Prof. Dr. Martin Hoefer
Niklas Hahn

Frankfurt, den 09.10.2017

Modulabschlussprüfung
Datenstrukturen

WS 2017

Name: _____ Vorname: _____

Matrikelnummer: _____ Geburtsdatum: _____

Studiengang: _____

⇓ BITTE GENAU LESEN ⇓

Die Klausur besteht aus **9** Aufgaben. Die Klausur ist mit Sicherheit bestanden, wenn (zusammen mit der Bonifikation aus den Übungspunkten) mindestens **50%** der Höchstpunktzahl erreicht wird.

Bitte schreiben Sie oben auf **jeder** Seite in Blockschrift Ihren Namen und Ihre Matrikelnummer. Überprüfen Sie, ob die Klausur aus insgesamt **14** durchnummerierten Blättern besteht.

Es dürfen nur **dokumentenechte Stifte** in den Farben blau und schwarz verwendet werden. Zugelassene Hilfsmittel: 1 Blatt DIN A4 mit handschriftlichen Notizen (beidseitig).

Bitte beachten Sie, dass gemäß der Studienordnung das Mitbringen nicht zugelassener Hilfsmittel eine Täuschung darstellt und zum Nichtbestehen der Klausur führt. Schalten Sie bitte deshalb alle elektronischen Geräte, insbesondere **Handys** und **Smartwatches**, vor Beginn der Klausur aus und packen Sie diese weg.

Bitte benutzen Sie Rückseiten und die beigefügten Zusatzblätter. Weitere Blätter sind bei Bedarf erhältlich. Das Benutzen eigens mitgebrachter Blätter ist untersagt.

Bitte vermerken Sie dies entsprechend bei jeder Aufgabe, wenn sich Ihre Lösung dazu teilweise oder ganz auf Rückseiten oder Zusatzblätter befindet.

Werden zu einer Aufgabe zwei oder mehr Lösungen angegeben, so gilt die Aufgabe als nicht gelöst. Entscheiden Sie sich also immer für **eine** Lösung. Begründungen sind nur dann notwendig, wenn die Aufgabenformulierung dies verlangt.

Die Klausur dauert 100 Minuten.

Die Einsichtnahme in die Klausur ist am Freitag, den 20.10.2017 von 14-17 Uhr in Raum 307 (Robert-Mayer-Straße 11-15) möglich.

1	2	3	4	5	6	7	8	9	Σ
5	15	12	8	14	10	18	10	8	100

Note

Bonifikation	Σ

Name:	Matrikelnummer:

AUFGABE 1 — 5 Punkte

Ordnen Sie die folgenden Funktionen nach asymptotischem Wachstum, beginnend mit der am langsamsten wachsenden Funktion. Eine Begründung ist nicht erforderlich.

a) $a : \mathbb{N} \to \mathbb{R},\ a(n) = 2^{\log_4 n^8} \log_2 n$

b) $b : \mathbb{N} \to \mathbb{R},\ b(n) = n^3 \log_n 6$

c) $c : \mathbb{N} \to \mathbb{R},\ c(n) = n^3 \log_5 n$

d) $d : \mathbb{N} \to \mathbb{R},\ d(n) = \sum_{i=1}^{n} \left(\frac{1}{2^i} \log_2 n\right)$

e) $e : \mathbb{N} \to \mathbb{R},\ e(n) = \log_2(n!)$

langsamstes Wachstum						schnellstes Wachstum

Name:	Matrikelnummer:

AUFGABE 2 **3 + 3 + 3 + 3 + 3 = 15 Punkte**

Geben Sie jeweils eine kurze Antwort. Eine Begründung ist nicht erforderlich.

a) Geben Sie eine möglichst scharfe Laufzeitschranke für den Aufbau eines Heaps mit n Elementen an:

$\Theta($_____$)$

b) Geben Sie die kleinstmögliche Tiefe eines Binärbaums mit n Knoten an:

$\Theta($_____$)$

c) Gegeben ein gerichteter Graph $G = (V, E)$ in Adjazenzlistendarstellung mit n Knoten und m Kanten. Sie möchten überprüfen, ob der Graph azyklisch ist. Welches Verfahren aus der Vorlesung ist hierfür geeignet? Geben Sie eine möglichst scharfe Laufzeitschranke für ihren Algorithmus an.

