Hinweise zur Bearbeitung der Klausur zum Kurs 1663 „Datenstrukturen“

Bitte lesen Sie sich diese Hinweise vollständig und aufmerksam durch, bevor Sie mit der Bearbeitung der Klausur beginnen. Beachten Sie insbesondere die Punkte 7 und 8!

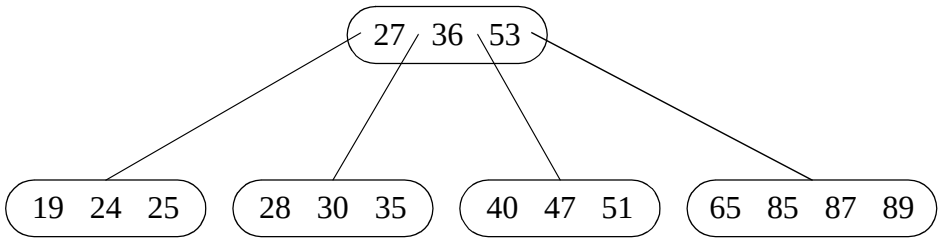
1. Die Klausurdauer beträgt 3 Stunden.
2. Prüfen Sie bitte die Vollständigkeit Ihrer Unterlagen. Die Klausur umfasst:
   - 2 Deckblätter
   - diese Hinweise
   - 1 Formblatt für eine Teilnahmebescheinigung zur Vorlage beim Finanzamt
   - 6 Aufgaben auf den Seiten 2 – 6
3. Bevor Sie mit der Bearbeitung der Klausuraufgaben beginnen, füllen Sie bitte die folgenden Teile der Klausur aus:
   (a) *sämtliche* Deckblätter mit Name, Anschrift sowie Matrikelnummer. Markieren Sie vor der Abgabe auf allen Deckblättern die von Ihnen bearbeiteten Aufgaben.
   (b) die Teilnahmebescheinigung, falls Sie diese wünschen.
4. Schreiben Sie Ihre Lösungen auf ihr eigenes Papier (DIN A4) und nicht auf die Seiten mit den Aufgabenstellungen. Heften Sie vor Abgabe der Klausur die Deckblätter (und evtl. die Teilnahmebescheinigung) an Ihre Bearbeitung.
5. Schreiben Sie bitte auf jedes Blatt oben links Ihren Namen und oben rechts Ihre Matrikelnummer. Nummerieren Sie Ihre Seiten bitte durch.
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 PASCAL- oder MODULA-2-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 aber 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*** PASCAL- oder MODULA-2-Programme sowie Algorithmen. Geben Sie lediglich mathematische Definitionen an wie sie im Kurstext verwandt worden sind!

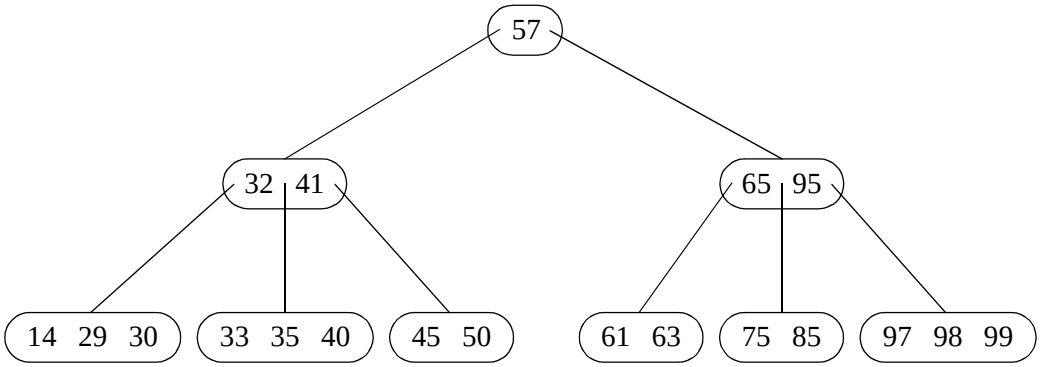
   Ebenso sind Mengendefinitionen in mathematischer Mengennotation durchzuführen. Geben Sie auf keinen Fall Datentypdefinitionen an, und verwenden Sie keine konkreten Datenstrukturen!
9. Als Hilfsmittel sind nur unbeschriebenes Konzeptpapier und Schreibzeug (kein Bleistift!) zugelassen.
10. Es sind maximal 100 Punkte erreichbar. Zur Erlangung eines Übungsscheins bzw. Zertifikats benötigen Sie etwa 50 Punkte.

