Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Institut für Simulation und Graphik
Prof. Dr. Klaus-Dietz Tönnies, Prof. Dr. Thomas Strothotte

Magdeburg, 04.08.05

Prüfungsklausur

Einführung in die Informatik/Algorithmen und Datenstrukturen

Name:	Matrikelnummer:
Vorname:	Studiengang:
Blattanzahl:	
Unterschrift Student/in:	Unterschrift Aufsicht:

Aufg. 1	Aufg. 2	Aufg. 3	Aufg. 4	Aufg. 5	Aufg. 6	Aufg. 7	Summe
(von 6)	(von 14)	(von 13)	(von 6)	(von 16)	(von 8)	(von 7)	(von 70)

Allgemeine Hinweise

- Schreiben Sie auf jedes Blatt Ihren Namen, Ihre Matrikelnummer und die Seitennummer.
- Die Programme sind gut zu kommentieren! Programmstrukturen sind in **Java-ähnlichem Pseudocode** zu notieren.
- Beginn und Ende einer Aufgabenlösung sind durch einen waagerechten Strich deutlich zu kennzeichnen. Falsche Lösungen sind deutlich als solche zu kennzeichnen.
- Zugelassene Hilfsmittel: ausschließlich Schreibmaterialien
- Die Klausur besteht aus 7 Aufgaben, die Bearbeitungszeit beträgt 210 Minuten.
- Zum Bestehen der Klausur sind 35 Punkte erforderlich.

Aufgabe 1 (2 + 4 = 6 Punkte)

In einem *(x, y)*-Koordinatensystem soll ein Kreis dargestellt werden.

a) Welche Bedingung muss erfüllt sein, damit sich zwei Kreise schneiden bzw. berühren?

b) Schreiben Sie eine Java-Klasse *Circle* für Kreise mit geeigneten Attributen, sie soll einen Konstruktor enthalten und eine Methode

boolean intersect(Circle c),

die *true* zurückgibt, wenn sich beide Kreise schneiden bzw. berühren. Ansonsten soll *false* zurückgeliefert werden.

Aufgabe 2 (8 + 1 + 1 + 2 + 1 + 1 = 14 Punkte)

a) Schreiben Sie eine Java-Methode *quickSort (Comparable[] a).*
b) Geben Sie die Komplexität in O-Notation für den durchschnittlichen Fall an. Begründen Sie Ihre Antwort.
c) Wie sieht der *schlechteste Fall* für Quicksort aus und welche Komplexität in *O*-Notation hat er?
d) Warum kann es kein allgemeines Sortierverfahren geben, das im *schlechtesten Fall* schneller ist als $O(n \log n)$? Machen Sie Ihre Antwort plausibel.
e) Mit welchem Sortierverfahren sollte man arbeiten, wenn man weiß, dass eine Zahlenfolge bereits zum großen Teil sortiert vorliegt. Begründen Sie kurz.
f) Wenn Methode *A* den Aufwand $O(n^2+k)$ hat und Methode *B* den Aufwand $O(n + n \cdot \log k)$, wie groß ist dann der Aufwand von einer Methode *C* die nacheinander *A* und *B* aufruft?
Hinweis: k sei keine Konstante

Aufgabe 3 (2 + 2 + 5 + 1 + 3 = 13 Punkte)

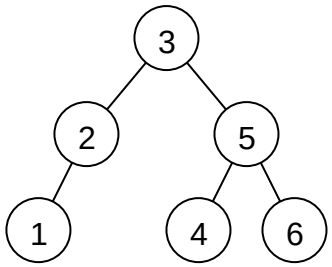
a) Was ist ein *Min-Heap*?
b) Skizzieren Sie einen *Min-Heap* aus den Zahlen *13, 55, 23, 47, 29, 17, 53, 15* und *12*.
c) Schreiben Sie eine JAVA-Methode *trickleDown(int pos)*, die ein Element ab Position *pos* im *Heap* versickern lässt.
d) Mit welcher Komplexität in O-Notation arbeitet die Methode *trickleDown*? Begründen Sie kurz.
e) Erläutern Sie den Algorithmus für eine Methode *bubbleUp*, die mit $log_2(log_2 n)$ Schlüsselvergleichen auskommt.
Hinweis:
Die aus der Vorlesung bekannte Methode *bubbleUp* benötigt $log_2 n$ Schlüsselvergleiche.

