Technische Universität Darmstadt
Fachgebiet Theoretische Informatik

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

Grundlagen der
Informatik 2
WS 2013/2014

TECHNISCHE UNIVERSITÄT DARMSTADT

Probeklausur — 20.07.2014

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

Studiengang: ..... Diplom ☐ Bachelor ☐ Master ☐

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

Prüfungssekretariat/Büro: ..... None ☐

Studienleistung erbracht? ..... Wenn ja, wann? .....

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

Unterschrift: .....

Aufgabe	K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	Σ
Mögliche Punkte	10	10	8	12	8	11	10	13	9	11	8	10	120
Erreichte Punkte													

Hinweise

- Diese Probeklausur hat den selben Umfang wie die Semestralklausur am 18. September. Sie sollten daher versuchen, die Aufgaben in einer Zeit von 120 Minuten zu bearbeiten.
- Um eine möglichst klausurnahe Atmosphäre zu erreichen, wird empfohlen, bei der Bearbeitung der Aufgaben nicht auf Bücher oder sonstige Mitschriften zurückzugreifen. Statt dessen empfehlen wir, bei dieser Probeklausur ein doppelseitig handbeschriebenes DIN-A4 Blatt zu verwenden, wie es auch bei der Semestralklausur als Hilfsmittel erlaubt ist.
- Für diese Probeklausur wird im moodle eine Musterlösung zur Selbstkorrektur bereitgestellt. Neben den Lösungshinweisen enthält diese auch eine ausführliche Beschreibung der Punktevergabe. Sie können also selbst ausrechnen, wie viele Punkte Sie erhalten hätten. Weiterhin wird ein Notenspiegel bereit gestellt, der Ihnen helfen soll, Ihre Leistung einzuschätzen.

K1 Verschiedenes (10 Punkte)

a) Sei $G = (V, E)$ ein ungerichteter Graph. Nennen sie zwei Bedingungen, die jeweils zu der Aussage "G ist ein Baum“ äquivalent sind.

b) Geben Sie die worst-case Laufzeit der folgenden Algorithmen in Abhängigkeit von der Eingabelänge n an.

INSERTION-SORT	
MERGE-SORT	
QUICKSORT	

c) Vervollständigen Sie folgenden Text.

Sei x ein Knoten in einem binären Suchbaum.
Ist y ein Knoten im linken Teilbaum von x, so gilt key$[y]$ _____ key$[x]$. (Wählen Sie hierbei aus $<$, $\leq$, $=$, $\geq$ und $>$.)

Ist y ein Knoten im rechten Teilbaum von x, so gilt key$[x]$ _____ key$[y]$. (Wählen Sie hierbei aus $<$, $\leq$, $=$, $\geq$ und $>$.)

Die Zahl der Schlüssel in einem inneren Knoten eines B-Baums mit Minimalgrad t ist mindestens _____

und höchstens _____.

Die Höhe eines B-Baums mit n Knoten und Minimalgrad t ist nach oben durch _____ beschränkt.

K2 Asymptotik (10 Punkte)

a) Vervollständigen Sie die folgenden Definitionen.

$$f(n) = \Omega(g(n)) \Leftrightarrow \exists c \in \mathbb{R}_{>0}, n_0 \in \mathbb{N} \text{ so dass } \forall n \geq n_0 \_\_\_\_\_$$

$$f(n) = o(g(n)) \Leftrightarrow \forall \_\_\_\_\_ \exists \_\_\_\_\_ \text{ so dass } \forall n \geq n_0 \_\_\_\_\_$$

b) Kreuzen Sie in der Tabelle an, ob $f = o(g)$, $f = \omega(g)$ oder $f = \Theta(g)$ gilt. In der Tabelle bezeichnet $k > 0$ eine Konstante. Bei der letzten Teilaufgabe gilt $n > 0$. In jeder Halbzeile ist nur eine Lösung richtig.

f(n)	**g(n)**	o	ω	Θ	**f(n)**	**g(n)**	o	ω	Θ
$\sqrt{\log n}$	$\log\log n$				n^n	120.000^n			
$2,5 \cdot n^3$	$e^{\ln e}$				$n^{\log k}$	$k^{\log n}$			
4^n	$n!$				$\log_2 n^n$	3^{n+2}			

K3 Rekurrenzgleichungen (8 Punkte)

Lösen sie folgende Rekurrenzgleichungen mit Hilfe des Mastertheorems.

a) $T(n) = 5T(n/8) + n \log n$

b) $T(n) = T(3n/5) + 1$

K4 Sortieren (12 Punkte)

a) Vervollständigen Sie den folgenden Algorithmus zu INSERTION-SORT. Der Algorithmus erhält als Eingabewert ein (unsortiertes) Array *A*. Beachten Sie: Das Array *A* beginnt mit Index 1.

