

## Übungsblatt 8

Abgabe bis Dienstag, 19.12.06, 11 Uhr

**Hinweis:** Programmieraufgaben immer per Email (eine Email pro Blatt und Gruppe) an den zuständigen Tutor schicken (Java Quellcode und eventuell benötigte Datendateien). Bitte werfen Sie Ihre schriftlichen Lösungen in die Briefkästen in Geb. 051, Erdgeschoss ein. Für den Erhalt von Bonuspunkten müssen Sie in wenigstens 10 Übungen anwesend sein.

### Aufgabe 1

1. Schreiben Sie eine rekursive Java Methode `fakultaet`, die zu einer Eingabe  $n \in \mathbb{N}$  die Fakultät ( $n! = 1 \cdot 2 \cdot \dots \cdot n$ ) berechnet.
2. Zeichnen Sie die Activation Records für den Aufruf `fakultaet(5)` zu dem Zeitpunkt, an dem die maximale Rekursionstiefe erreicht ist.
3. Schreiben Sie nun eine interative Methode, die  $n!$  berechnet.
4. Schreiben Sie ein Java Programm, die die Fakultät für alle Zahlen zwischen 1 und 10 berechnet und auf dem Bildschirm ausgibt.

### Aufgabe 2

Betrachten Sie die folgende Funktion  $f : \mathbb{N}_0 \rightarrow \mathbb{N}_0$  ( $\mathbb{N}_0$  sind die natürlichen Zahlen inklusive 0):

$$f(n) = \begin{cases} 0 & , n = 0 \\ 1 & , n = 1 \\ f(n-1) + f(n-2) & , n > 1 \end{cases}$$

1. Schreiben Sie eine rekursive Java-Methode, die die Funktion  $f$  implementiert.
2. Schreiben Sie eine *nicht*-rekursive Java-Methode, die die Funktion  $f$  implementiert.

### Aufgabe 3

Betrachten Sie die folgende Methode zum Sortieren eines Arrays von ganzen Zahlen:

```
public static void sort(int a[]) {
    int i, j, v;
    int n = a.length;
    for (i = 1; i < n; i++) {
        v = a[i];
        j = i;
        while (j > 0 && a[j-1] > v) {
            a[j] = a[j-1];
            j = j-1;
        }
        a[j] = v;
    }
}
```

Machen Sie Laufzeitanalysen in Abhängigkeit von der Länge  $n$  des Arrays  $a$  für den Best, den Worst sowie für den Average Case (nehmen Sie in diesem Fall an, dass die `while`-Schleife in jeder Runde  $i/2$  oft durchlaufen wird).