Aufgabe 4 (4 + 2 = 6 Punkte)

a) Notieren Sie für den folgenden Text den Huffman-Baum und geben Sie die Huffman-Kodierung an.
Text: *BACDEBBAACABBADBC*
b) Begründen Sie, weshalb die Huffman-Kodierung effizienter ist als die folgende Kodierung:
$A \rightarrow 0\ 0\ 1$
$B \rightarrow 0\ 1\ 0$
$C \rightarrow 0\ 1\ 1$
$D \rightarrow 1\ 0\ 0$
$E \rightarrow 1\ 0\ 1$

Aufgabe 5 (1 + 2 + 2 + 5 + 6 = 16 Punkte)

a) Gegeben sei folgender Baum:



Welche Traversierungsart gibt den Baum in der Reihenfolge 3, 2, 5, 1, 4, 6 aus? Wie hoch ist für diese Traversierung der Aufwand in O-Notation?

b) In welcher Reihenfolge müssen Sie die Zahlen von 1 bis 8 in einen *SimpleSearchTree* (einfacher Binärbaum) einfügen, damit er ausgeglichen ist?

c) Gegeben sei ein *SimpleSearchTree*. Der folgende Code sollte die Summe aus allen *wert*-Attributen aller Knoten zurückgeben. Der Code enthält keine Syntaxfehler, leider arbeitet der Code trotzdem nicht wie gewünscht. Korrigieren Sie alle Fehler, die Sie finden!

```
public static int nodeSum(Node k) {
   if(k.getLeft()==null && k.getRight()==null)return k.wert;
   int i = nodeSum(k.getLeft());
   int j = nodeSum(k.getRight());
   return i + j;
}
```

d) Schreiben Sie eine Java-Methode, die genau die drei Knoten eines *SimpleSearchTree* ausgibt, deren Werte die geringste Differenz zu einem vorgegebenen Suchwert aufweisen. Die Knoten speichern *int*-Werte. Gesucht ist eine Methode, die besser ist als $O(n^2)$.

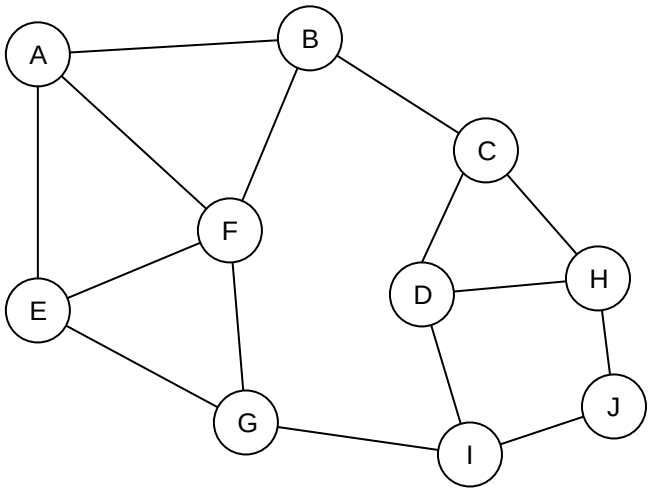
Hinweis: Überlegen Sie, welche der Hilfsstrukturen *Stack*, *Liste*, oder *PriorityQueue* Sie dabei am besten einsetzen können. Sie brauchen die Implementierung dieser Hilfsstrukturen nicht mit anzugeben.

e)

- Wie viele Blätter hat ein AVL-Baum der Höhe h maximal?
- Wenn Sie Werte in einen AVL-Baum sortiert einfügen, werden die Kosten dann besser oder schlechter als bei einem Einfügen in zufälliger Reihenfolge? Begründen Sie kurz ihre Antwort.
- Wie lang ist der Suchweg von der Wurzel zu einem beliebigen Knoten in einem *SimpleSearchTree* mit n Knoten im *besten*, im *mittleren* und im *schlechtesten* Fall?
- Begründen Sie kurz, warum die Inorder-Traversierung für beliebige *n*-äre Bäume nicht existiert.
- Zeichnen Sie einen AVL- ***oder*** einen Rot-Schwarz-Baum, in den Sie der Reihe nach die Werte 6, 3, 0, 2, 4, 5 einfügen. Geben Sie dabei alle notwendigen Rotationen an.

Aufgabe 6 (4 + 4 = 8 Punkte)

a) Gegeben ist folgender Graph:



Welche Arten der Traversierung von Graphen kennen Sie? Wenden Sie diese auf den gegebenen Graphen an. Der Startpunkt sei A. Beschreiben Sie kurz für die Traversierungen Ihre Vorgehensweise.

b) Nennen Sie zwei Möglichkeiten, um einen Graphen als Datenstruktur zu beschreiben und geben Sie den gegebenen Graphen in diesen beiden Darstellungen an.

Aufgabe 7 (1 + 3 + 3 = 7 Punkte)

a) Gegeben ist die folgende Zeichenkette (String):
FISCHERS FRITZE FISCHT
Gesucht ist das Muster: FISCH.
Wie viele Vergleiche werden benötigt, um alle Vorkommen von „Fisch“ mit dem *„brute-force-Algorithmus“* zu ermitteln?

b) Schreiben Sie eine Java-Methode
bruteForce(String text, String pattern),
in der alle Vorkommen des Musters *pattern* in der Zeichenkette *text* ermittelt und zurückgegeben werden.

c) Welcher Algorithmus kann für das Beispiel aus a) zur Verbesserung eingesetzt werden? Erklären und zeigen Sie die Arbeitsweise des von Ihnen gewählten Algorithmus.

Wir wünschen Ihnen viel Erfolg!