Pseudocode, welches das Gesamtrisiko eines sichersten Weges berechnet. Geben Sie die Laufzeit des rekursiven Algorithmus an. **2 Punkte**
- Entwerfen Sie einen Algorithmus nach dem Muster der dynamischen Programmierung, der das Gesamtrisiko eines sichersten Weges berechnet. Geben Sie die Laufzeit Ihres Algorithmus an. **3 Punkte**
- Beschreiben Sie in Worten, wie durch Rückverfolgung in der Lösungstabelle, die Ihr Algorithmus aus b) berechnet, die traversierten Zellen eines sichersten Weges gefunden werden können. Geben Sie die Laufzeit für die Rückverfolgung an. **2 Punkte**
- Kann das Problem immer noch mittels dynamischer Programmierung gelöst werden, wenn die Strassen in beide Richtungen befahrbar sind? Das würde heissen, der Taxifahrer könnte sich in jedem Schritt entweder um eine Zelle nach rechts, links, unten oder oben bewegen. Begründen Sie Ihre Antwort. **1 Punkt**