INSERTION-SORT(A)

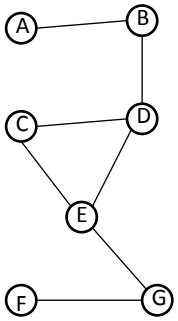
```
1 for j= _____ to length(A)
2   key = A[j]
3   i = _______
4   while i>0 and _____
5       _______________
6       i=i-1
7   _______________
```

b) Demonstrieren Sie die Arbeitsweise von Quicksort anhand des Arrays $A = \langle 18, 27, 34, 23, 12, 21, 29 \rangle$. Verwenden Sie dabei immer das letzte Element eines (Teil-)Arrays als Pivotelement. Verwenden Sie die Baumstruktur, wie sie auch in den Übungen und den Vorlesungsfolien verwendet wurde. Geben Sie das sortierte Array an.

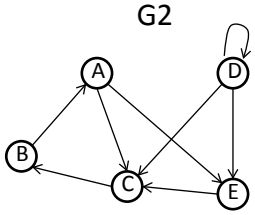
K5 Graphen (8 Punkte)

Gegeben seien die folgenden zwei Graphen.

G1



G2



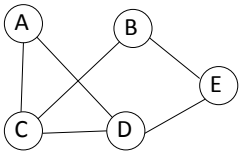
a) Ist G1 ein Baum? Wenn nein, warum nicht? Wie kann man G1 (unter Beibehaltung der Knoten) verändern, damit es zu einem Baum wird?

b) Wie viele Möglichkeiten gibt es, G1 durch Entfernen einer Kante zu einem bipartiten Graph zu machen? Welche sind das? Geben sie jeweils die beiden disjunkten Teilmengen von Knoten an.

c) Wie viele Kanten muss man zu G2 hinzufügen, damit ein stark zusammenhängender Graph entsteht? Begründung?

K6 Breitensuche (11 Punkte)

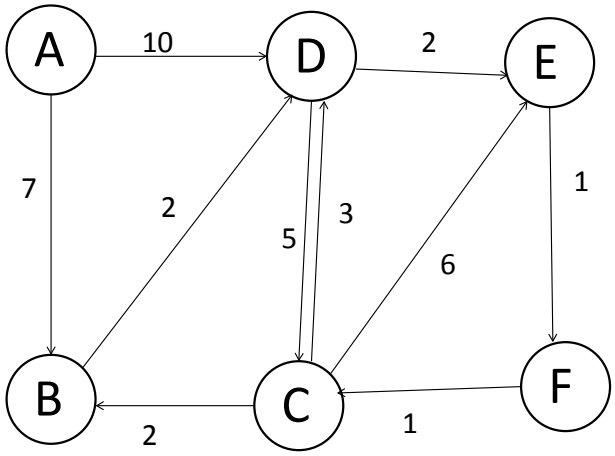
Führen Sie auf dem folgenden Graphen eine Breitensuche aus. Der Algorithmus arbeitet sämtliche Knoten nacheinander ab und färbt diese schließlich schwarz.
Tragen Sie die Werte von $d[v]$ (Abstand), $\pi[v]$ (Vorgänger) (für alle Knoten v), Q (Warteschlange) und u zu jedem Zeitpunkt, zu dem ein Knoten u schwarz gefärbt wird, in die folgende Tabelle ein. Jede Zeile entspricht dabei einem Iterationsschritt.
Benutzen Sie dabei Knoten A als Startknoten und nehmen Sie an, dass die Knoten in den Adjazenzlisten alphabetisch sortiert sind. Geben Sie den durch den Algorithmus erzeugten Vorgängerbaum an, indem Sie die entsprechenden Kanten im untenstehenden Graphen markieren.