Verfahren:_____ $\Theta($_____$)$

d) Gegeben ein ungerichteter Graph $G = (V, E)$ mit Kantengewichten sowie ein Startknoten $v \in V$. Ist der minimale Spannbaum von G immer der Baum der kürzesten Wege von v aus?

e) Gegeben eine Hashtabelle der festen Größe 12. Wählen Sie die geeignetste Familie von Hashfunktionen für $i = 0, ..., 11$ aus und markieren Sie sie mit X:

☐ $a_i(x) = \Big((x \bmod 11) + i \cdot (9 - (x \bmod 9))\Big) \bmod 11$

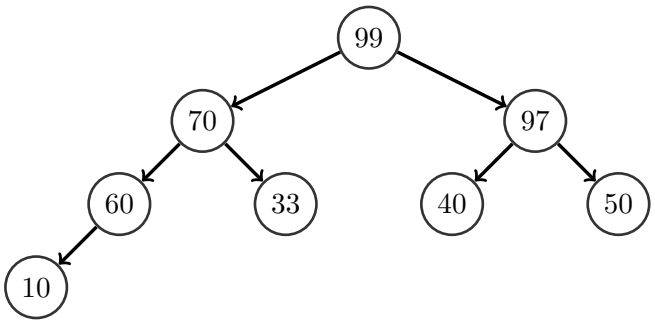
☐ $b_i(x) = \Big((x \bmod 12) + i \cdot (7 - (x \bmod 7))\Big) \bmod 12$

☐ $c_i(x) = \Big((x \bmod 13) + i \cdot (5 - (x \bmod 5))\Big) \bmod 13$

Name:	Matrikelnummer:

AUFGABE 3 **3 + 9 = 12 Punkte**

Gegeben Sie der folgende initiale Heap H:



a) Geben Sie die Arraydarstellung von H an. Eine Begründung ist nicht erforderlich.

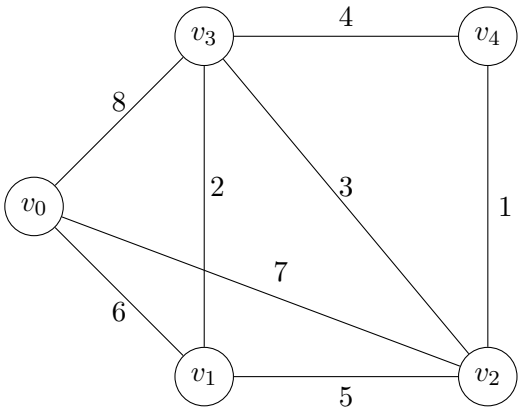
b) Führen Sie die folgenden Operationen auf H (in dieser Reihenfolge) aus. Geben Sie nach jeder Operation (und möglicherweise benötigten Reparaturen) den entstehenden Heap aus. Eine Begründung ist nicht erforderlich.

```
delete_max()
insert(85)
change_priority(3,35)
```

Name:	Matrikelnummer:

AUFGABE 4 **8 Punkte**

Bestimmen Sie mithilfe des Algorithmus von Kruskal den minimalen Spannbaum des folgenden Graphen. Geben Sie zusätzlich in jedem Schritt den Zustand der Union-Find-Datenstruktur an. Sie können dies entweder graphisch darstellen oder das Eltern-Array nutzen.
Wenn zwei Bäume derselben Größe vereinigt werden, soll der Knoten mit kleinerem Index die neue Wurzel werden. Eine Begründung ist nicht erforderlich.