18 Punkte **Aufgabe 1 (B-Baum)**

9 Punkte (a) Erweitern Sie den nachfolgend dargestellten B-Baum der Ordnung 2, indem Sie nacheinander die Schlüsselelemente 55, 75, 45, 88 und 46 einfügen. Stellen Sie jeweils diejenigen Bäume dar, für die eine Overflow-Operation durchgeführt werden muss. Geben Sie den Baum an, den man als Ergebnis aller Einfügeoperationen erhält.



9 Punkte (b) Löschen Sie aus dem nachstehend dargestellten B-Baum der Ordnung 2 nacheinander die Schlüsselelemente 98, 50, 85, 65, 30, 61, 99, 33, 40 und 57. Geben Sie an, welche Löschungen zu einem Underflow führen und mit welcher Operation zu reagieren ist. Stellen Sie jeweils den dazugehörigen Ergebnisbaum dar.



Hinweis: Sollte in einem Merge- oder Balance-Schritt sowohl der linke als auch der rechte Nachbar des betroffenen Knotens zum Verschmelzen bzw. Ausgleichen zur Verfügung stehen, wählen Sie bitte den linken.

Aufgabe 2 (Heapsort) *20 Punkte*

Gegeben sei ein Maximum-Heap. Beantworten Sie folgende Fragen und begründen Sie Ihre Antworten:

(a) Was versteht man hierbei unter dem Begriff „Heap-Eigenschaft“? *2 Punkte*

(b) Was ist die minimale und maximale Anzahl von Elementen in einem Heap der Höhe h? *2 Punkte*

(c) Wo in einem Ausgangsheap kann sich das kleinste Element einer zu sortierenden Folge nur befinden? *2 Punkte*

(d) Welche Elemente in einer zu sortierenden Folge bilden zu Anfang bereits einen Heap, und warum wird der Ausgangsheap von hinten nach vorne in der Array-Repräsentation eines Heaps bzw. von unten nach oben in der Baumrepräsentation eines Heaps aufgebaut? *4 Punkte*

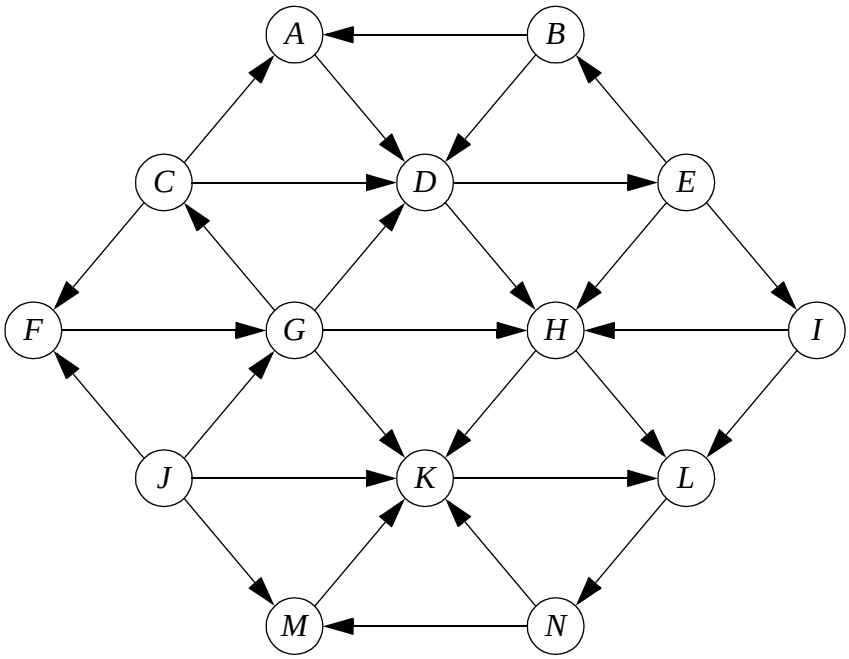
Gegeben sei die Folge 4, 1, 3, 2, 16, 9, 10, 14, 8, 7. Diese Folge ist mittels Heapsort in aufsteigender Reihenfolge zu sortieren.

(e) Geben Sie den Ausgangsheap sowie die Heaps nach den ersten beiden Schritten des Heapsort-Algorithmus an. Markieren Sie die Pfade, entlang denen Zahlen sinken. Kennzeichnen Sie ferner Knoten, die zum sortierten Heap gehören. *10 Punkte*

16 Punkte

Aufgabe 3 (Starke Komponenten)

