Prof. Dr. Ir. Joost-Pieter Katoen

Christian Dehnert, Jonathan Heinen, Thomas Ströder, Sabrina von Styp

Hinweise:

- Die Übungsblätter sollen in Gruppen von je 3 Studierenden aus der gleichen Kleingruppenübung bearbeitet werden.
- Die Lösungen müssen bis Montag, den 30. April um 11:00 Uhr in den entsprechenden Übungskasten eingeworfen werden. Sie finden die Kästen am Eingang Halifaxstr. des Informatikzentrums (Ahornstr. 55).
- Namen und Matrikelnummern der Studenten sowie die **Nummer der Übungsgruppe** sind auf jedes Blatt der Abgabe zu schreiben. Heften bzw. tackern Sie die Blätter!
- Bitte beachten Sie, dass am 1. Mai keine Übungen stattfinden. Die Ausweichtermine finden Sie auf unserer Webseite.

Aufgabe 1 (Lineare Suche): (5 Punkte)

Geben Sie einen Algorithmus an, der in $\mathcal{O}(n)$ für eine Folge von n ganzen Zahlen (gegeben als Array) eine maximale Teilfolge findet. Eine Teilfolgen wird hierbei von beliebig vielen (maximal n) *aufeinanderfolgenden* Array Einträge gebildet. Sie ist maximal, wenn die Summe ihrer Elemente maximal ist, d.h. wir suchen aus allen möglichen Teilfolgen eine mit maximaler Summe.
Die Teilfolge soll dabei als *Startindex*, *Endindex* sowie *Summe* der Folge ausgegeben werden. Die Eingangsfolge

$$12,\ -34,\ 56,\ -5,\ -6,\ 78,\ -32,\ 8$$

liefert beispielsweise die Indizes 3 und 6 sowie die Summe $56 - 5 - 6 + 78 = 123$.

Aufgabe 2 (Rekursionsgleichungen): (3+4 Punkte)

a) Zeigen Sie mit Hilfe der Substitutionsmethode, dass für $T(n) = 2 \cdot T(\frac{n}{3}) + T(\frac{n}{4}) + 4n + 3$ mit $T(0) = -\frac{3}{2}$ gilt, dass $T(n) = 48n - \frac{3}{2}$

b) Nutzen Sie die Methode der Variablentransformation um die exakte Lösung der Rekursionsgleichung $T(n) = T(\sqrt[3]{n}) + 3$ mit $T(2) = 2$ zu bestimmen.

Aufgabe 3 (Rekursionsbäume): (5+4 Punkte)

Erstellen Sie zu den folgenden Rekursionsgleichungen die entsprechenden Rekursionsbäume und lesen Sie eine möglichst kleine obere Schranke der Lösung ab. Überprüfen Sie die Schranke mit Hilfe des Substitutionsverfahrens.

a)

$$T(n) = 4 \cdot T(\frac{n}{3}) + \frac{n}{2} + 3 \text{ mit } T(0) = 1$$

b)

$$T(n) = T(n-2) + T(n-1) + 7 \text{ mit } T(0) = 1 \text{ und } T(1) = 7$$

Aufgabe 4 (Analyse rekursiven Codes): (2+2+2+2 Punkte)

Geben Sie für die folgenden Programme die Laufzeitkomplexität als Rekursionsgleichung in Abhängigkeit des Eingabeparameters n an:

a)

```
int rekursion1(int n){
    if(n == 0)
        return 1;

    int sum = 1;
    while(sum < n){
        sum = sum * 2;
    }

    return n * rekursion1(n/2) + sum
}
```

b)

```
int rekursion2(int n){
    if(n <= 0)
        return n*n;

    int wert = rekursion2(n-1) * (rekursion2(n-1) + 2);
    n--;

    for(int i = 0; i < n; i++){
        for(int j = n; j > 0; j--){
            sum += i * j;
        }
    }
    return wert * rekursion2(n);
}
```

c)

```
int rekursion3(int n){
    if(n/2 <= 0){
        int value = 3;
        for(int i = 0; i < n; i++)
            value *= value;
        return value;
    }else
        return rekursion3(n-1) * rekursion3(n-2) * rekursion3(n-4);
}
```

d)

```
int rekursion4(int n){
    if(n <= 0)
        return foo(5);
    return rekursion4(n/3) * foo(n*n);
}

int foo(n){
    int k = 1;
    for(int i = 0; i < n; i++)
        k *= 2;
    for(int i = 0; i < k; i++)
        n *= n;
    return n;
}
```