Name:	Matrikelnummer:

AUFGABE 5 **3 + 6 + 5 = 14 Punkte**

Betrachten Sie das folgende Programm in Pseudocode:

```
 1    list func(int n):
 2        {
 3        liste = new list();
 4        bool tmp = True;
 5        for (int i=2; i<=n; i++)
 6            {
 7            tmp = True;
 8            for (int j=2; j<= sqrt(i); j++)  // sqrt(i) = √i
 9                {
10                if (i mod j == 0)
11                    { tmp = False; }
12                }
13            if (tmp == True)
14                { liste.insert(i); }
15            }
16        return liste;
17        }
```

a) Welche Ausgabe hat der Aufruf `func(6)`? Eine Begründung ist nicht erforderlich.

b) Was berechnet das Programm für allgemeines n? Begründen Sie Ihre Antwort unter Zuhilfenahme des Codes.

c) Welche Laufzeit hat das Programm? Geben Sie eine asymptotisch genaue Schranke in der Form $\Theta(\cdot)$ an. Beweisen Sie Ihre Antwort.

Name:	Matrikelnummer:

AUFGABE 6 **5 + 5 = 10 Punkte**

a) Bestimmen Sie die Laufzeit asymptotisch genau in der Form $\Theta(\cdot)$ in Abhängigkeit von n. Begründen Sie Ihre Antwort.

```
1    funcA(int n, string s1, string s2, string s3):
2    {
3      if (n==1)
4        { aux(s1, s2); }
5      else
6        {
7          funcA(n-1, s1, s3, s2);
8          aux(s1, s2);
9          funcA(n-1, s3, s2, s1);
10       }
11   }
```

Hinweis: Die Funktion `aux` benötigt nur konstante Laufzeit. Sie können davon ausgehen, dass die Reihenfolge der Parameter `s1`, `s2` und `s3` keinen Einfluss auf die Laufzeit von `funcA` hat.

Name:	Matrikelnummer:

b) Gegeben sei ein vollständiger Binärbaum mit n Knoten, bei dem jeder Knoten v einen positiven ganzzahligen Wert $v.wert$ besitzt. Das linke Kind ist über $v.links$ und das rechte Kind über $v.rechts$ erreichbar. Zusätzlich sei die Funktion $istBlatt(v)$ gegeben, die in konstanter Zeit bestimmt, ob v ein Blatt ist (Ausgabe True) oder nicht (Ausgabe False). Welche Laufzeit hat der Aufruf `treeFindMax(root)`, wenn `root` die Wurzel des Baums ist? Bestimmen Sie die Laufzeit asymptotisch genau in der Form $\Theta(\cdot)$ und begründen Sie Ihre Antwort.

```
1   treeFindMax(v):
2   {
3     l = -1;
4     r = -1;
5     if (istBlatt(v) == True)
6       { return v.value; }
7     if (v.links != null)
8       { l = treeFindMax(v.links); }
9     if (v.rechts != null)
10      { r = treeFindMax(v.rechts); }
11    return max(v.value, l, r);
12  }
```

Name:	Matrikelnummer:

AUFGABE 7 **5 + 3 + 5 + 5 = 18 Punkte**

Ein AVL-Bäum ist ein binärer Suchbaum mit der Balance-Eigenschaft:
Für jeden Knoten v und die Teilbäume T_1 und T_2 direkt unterhalb von v gilt

$$\text{Tiefe}(T_1) \leq \text{Tiefe}(T_2) + 1 \ .$$

Ein *k-AVL-Baum* ist ein binärer Suchbaum mit einer k-Balance-Eigenschaft:
Für jeden Knoten v und Teilbäume T_1 und T_2 direkt unterhalb von v gilt

$$\text{Tiefe}(T_1) \leq \text{Tiefe}(T_2) + k \ .$$

Hierbei ist $k \geq 2$ eine konstante natürliche Zahl, die nicht von der Anzahl der Knoten n im Baum abhängt.

