

Übungen zur Vorlesung
Datenstrukturen, Algorithmen und Programmierung 2 (DAP2)
Sommersemester 2007
Blatt 1

Aufgabe 1.1 (5 Punkte)

Implementiere die in der Vorlesung vorgestellten Algorithmen für das Maxsummenproblem in Java. Vergleiche ihre Rechenzeiten für zufällige Eingaben mit $n = 1.000$, 10.000 , 100.000 und $1.000.000$ (soweit das zeitlich möglich ist). Belege das Array mit Hilfe des Java-Zufallszahlengenerators wie folgt:

```
a = new int[n];  
Random Zufallszahl = new Random(8);  
for (int i=0; i<n; i++)  
{  
    a[i] = (Zufallszahl.nextInt() % 4793) - 2396;  
}
```

(Zur Kontrolle: Wenn die Arrays mit dem obigen Verfahren initialisiert werden, sind die Werte der maximalen Lösungen für die obigen Eingabelängen 5.687, 9.701, 12.983 und 13.298.)

Aufgabe 1.2 (5 Punkte)

Die in der Vorlesung vorgestellten Algorithmen für das Maxsummenproblem berechnen jeweils den maximalen Wert $f(i, j)$, aber nicht ein Paar (i, j) , für das der maximale Wert angenommen wird.

Wie können der Divide-And-Conquer-Algorithmus und der Algorithmus der dynamischen Programmierung erweitert werden, sodass sie auch jeweils ein solches Paar (i, j) ausgeben?

Aufgabe 1.3 (5 Punkte)

Löse die drei folgenden Rekursionsgleichungen (wobei n im ersten und zweiten Fall bzw. im dritten Fall von der Form 4^k bzw. 2^k ist) und gib jeweils einen Induktionsbeweis über k an:

a) $T(1) = 1, T(n) = 16 \cdot T(n/4)$

b) $T(1) = 1, T(n) = 8 \cdot T(n/4)$

c) $T(1) = 1, T(n) = 4 \cdot T(n/2)$

Aufgabe 1.4 (5 Punkte)

Schreibe ein möglichst effizientes Registermaschinenprogramm für die Exponentiation von zwei ganzen Zahlen, d. h. die Berechnung von x^y für $x, y \in \mathbb{N}_0^+$. Wie viele Multiplikationen MUL braucht dein Algorithmus im Worst- und im Best-Case?