Hinweise zur Bearbeitung der Klausur zum Kurs 1661 Datenstrukturen I

Bitte **lesen Sie** sich diese Hinweise **vollständig und aufmerksam** durch, bevor Sie mit der Bearbeitung der Klausur beginnen. **Beachten Sie** insbesondere die **Punkte 6 bis 8!**

1. Prüfen Sie bitte die Vollständigkeit Ihrer Unterlagen. Die Klausur umfasst:
   - 2 Deckblätter
   - 1 Formblatt für eine Teilnahmebescheinigung zur Vorlage beim Finanzamt
   - diese Hinweise
   - 5 Aufgaben auf den Seiten 3 - 8
2. Die **Klausurdauer** beträgt **2 Stunden**.
3. Für die Klausur sind **keine Hilfsmittel zugelassen**. Es darf nur unbeschriebenes Konzeptpapier und Schreibzeug verwendet werden. Die Reinschrift der Klausur darf **nicht mit Bleistift** erfolgen.
4. Schreiben Sie Ihre Lösungen auf Ihr **eigenes Papier** (DIN A4) und nicht auf die Seiten mit den Aufgabenstellungen.
5. Bevor Sie mit der **Bearbeitung der Klausuraufgaben** beginnen, füllen Sie bitte die folgenden Teile der Klausur aus:
   - **sämtliche Deckblätter mit Name, Anschrift sowie Matrikelnummer.**
   - Schreiben Sie bitte **auf jedes weitere Blatt** oben links Ihren **Namen und** oben rechts Ihre **Matrikelnummer. Nummerieren Sie Ihre Seiten** bitte durch.
   - die Teilnahmebescheinigung, falls Sie diese wünschen.
6. **Wenden Sie** bei der Lösung der Aufgaben, soweit dies möglich ist, die **Algorithmen und Notationen aus den Kurseinheiten an.**
7. **Schreiben Sie,** wenn Algorithmen gefordert sind, **keine kompletten Java-Programme,** sondern beschränken Sie sich auf die wesentlichen Teile des Algorithmus, die z.T. auch in Prosa formuliert werden können. Formulieren Sie Algorithmen aber so, dass die elementaren Einzelschritte erkennbar werden.
   Sparen Sie bei solchen Aufgaben **nicht mit Kommentaren.** Wenn Ihre Lösung aufgrund fehlender Kommentare nicht verständlich ist, führt das zu Punktabzug.
8. **Vermeiden Sie** in jedem Fall **bei der Definition von Funktionen in Algebren Java-Programme sowie Algorithmen.** Geben Sie lediglich mathematische Definitionen an, wie sie im Kurstext verwandt worden sind!
   Ebenso sind **Mengendefinitionen in mathematischer Mengennotation** durchzuführen. Geben Sie auf **keinen** Fall **Datentypdefinitionen** an, und verwenden Sie **keine konkreten Datenstrukturen!**
9. **Vor der Abgabe** Ihrer Klausur:
   - **Heften** Sie Ihre Bearbeitung an Ihr vollständiges Klausurexemplar. **Die Aufgabenblätter müssen mit abgegeben werden!**
   - **Kreuzen** Sie auf beiden Deckblättern die von Ihnen **bearbeiteten Aufgaben an.**

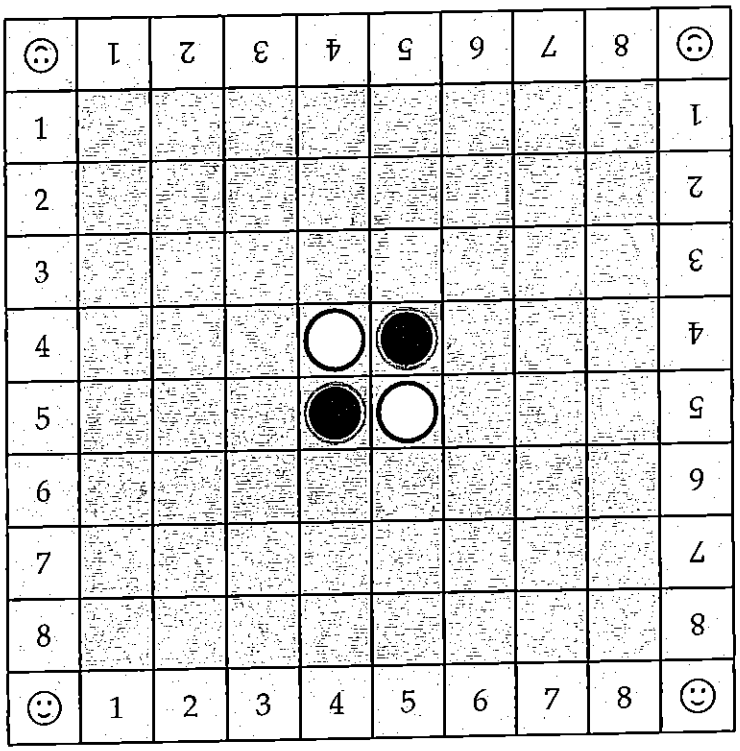
10. Durch Lösen der Aufgaben sind maximal 100 Punkte erreichbar. Sie dürfen damit rechnen einen Übungsschein bzw. ein Zertifikat zu erhalten, wenn Sie insgesamt mindestens 50 Punkte erreichen.

