

# Übungsblatt 14

## RECHENSTRUKTUREN und FUNKTIONALE PROGRAMMIERKONZEPTE UND REKURSION

**Abgabe: bis Freitag 11.02.2005, 13:00 Uhr in die Einwurfkästen im Untergeschoß des neuen Informatikgebäudes am Fasanengarten**

**Erreichbare Punkte: 7 P / 53 T  
(Praktische Aufgaben / Theoretische Aufgaben)**

### Hinweis:

Alle Programmieraufgaben sind unter Verwendung der in der Vorlesung vorgestellten In-/Out-Klassen zu erstellen und (ähnlich wie die Beispielprogramme im Skript) in ausführlicher Form zu kommentieren.

Versehen Sie die Programme mit Eingabeüberprüfungen, die sicherstellen, dass der Anwender korrekte Daten eingegeben hat.

## 1 GRAMMATIKEN

### 1.1 Sprache und kontextfreie Grammatik (8T)

Geben Sie für die Sprache  $L = \{a^n b^{2n} \mid n > 0\}$  eine anständige (*proper*), kontextfreie Grammatik  $G$  an und beweisen Sie, dass  $L = L(G)$  gilt.

## 2 MATHEMATISCHE LOGIK

### 2.1 Aussagenlogik (6T)

Vereinfachen (verkürzen) Sie den folgenden booleschen Ausdruck durch Anwendung von Regeln der Booleschen Algebra so weit es geht.

$$\neg(a \leftrightarrow \neg(b \leftrightarrow (a \wedge b)))$$

### 2.2 Prädikatenlogik (12T)

Gegeben sind folgende Prädikate:

$E(a,b)$ : a ist Elternteil von b

$m(x)$ : x ist männlich

$w(x)$ : x ist weiblich

Formulieren Sie mit Hilfe der gegebenen Prädikate die folgenden Aussagen.

- a) x ist der Großvater von y (2T)
- b) x ist eine Schwester von y (2T)
- c) Jeder hat eine Mutter und einen Vater. (2T)
- d) x ist Großvater väterlicherseits und y ist Großmutter mütterlicherseits von z (3T)
- e) x ist eine Cousine von y (3T)

## 3 FUNKTIONALE PROGRAMMIERKONZEPTE

### 3.1 Palindrome in Java (7P)

In der Übung wurde die Eigenschaft eines Palindroms erklärt. Entwickeln Sie ein Java-Programm, das für eine Zeichenkette (String) die Palindromeigenschaft überprüft. Schreiben Sie hierzu zwei Versionen – eine iterative und eine rekursive.

### 3.2 Vollständige Induktion und rekursive Funktionen (22T)

Welche der folgenden Rekursionen realisieren eine Funktion, bei der alle Zahlen zwischen  $n$  und 100 aufmultipliziert werden. Gehen Sie dabei davon aus, dass  $0 < n \leq 100$  gilt.

Beispiel: Für  $n = 50$  soll berechnet werden:  $50 * 51 * 52 * \dots * 99 * 100$ .

fact ist dabei die aus der Vorlesung bekannte Funktion für die normale Fakultät.

Begründen Sie Ihre Antwort. Verwenden Sie im positiven Fall einen Beweis durch vollständige Induktion.

a) `static int fact_A(int n) { // assertion: 0 < n <= 100 (3T)`  
`if (n < 100) return n * fact_A(n + 1);`  
`return 100;`  
`}`

b) `static int fact_B(int n) { // assertion: 0 < n <= 100 (3T)`  
`if (n == 100) return n;`  
`return n * fact_B(n + 1);`  
`}`

c) `static int fact_C(int n) { // assertion: 0 < n <= 100 (3T)`  
`return fact(100) / fact(n - 1);`  
`}`

d) `static int fact_D(int n) { // assertion: 0 < n <= 100 (3T)`  
`return n * fact_D(n + 1);`  
`if (n == 100) return n;`  
`}`

e) `static int fact_E(int n) { // assertion: 0 < n <= 100 (3T)`  
`int t = 0;`  
`int erg = 0;`  
`if ((100 - t) == n) return n;`  
`erg = fact_E(100 - t);`  
`t = t + 1;`  
`return erg;`  
`}`

f) `static int fact_F(int n) { // assertion: 0 < n <= 100 (3T)`  
`return n + fact_D(n + 1);`

```
    if (n == 100) return n;
}
```

g) `static int fact_G(int n) {` // assertion:  $0 < n \leq 100$  (4T)  
    `return help(n,100);`  
    `}`  
    `static int help(int e, f){`  
        `if (e == f) return f;`  
        `return f * help(e, f - 1);`  
    `}`

### 3.3 Die Funktion down (5T)

Gegeben ist folgende rekursive Funktion:

```
static int down(int n) { // assertion: n>0
    if (n > 20) return down(down(n - 1));
    return n;
}
```

- a) Geben Sie eine nichtrekursive Funktion an, die down berechnet. (2T)  
b) Beweisen Sie Ihre Aussage mit einer Induktion über n. (3T)