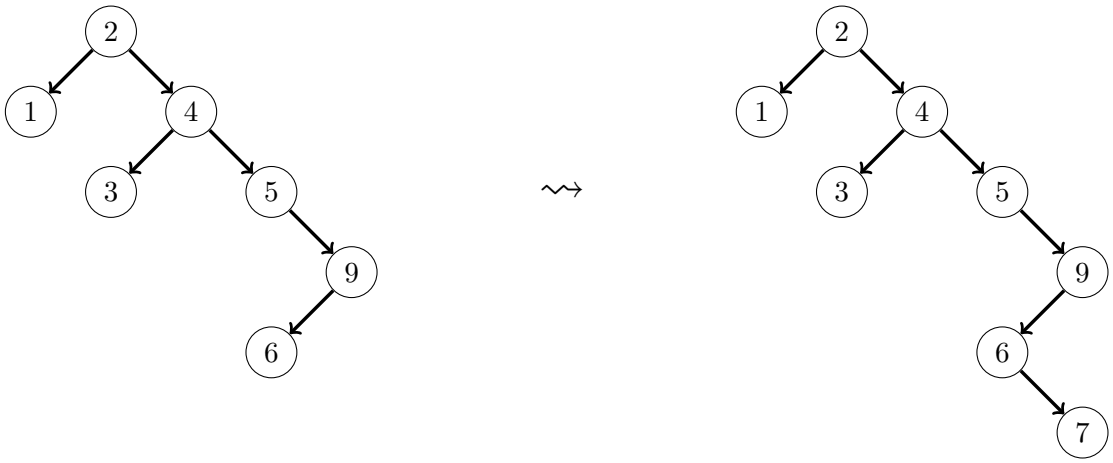
a) Zeigen Sie mit Induktion nach t, dass ein k-AVL-Baum B mit Tiefe t mindestens $2^{t/k}$ viele Knoten enthält.

b) Betrachten Sie die maximale Tiefe $t(n, k)$ eines k-AVL-Baumes mit n Knoten. Leiten Sie eine möglichst gute asymptotische obere Schranke für $t(n, k)$ her. Die Schranke soll eine Funktion von n und k sein.

Hinweis: Selbst wenn Sie Teilaufgabe a) nicht bearbeitet haben, können Sie die Aussage benutzen.

Name:	Matrikelnummer:

c) Betrachten Sie den folgenden 3-AVL-Baum, in den der Knoten 7 eingefügt wird.



Offensichtlich ist hier die 3-Balance-Eigenschaft verletzt, der Baum befindet sich im Analogon zum „Zick-Zick“-Fall.
Führen Sie eine Linksrotation in der Wurzel durch, um die 3-Balance-Eigenschaft wiederherzustellen und geben Sie das Ergebnis in graphischer Darstellung an. Eine Begründung ist nicht erforderlich.

Name:	Matrikelnummer:

d) Argumentieren Sie, warum die Linksrotation für alle $k \in \mathbb{N}_{\geq 1}$ den „analogen Zick-Zick“-Fall im k-AVL-Baum löst.

Name:	Matrikelnummer:

AUFGABE 8 — 10 Punkte

Gegeben sei ein zusammenhängender ungerichteter Graph $G = (V, E)$ mit paarweise verschiedenen Kantengewichten $c(e)$ für alle Kanten $e \in E$ sowie eine Kante $f \in E$.

Entwerfen Sie einen Algorithmus, der in Laufzeit $O(|V| + |E|)$ entscheidet, ob f im minimalen Spannbaum von G enthalten ist.

Zeigen Sie, dass die Laufzeitschranke eingehalten wird und begründen Sie die Korrektheit Ihres Algorithmus.

Hinweis: Leider ist es mit Kruskals Algorithmus nicht möglich, in linearer Laufzeit den Spannbaum aufzubauen. Nutzen Sie stattdessen das folgende Kriterium:

Eine Kante $e \in E$ ist genau dann nicht im minimalen Spannbaum enthalten, wenn es einen Kreis gibt, in dem e die Kante mit dem größten Kantengewicht ist.

Name:	Matrikelnummer:

AUFGABE 9 — 8 Punkte

Sei $G = (V, E)$ ein gerichteter Graph in Adjazenzlistendarstellung. Entwickeln Sie einen Algorithmus, der in Zeit $\mathcal{O}(|V| + |E|)$ bestimmt, ob der Graph stark zusammenhängend ist.

Zeigen Sie, dass die Laufzeitschranke eingehalten wird und begründen Sie die Korrektheit Ihres Algorithmus.

Name:	Matrikelnummer: