Kanaeva / Schweppe, FUB Informatik

Übung zu Algorithmen und Programmieren III, WS 2001/2

Übung 10

Ausgabe: 18.12.01
Abgabe: **14.01.02** spätestens 12.00 Uhr

Aufgabe 1 (2,5+1+2,5+1+1+1=9 P)

a) Zeichnen Sie binäre Suchbäume der Höhen 2, 3, 4, 5 und 6 für die Schlüsselmenge {1,2,5,10,16,17,21}

b) Welche minimale bzw. maximale Anzahl von Knoten enthält ein binärer Suchbaum der Höhe h?

c) Angenommen, wir haben in einem binären Suchbaum Zahlen zwischen 1 und 1000 gespeichert und wollen nach der Zahl 363 suchen. Welche der folgenden Zahlenfolgen kann <u>nicht</u> die Folge mit den Zahlen, die bei der Suche besucht wurden, sein?

1) 2, 252, 401, 398, 330, 344, 397, 363
2) 924, 220, 911, 244, 898, 258, 362, 363
3) 925, 202, 911, 240, 912, 245, 363
4) 2, 399, 387, 219, 266, 382, 381, 278, 363
5) 935,278,347,621,299,392,358,363

d) Sei T ein binärer Suchbaum, `x` ein Blattknoten, `y` sein Vaterknoten. Der Schlüssel eines Knotens `k` wird mit `val(k)` bezeichnet. Zeigen Sie: `val(y)` ist entweder der kleinste Schlüssel im Baum größer als `val(x)` oder der größte Schlüssel in dem Baum, der kleiner als `val(x)` ist.

e) S. Tupid glaubt eine wichtige Eigenschaft binärer Suchbäume entdeckt zu haben. Angenommen, die Suche nach dem Schlüssel k endet erfolgreich in einem Blatt. Folgende 3 Knotenmengen lassen sich dann definieren: A als Menge der Knoten, die links vom Suchpfad liegen, B als Menge der Knoten auf dem Suchpfad zu k und C als Menge der Knoten rechts vom Suchpfad. Die Behauptung von Tupid lautet: für alle $a \in A$, $b \in B$, $c \in C$ gilt: $a \leq b \leq c$. Finden Sie ein möglichst kleines Gegenbeispiel für diese Behauptung.

f) `G = (E,K)` sei ein ungerichteter Graph. Zeigen Sie:
`G` ist ein freier Baum $\Leftrightarrow$ `G` azyklisch und `|K| = |E| - 1`

Aufgabe 2 (6+4=10 P)

a) Implementieren Sie binäre Suchbäume mit Basistyp `Object`. Abstraktionsfunktion und Repräsentations-Invariante nicht vergessen! Die Klasse soll mindestens die Methoden `boolean add(Object o), boolean contains(Object o), boolean isEmpty()` und `int height()` besitzen.

b) Ein zufällig erzeugter binärer Suchbaum mit `n` Knoten hat eine Tiefe von ca. `C*log n`. Sie sollen dies experimentell bestätigen. Erzeugen Sie binäre Suchbäume mit `n` Elementen und finden Sie deren Tiefe (`n=100,200,...,1000`). Um die Suchbäume aufzubauen , erzeugen Sie Zufallszahlen zwischen `0` und `MAXINT` und bestimmen Sie die Konstante `C` näherungsweise.

Bitte wenden

Aufgabe 3 (7+3+2=12 P)

a) Spezifizieren und implementieren Sie eine Klasse `Huffman` zur Berechnung von Huffman-Codes (imperativ in Java). Ein Konstruktor soll einen Huffman-Codebaum für das natürliche Alphabet, dessen Zeichenhäufigkeit (bzw. Auftretenswahrscheinlichkeit) bekannt ist, bestimmen, z.B. `Huffman(AlphFreq af)`. Die Klasse soll Methoden zur Kodierung und Dekodierung von Zeichen eines Alphabets enthalten.

b) Gegeben ist der Text, dazu siehe `http://www.inf.fu-berlin.de/lehre/WS01/ALP3/uebungen/u10/weihnachtsgesch.html.`
Schreiben Sie eine Java-Klasse, in der Sie den gegebenen Text mit einem Huffman-Code kodieren. Dazu muss zunächst ein Objekt `h` einer Klasse `TextHistogramm` erzeugt werden, das die Häufigkeit der in einem Text vorkommenden Zeichen bzw. die entsprechenden Wahrscheinlichkeiten enthält. Betrachten Sie dabei nur die Zeichen, die in der Tabelle in dem Punkt c) vorkommen. Mit `Huffman hCode = Huffman(h)` wird der Code erzeugt. Kodieren (und komprimieren) Sie den Text unter Verwendung von Huffman und bestimmen Sie die Länge des kodierten Textes. Zum Testen ihrer Implementierung dekodieren Sie den Text.

c) Bestimmen Sie zu den statischen Wahrscheinlichkeiten:

Buchstaben	Häufigkeit	Buchstaben	Häufigkeit
a	6,51%	n	9,78%
b	1,89%	o	2,51%
c	3,06%	p	0,79%
d	5,08%	q	0,02%
e	17,40%	r	7,00%
f	1,66%	s	7,27%
g	3,01%	t	6,15%
h	4,76%	u	4,35%
i	7,55%	v	0,67%
j	0,27%	w	1,89%
k	1,21%	x	0,03%
l	3,44%	y	0,04%
m	2,53%	z	1,13%

den Huffman-Code für das Alphabet. Kodieren Sie damit den im b) angegebenen Text und vergleichen Sie die Länge mit dem in b) erzielten Ergebnis.

Aufgabe 4 (2+2=4 P)

a) Sei $f(n) = \Sigma_{i=1..n}\, i$, $g(n) = n^2$
Zeigen Sie, dass $g(n) = O(f(n))$ und $f(n) = O(n^2)$.

b) Zeigen Sie $ld(2n^2) = O(ld\ n)$, $ld\ n$ = Logarithmus zur Basis 2 von n.

Wir wünschen Ihnen frohe Weihnachten und alles Gute für 2002!