Institut für Computergraphik und Algorithmen

Abteilung für Algorithmen und Datenstrukturen

TECHNISCHE UNIVERSITÄT WIEN

Prof. Petra Mutzel
Gunnar Klau
Ivana Ljubić
Günther Raidl
René Weiskircher

Sommersemester 2000/2001

Klausur zur Vorlesung
Algorithmen und Datenstrukturen 1
09. März 2001

a.) Machen Sie bitte die folgenden Angaben in deutlicher Blockschrift:

Name: _____ Vorname: _____

Matrikelnummer: _____ Studienkennzahl: _____

b.) Legen Sie während der Klausur Ihren Studentenausweis vor sich auf das Pult.

c.) Schreiben Sie die Lösungen direkt auf das jeweilige Aufgabenblatt. Wenn Ihnen das Papier ausgeht, bitten Sie die Aufsicht um Nachschub. Es ist nicht erlaubt eigenes Papier zu verwenden!

d.) Denken Sie daran, dass keinerlei Hilfsmittel erlaubt sind – weder Taschenrechner, irgendwelche Unterlagen, Handys,...

VOR DER ABGABE AUSZUFÜLLEN:

e.) Geben Sie bitte die Anzahl der zusätzlich abgegebenen Blätter an: _____

f.) Kreuzen Sie bitte die von Ihnen bearbeiteten Aufgaben in der ersten Zeile der Tabelle an:

Aufgabe	A 1	A 2	A 3	A 4	$\sum$	Note
bearbeitet					—	_____
maximale Punktzahl	10	15	15	15	55	_____
erreichte Punktzahl						

Viel Erfolg!

Aufgabe 1: O-Notation **(10 Punkte)**

a.) Seien $f(n)$ und $g(n)$ Funktionen mit positiven Wertebereichen. Sind die folgenden Aussagen wahr oder falsch?

Diese Aussage ist	Wahr	Falsch
$f(n) = \Theta(g(n)) \leftrightarrow g(n) = \Theta(f(n))$.		
Aus $f(n) \in \Theta(n^2)$ folgt $f(n) \in O(n^3)$.		
Aus $f(n) \in \Omega(n^2)$ folgt $f(n) \in O(n)$.		
Aus $f(n) \in \Omega(n^2)$ folgt $n^2 \in \Theta(f(n))$.		
$f(n) + g(n) = O(max(f(n), g(n))$.		
$f(n) = O(\sqrt{f(n^2)})$.		

b.) Sei

$$f(n) = \begin{cases} \frac{\log n}{125} + 5, & n > 125 \\ \frac{n^3}{125} + 25n^2 + 5n, & n \leq 125 \end{cases}$$

Ist $f(n) = O(\log n)$? Beweisen Sie Ihre Antwort.

Aufgabe 2: Sortierverfahren **(15 Punkte)**

a.) Führen Sie das Sortierverfahren Quicksort mit der folgenden Folge durch:

$$25, 16, 3, 10, 19, 38, 17, 20, 7, 5.$$

Stellen Sie die Teilfolge nach jeder Iteration dar. Markieren Sie jeweils das Pivotelement.

b.) Wie ist die Worst-Case Laufzeit von Quicksort für eine n-elementige Teilfolge?

c.) Wie ist die Laufzeit von Quicksort für den Fall, dass alle n Schlüssel der Folge gleich sind?

Aufgabe 3: Was macht der folgende Algorithmus? **(15 Punkte)**

Gegeben sei ein Array $A[1, ..., n]$ mit n ganzen Zahlen.

(0) **Algorithmus WhatAmI (int** $A[1..n]$**, int i, int j)**
(1) /* **Input:** Folge von n Zahlen $A[1, ..., n]$ */
(2) /* **Output:** ??? */
(3) Falls $(i > j)$
(4) **return** true;
(5) Falls $(A[i] == A[j])$
(6) **return** WhatAmI $(A, i + 1, j - 1)$;
(7) **return** false;

a.) Was können Sie nach Ablauf des Algorithmus mit Aufruf WhatAmI (A,1,n) über den Rückgabewert sagen?

b.) Erklären Sie in eigenen Worten wie der Algorithmus genau funktioniert.

c.) Wie ist die Worst-Case Laufzeit des Algorithmus WhatAmI(A,1,n)?

Aufgabe 4: Binäre Bäume - JAVA Implementierung **(15 Punkte)**

Geben Sie eine effiziente Java–Implementierung für folgendes Problem an:

a.) Gegeben sei ein binärer Baum B mit ganzzahligen Schlüsseln. Gesucht ist die Summe der Schlüssel von allen Blättern.

Geben Sie eine Java–Implementierung für einen Algorithmus `Summe` an, der diese Aufgabe löst. Verwenden Sie hierzu die folgende Klasse `binaryTree` und ergänzen Sie die Methode `TreeSum`. Achten Sie darauf, aussagekräftige Bezeichner zu verwenden und versehen Sie Ihren Code mit Kommentaren.

```
class Node {
  private int key;
  private Node left, right;
  public Node(int val, Node l, Node r) {
    key = val;
    left = l;
    right = r;
  }
  void setKey(int val) {key = val;}
  void setLeft(Node newLeft) {left = newLeft;}
  void setRight(Node newRight) {right = newRight;}
  int getKey() { return key;}
  Node getLeft() {return left;}
  Node getRight() {return right;}
  . . .
}

public class binaryTree {
  private Node root;
  public binaryTree() {
    root = null;
  }
  public binaryTree(Node n) {
       root = n;
  }
  . . .
  public int TreeSum(){
   . . .
  }
}
```

Hinweis: Eine Lösung könnte darin bestehen, eine zusätzliche rekursive Methode `Sum(Node n)` zu implementieren, die die Summe der Werte in den Blättern des Teilbaums mit Wurzel n berechnet.

b.) Was können Sie über die Laufzeit Ihres Algorithmus sagen?