11. Die Klausurergebnisse können Sie ab dem 21. August 2013 in der Virtuellen Universität einsehen. Die Anmeldefrist zur Nachklausur endet am 29. August 2013. Bitte beachten Sie, dass Sie sich ebenfalls **selbstständig für die Nachklausur anmelden** müssen, da bei Nichtbestehen **keine automatische Anmeldung** erfolgt. Eine Abmeldung ist jederzeit möglich.

Aufgabe 1 Reversi-Algebra

30 Punkte

In dieser Aufgabe sollen Sie eine Algebra für das Spiel Reversi entwickeln. Das Spiel wird von zwei Spielern auf einem 8×8-Spielfeld gespielt. Es gibt schwarze und weiße Steine, die jeweils einem der Mitspieler zugeordnet sind. Zu Beginn des Spiels hat das Spielfeld die folgende Konfiguration:



Der Spieler mit den schwarzen Steinen beginnt. Beim Zug eines Spielers wird ein Stein der Farbe des Spielers auf ein freies Feld des Spielfelds gesetzt. Gegnerische Steine, die zwischen dem neuen Stein und einem nächstgelegenen Stein des setzenden Spielers liegen, werden umgefärbt, d.h. erhalten die Farbe des setzenden Spielers. Dabei darf kein leeres Feld dazwischen liegen. Dies gilt für alle 8 möglichen Richtungen (horizontal, vertikal, diagonal). Ein Zug ist gültig, wenn mindestens ein gegnerischer Stein umgefärbt wird. Besteht nicht die Möglichkeit, einen solchen Zug zu machen, kann der Spieler auch passen, d.h. der gegnerische Spieler ist wieder an der Reihe. Ist ein normaler Zug möglich, darf nicht gepasst werden. „Passen" wird in dieser Algebra als ein spezieller Zug angesehen.

Das Spiel ist beendet, wenn beide Spieler keine Züge (außer passen) mehr machen können, z.B.:

- weil einer der Spieler keine Steine mehr auf dem Brett hat *oder*
- das Spielbrett keine freien Felder mehr enthält

Gewonnen hat am Ende des Spiels derjenige Spieler, der die meisten Steine auf dem Spielbrett hat. Solange das Spiel nicht beendet ist, gibt es keinen Gewinner.

Ihre Algebra soll die folgenden Operatoren bereitstellen:

rev bestimmt zu einer Farbe deren Gegenstück, d.h. schwarz wird zu weiß und umgekehrt. Sollten in Ihrer Farbdarstellung weitere Farbwerte enthalten sein, wird die Ausgangsfarbe zurückgeliefert.

create erzeugt ein neues Spiel mit der oben angegebenen Konfiguration

isValid prüft, ob ein Zug gültig ist

isFinished prüft, ob das Spiel beendet ist

winner ermittelt die Farbe des Spielers, der das Spiel gewonnen hat

move führt einen Zug durch, falls dieser gültig ist, ansonsten bleibt das Spielbrett unverändert

(a) Geben Sie die Signatur der Reversi-Algebra an. *5 Punkte*

(b) Geben Sie unter Beachtung der nachstehenden Punkte die Trägermengen Ihrer Sorten an. *10 Punkte*

- das Spielbrett ist als **Menge** von Feldern modelliert
- jedes Feld des Spielbretts „kennt" seine Position und seine Farbe
- die Modellierung des Spielbretts erlaubt nur gültige Konfigurationen, d.h.
  - keine zwei Felder in der Menge haben die gleiche Position
  - es gibt keine belegten Felder, bei denen alle Nachbarfelder (inklusive diagonal liegende Felder) nicht belegt sind
- das Spielbrett „kennt" den Spieler, der aktuell an der Reihe ist.

(c) Geben Sie die Funktionsdefinitionen für die Operatoren der Algebra an. *15 Punkte*

Hinweise:

- Wenn es sinnvoll ist, können Sie zusätzliche Mengen definieren, die nicht in den Sorten auftauchen müssen.
- Trägermengen für *int*, *bool*, *real* und *string* müssen nicht angegeben werden.
- Zerlegen Sie Funktionen, falls es Ihnen sinnvoll erscheint, in Teilfunktionen.
- Die Menge aller Nachbarpositionen einer Position (x,y) in einem Reversi-Feld lässt sich wie folgt darstellen:

$$neighbors((x,y)) = \{(a,b) \mid \begin{array}{l} 1 \leq a,b \leq 8 \wedge \\ (a \neq x \vee b \neq y) \wedge \\ a = x+i, b = y+j,\ i,j \in \{-1,0,1\}\} \end{array}$$