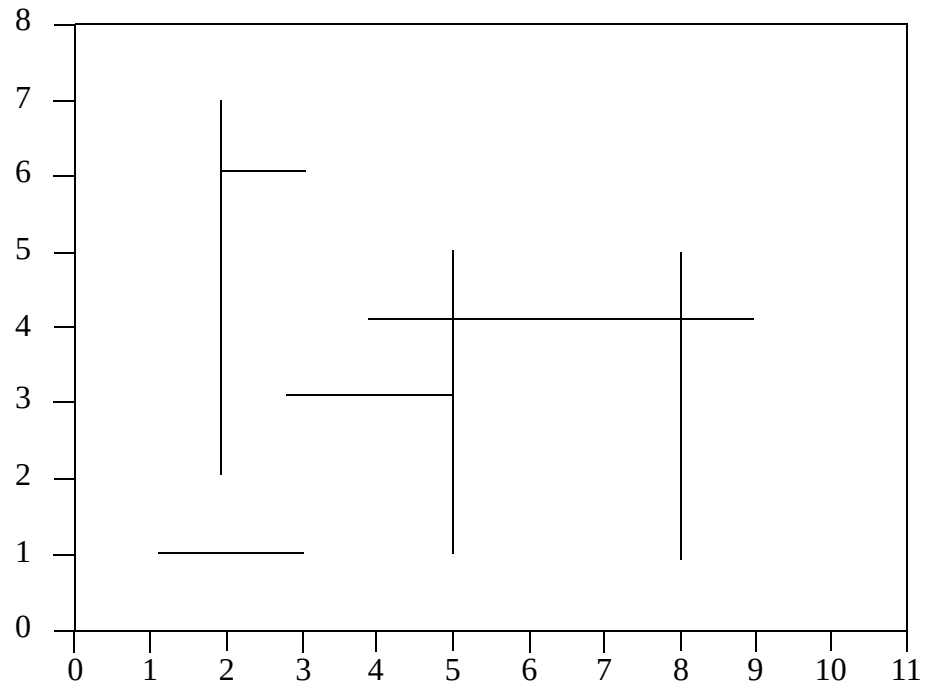
Berechnen Sie zu dem unten gezeigten Graphen *G* die starken Zusammenhangskomponenten mit dem Algorithmus aus dem Kurstext. Geben Sie die Zwischenschritte an. Gehen Sie dabei stets in lexikographischer Reihenfolge vor.



Aufgabe 4 (Segmentschnitt-Problem)

14 Punkte

Das Segmentschnitt-Problem soll für die im folgenden dargestellten Segmente mittels Plane-Sweep gelöst werden.



Führen Sie den Plane-Sweep-Algorithmus aus. Geben Sie dazu zunächst die Sweep-Event-Struktur an und geben Sie dann für jedes Ereignis die durchzuführende Aktion an. Stellen Sie dabei die Sweepline-Status-Struktur in der Form $(S_1, ..., S_n)$ dar. Die S_i bezeichnen hierbei horizontale Segmente der Form (x_1, x_2, y).

16 Punkte

Aufgabe 5 (Rekursives Selection Sort)

Im Kurs haben wir eine *iterative* Version des Sortieralgorithmus *Selection Sort* kennengelernt, die (ein wenig modifiziert) wie folgt aussieht:

```
algorithm IterativeSelectionSort(var S : RecArray, n : integer);
var i, j, temp: integer; min : keytype; minindex : 1..n;
begin
  for i := 1 to n-1 do
     min := S[i].key;
     minindex := i;
     for j := i + 1 to n do
        if S[j].key < min then
           min := S[j].key;
           minindex := j
        fi
     od;
     temp := S[i];
     S[i] := S[minindex];
     S[minindex] := temp;
  od
end IterativeSelectionSort.
```

Der Typ *RecArray* entspricht hierbei einem Typ **array**[1..*n*] **of** *recordtype*, wobei *recordtype* eine Komponente *key* besitzt, die den Sortierschlüssel enthält.

Geben Sie zu *IterativeSelectionSort* einen rekursiven Algorithmus *RecursiveSelectionSort* an und erläutern Sie Ihre Vorgehensweise sowie Ihren Algorithmus. Wie lautet der Aufruf von *RecursiveSelectionSort*, um die gesamte Folge sortieren zu lassen?

16 Punkte

Aufgabe 6 (Streng binäre Bäume)

Ein *streng binärer Baum* ist ein Baum, in dem jeder innere Knoten genau zwei Söhne hat.

12 Punkte (a) Es sei für einen streng binären Baum ein $c > 1$ bekannt, das das Verhältnis der Längen des längsten und des kürzesten Pfades von der Wurzel zu einem Blatt angibt. Bestimmen Sie für einen solchen Baum die Ober- und Untergrenze der Anzahl N der Knoten im Baum in Abhängigkeit von c und der Höhe h.

4 Punkte (b) Nehmen wir nun an, dass ein innerer Knoten nicht 2, sondern k Söhne hat. Wie lauten in diesem Fall die Ober- und die Untergrenzen?