Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Christian Scheideler
Stefan Eckhardt

Wintersemester 2005/06
Übungsblatt 2
29. Oktober 2005

Effiziente Algorithmen und Datenstrukturen I

Abgabetermin: 4.11.2004 **vor** *der Zentralübung*

Aufgabe 1 (10 Punkte)

Bestimmen Sie eine obere Schranke für die amortisierte Laufzeit der Einzeloperationen in einer Folge von n Operationen $P = p_1 p_2 \dots p_n$, wobei für die Laufzeit t_i der i-ten Operation p_i gilt:

$$t_i = \begin{cases} \Theta(i) & \text{, falls } i = f(k) \text{ für ein } k \in \mathbb{N}, \\ O(1) & \text{, sonst.} \end{cases}$$

Dabei sei $f(x)$ eine streng monoton steigende Funktion. Was ergibt sich für $f(x) = x^2$ und für $f(k) = p_k$, wobei p_k die k-te Primzahl ist? (Hinweis: Verwenden Sie den Primzahlsatz.)

Aufgabe 2 (10 Punkte)

Gegeben sei untenstehendes Programm, das prüft, ob ein Array sortiert ist. (O.B.d.A. enthält $a[]$ keine gleichen Elemente.) Wie sind die Best-Case, die Worst-Case und Average-Case Laufzeiten (im uniformen Kostenmodell)? Beweisen Sie Ihre Antworten.

Input: Array $a[1, \dots, n]$
$i := 2$
while $i \leq n$ **and** $a[i-1] \leq a[i]$ **do**
  i:=i+1;
if $i > n$ then
  return „Array sortiert!“
  else
  return „Array nicht sortiert!“

Aufgabe 3 (10 Punkte)

Es seien T_1 und T_2 zwei (a, b)-Bäume mit n_1 bzw. n_2 Knoten, so dass für alle $x \in T_1$ und $y \in T_2$ gilt: key(x) < key(y). Entwerfen Sie eine Prozedur Concatenate, die T_1 und T_2 zu einem neuen (a, b)-Baum verschmilzt und deren Laufzeit $O(\log(n_1 + n_2))$ ist.

Aufgabe 4 (10 Punkte)

Die Schlüssel $(5, 16, 22, 45, 2, 10, 18, 30, 50, 12, 1)$ werden in der vorliegenden Reihenfolge

1. in einen anfangs leeren $(2, 4)$-Baum T' eingefügt.
2. in einen anfangs leeren rot-schwarz-Baum T'' eingefügt.

Zeichen Sie die Bäume T' und T'' jeweils nachdem alle Schlüssel eingefügt wurden.