$d[A]$	$d[B]$	$d[C]$	$d[D]$	$d[E]$	$\pi[A]$	$\pi[B]$	$\pi[C]$	$\pi[D]$	$\pi[E]$	Q	u
0	∞	∞	∞	∞	nil	nil	nil	nil	nil	$\{1\}$	–

K7 Dijkstra's Algorithmus (10 Punkte)

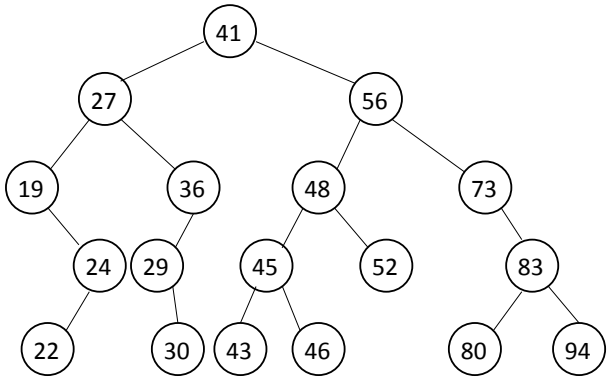
Führen Sie Dijkstras Algorithmus auf folgendem Graphen aus. Benutzen Sie dabei den Knoten A als Startknoten. Im Verlauf des Algorithmus werden die Knoten nacheinander zur Menge S hinzugefügt. Tragen Sie die Werte von d (Abstand), π (Vorgänger) und S (bereits abschließend betrachtete Knoten) unmittelbar nachdem ein neuer Knoten zu S hinzugefügt wurde, in nachstehende Tabelle ein. Jede Tabellenzeile entspricht dabei einem Iterationsschritt.



$d[A]$	$d[B]$	$d[C]$	$d[D]$	$d[E]$	$d[F]$	$\pi[A]$	$\pi[B]$	$\pi[C]$	$\pi[D]$	$\pi[E]$	$\pi[F]$	S
0	∞	∞	∞	∞	∞	nil	nil	nil	nil	nil	nil	$\emptyset$

K8 Binäre Suchbäume (9 Punkte)

Betrachten Sie den folgenden binären Suchbaum.

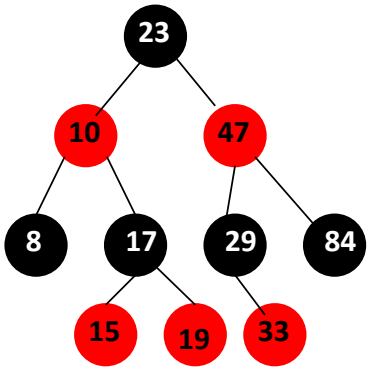


(a) Fügen Sie die Knoten z_1, z_2, z_3 mit $key[z_1] = 16, key[z_2] = 26, key[z_3] = 54$ in dieser Reihenfolge in den oben gezeigten Suchbaum ein. Skizzieren Sie den Baum, nachdem die drei Schlüssel eingefügt sind. Sie können dafür die drei Knoten in den obigen Baum einzeichnen.

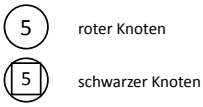
(b) Löschen Sie die Knoten z_4 und z_5 mit $key[z_4] = 29$ und $key[z_5] = 48$ in dieser Reihenfolge aus dem aus a.) gewonnenen Suchbaum. Skizzieren Sie den Baum nach jedem Löschvorgang. Dabei können Sie Teilbäume mit Wurzel *A*, die sich nicht verändert haben, in der Form  darstellen.

K9 Rot-Schwarz-Bäume (11 Punkte)

Gegeben sei der folgende Rot-Schwarz-Baum T.



Führen Sie auf dem Baum hintereinander die Operationen RB-Insert(T,13) und RB-Delete(T,8) durch.
Geben Sie jeweils an, ob und gegebenenfalls welche Fälle von RB-Insert-Fixup bzw. RB-Delete-Fixup in welcher Reihenfolge aufgerufen werden. Skizzieren Sie den Baum nach jeder Einfüge- bzw. Löschoperation. Stellen Sie dabei rote und schwarze Knoten folgendermaßen dar.



K10 Hashtabellen (8 Punkte)

Fügen Sie die Schlüssel $13, 28, 16, 9, 14$ und 43 in dieser Reihenfolge in eine Hashtabelle der Länge 5 ein. Benutzen Sie doppelt verkettete Listen, um Kollisionen zu behandeln. Benutzen Sie als Hashfunktion $h(k) = k \bmod 5$.

K11 Diskrete Fouriertransfomation (10 Punkte)

Berechnen Sie die diskrete Fouriertransformation des Vektors $a = (0, 1, 2, 3)$.