- Die Positionen, die sich auf der Diagonalen rechts unter einer Position (x, y) befinden, kann man wie folgt beschreiben:

Aufgabe 2 Hashing

18 Punkte

A	B	C	D	E	F	G	H	I	J	K	L	M
1	2	3	4	5	6	7	8	9	10	11	12	13
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

Die Städte Wolfsburg, Essen, Neuenahr, Frankfurt, Leverkusen, Potsdam, Sindelfingen, Jena und Duisburg sollen mittels Hashing verwaltet werden.
Fügen Sie die Städte in der angegebenen Reihenfolge in eine Hashtabelle mit 10 Behältern ein. Gehen Sie dabei von geschlossenem Hashing und einer Behältergröße von 1 aus.
Der Hashwert einer Stadt ist dabei gegeben durch die Summe der Buchstabenwerte der ersten drei Buchstaben des Städtenamens.
Die Zuordnung einer Stadt zu einem Behälter erfolgt über die Divisionsmethode.
Sollte es beim Einfügen zu Kollisionen kommen, wenden Sie lineares Sondieren in der verallgemeinerten Form mit c = 3 an.
Hinweis: Sie dürfen die Lösungen zu den Aufgabenteilen (a) und (b) in einer Tabelle angeben.

(a) Berechnen Sie die Hashwerte für die angegebene Städtenamen. *4 Punkte*

(b) Geben Sie tabellarisch für jede Stadt die zugeordneten Behälternummern und die im Verlauf der Berechnung geprüften Behälternummern an. *7 Punkte*

(c) Geben Sie die endgültige Hashtabelle an. *4 Punkte*

(d) Ist die Nutzung des geschlossenen Hashings in dieser Form hier sinnvoll? Begründen Sie Ihre Aussage. *3 Punkte*

Aufgabe 3 Quicksort

24 Punkte

(a) Sortieren Sie die Zahlenfolge *16 Punkte*

56 – 11 – 57 – 2 – 97 – 63 – 95 – 57 – 12 – 96 – 89 – 66

mit dem Verfahren Quicksort. Zeichnen Sie dazu den einen Baum, in dem jeder Knoten einen Aufruf von Quicksort darstellt und dessen Knotenmarkierung jeweils die entsprechende Eingabefolge ist. Markieren Sie in jeder Eingabefolge das Splitelement, welches durch die Funktion *findx* aus dem Kurstext zu bestimmen ist. Der Mergeschritt braucht nicht dargestellt zu werden.

(b) Ist Quicksort stabil? Begründen Sie Ihre Aussage. *3 Punkte*

(c) Bestimmen Sie die Laufzeitkomplexität von Quicksort, wenn die Folge *3 Punkte*

1 – 2 – 3 – 4 – 5

aufsteigend sortiert werden soll, wobei das Pivotelement durch *findx* zu bestimmen ist. Begründen Sie Ihre Aussage.

(d) Nennen Sie zwei Sortierverfahren, bei denen es von Vorteil ist, wenn die zu sortierende Folge bereits sortiert vorliegt. *2 Punkte*

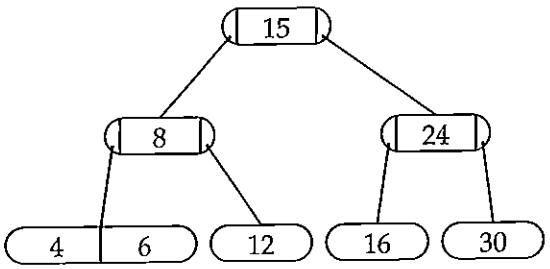
Aufgabe 4 B-Baum

27 Punkte

(a) Fügen Sie die nachstehende Folge von Zahlen in der gegebenen Reihenfolge in einen anfangs leeren B-Baum der Ordnung 1 ein. Geben Sie dazu den Baum vor und nach jedem Split an. Zeichnen Sie auch den endgültigen Baum. *18 Punkte*

13 – 31 – 19 – 23 – 2 – 5 – 11 – 17 – 3 – 29 – 7

(b) Gegeben Sei der folgende B-Baum der Ordnung 1. *9 Punkte*



Löschen Sie aus diesem die Schlüssel 8, 30 und 12 in der angegebenen Reihenfolge. Zeichnen Sie den Baum vor und nach jeder Unterlaufbehandlung und geben Sie die deren Art an. Können beide Nachbarn eines Knotens Elemente für eine Balance-Operation beisteuern, führen Sie die Balance-Operation mit dem linken Nachbarn durch.

Aufgabe 5 Deckblatt

1 Punkt

Lesen Sie sich die „Hinweise zur Bearbeitung" sorgfältig durch. Füllen Sie den dort aufgeführten Anweisungen entsprechend beide Deckblätter vollständig und korrekt aus.