Institut für Informatik I
Prof. Dr. Heiko Röglin
Tobias Brunsch

universitätbonn

Algorithmen und
Berechnungskomplexität I
Wintersemester 2010/11

Probeklausur

Name:

Matrikelnummer:

Studiengang:

Übungsgruppe:

◯	Montag, 13-15	◯	Mittwoch, 13-15
◯	Donnerstag, 11-13	◯	Donnerstag, 13-15
◯	Freitag, 11-13	◯	Freitag, 13-15

Bitte beachten Sie folgende Hinweise:

1. Tragen Sie bitte oben Ihren Namen, Ihre Matrikelnummer und Ihren Studiengang ein und kreuzen Sie Ihre Übungsgruppe an. Schreiben Sie Ihren Namen und Ihre Matrikelnummer außerdem auf jedes Blatt der Klausur.
2. Die Dauer der Klausur beträgt 90 Minuten.
3. Es sind **keinerlei Hilfsmittel** zugelassen. Täuschungsversuche jeglicher Art führen zur Bewertung der Probeklausur mit 0 Punkten.
4. Verwenden Sie keine eigenen Zettel. Sollte der vorgesehene Platz nicht ausreichen, benutzen Sie bitte die Rückseite.
5. Es können maximal 48 Punkte erreicht werden. Die Hälfte der erreichten Punkte geht als Zusatzpunkte für die Übungen ein.
6. In der gesamten Klausur werden die Laufzeiten im uniformen Kostenmaß betrachtet.

Von den Korrektoren auszufüllen:

Aufgabe	**1**	**2**	**3**	**4**	$\sum$
Punkte					

Note:

Aufgabe 1

2+2+3+3+4=14 Punkte

(a) Prof. G. Witzt hat mit Hilfe neuester Erkenntnisse aus der Theoretischen Informatik ein vergleichsbasiertes Sortierverfahren mit Laufzeit $O(n \log \log n)$ entworfen. Glauben Sie ihm? Begründen Sie Ihre Antwort.

(b) Geben Sie eine möglichst einfache Funktion f an, für die $\sum_{i=1}^{n} \frac{1}{i} = \Theta(f(n))$ gilt.

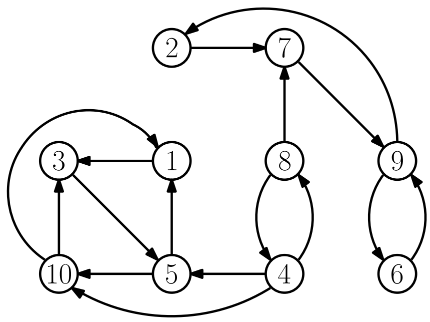
(c) Wir betrachten einen binären Suchbaum mit den Schlüsseln $1, \ldots, 1000$ und suchen nach dem Schlüssel 220. Die Schlüssel k_i, die wir uns während der Suche anschauen, bilden eine Sequenz $k_1, \ldots, k_m$. Welche der folgenden Sequenzen können bei der Suche **nicht** entstehen?

(1) $7, 13, 563, 561, 27, 144, 496, 220$

(2) $28, 42, 561, 127, 108, 496, 144, 220$

(3) $563, 28, 127, 561, 108, 496, 144, 220$

(d) Bestimmen Sie die starken Zusammenhangskomponenten des unten dargestellten gerichteten Graphen.



(e) Geben Sie die worst-case Laufzeit des Algorithmus von Kruskal in Θ-Notation an. Wie ändert sich diese Laufzeit, wenn man eine *Union*-*Find*-Datenstruktur verwendet, die für jede Folge von m *Union*-Befehlen und f *Find*-Befehlen nur Zeit $O(m \log\log m + f)$ benötigt? Begründen Sie Ihre Antwort.

Aufgabe 2

6+4=10 Punkte

(a) Gegeben sei ein Feld A mit den Einträgen $A[1], \dots, A[n]$ und eine Funktion `Swap`, die als Parameter einen Index i zwischen 1 und $n-1$ erhält und wie folgt implementiert ist.

`Swap`(i)
{
  if $A[i] > A[i+1]$ **then** Vertausche $A[i]$ und $A[i+1]$.
}

Zeigen Sie, dass jedes Sortierverfahren, welches Feld A nur mit Hilfe der `Swap`-Funktion verändert, genau $\chi(A)$ Vertauschungen durchführt, wobei

$$\chi(A) := |\{(i,j) :\ 1 \leq i < j \leq n \text{ und } A[i] > A[j]\}|$$

die Anzahl der Inversionen in A ist. Was lässt sich über die worst-case Laufzeit solcher Sortierverfahren schlussfolgern?

(b) Kann man ein Feld A mit n Einträgen aus $\{1, \ldots, n\}$ in Zeit $O(n)$ sortieren? Begründen Sie Ihre Antwort.

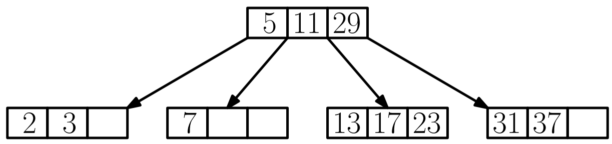
Aufgabe 3

4+4+4=12 Punkte

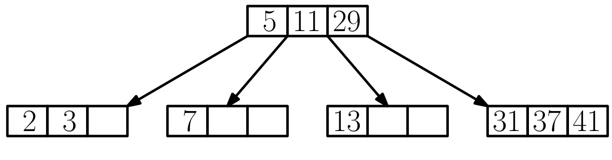
(a) Wie viele Knoten besitzt ein B-Baum der Ordnung $t \geq 2$ und der Höhe $h \geq 1$ mindestens? Es ist nicht notwendig, in den Formeln auftretende Summen zu berechnen.

Zur Erinnerung: Jeder Knoten eines B-Baumes der Ordnung t, außer der Wurzel, besitzt mindestens $t - 1$ und höchstens $2t - 1$ Schlüssel. Die Wurzel besitzt mindestens 1 und höchstens $2t - 1$ Schlüssel, sofern der Baum nichtleer ist.

(b) Fügen Sie den Schlüssel 19 in den unten abgebildeten B-Baum der Ordnung 2 ein.



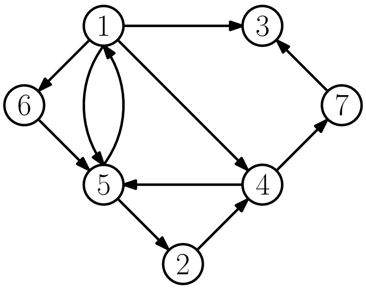
(c) Löschen Sie den Schlüssel 11 aus dem unten abgebildeten B-Baum der Ordnung 2.



Aufgabe 4

6+6=12 Punkte

(a) Führen Sie eine Tiefensuche auf dem unten abgebildeten Graphen durch. Beginnen Sie mit Knoten 1 und bevorzugen Sie bei der Auswahl des als nächstes zu betrachtenden Nachbarn eines Knotens stets den mit kleinerer Nummer. Ordnen Sie jede Kante ihrer Klasse (T-, F-, C- oder B-Kante) zu und bestimmen Sie die Reihenfolge, in der die Knoten besucht werden.



(b) Geben Sie einen Graphen $G = (V, E)$, eine Kantengewichtung $w\colon V \to \mathbb{R}$ und einen Knoten $s \in V$ an, sodass der Algorithmus von Dijkstra, ausgehend von Knoten s, nicht für alle Knoten $v \in V$ die Länge eines kürzesten Weges von s nach v bestimmt.

Hinweis: Wenn alle Kantengewichte positiv sind, dann arbeitet der Algorithmus von Dijkstra korrekt.