Rekursion

Vorteile	Nachteile
• Probleme lassen sich eleganter und intuitiver lösen • Schrittweise Aufteilung in immer kleinere Probleme	• Bei jedem Aufruf: neue Methodenschachtel auf dem Stack anlegen • Stack hat nur begrenzte Größe • Jeder Methodenaufruf und Rücksprung zum Aufrufer kostet Zeit • Häufig langsamer als äquivalente Schleife

Rekursionstypen

Lineare Rekursion

Die Funktion ruft sich im Rekursionsfall genau einmal selbst auf.

```
public static int sum(int n) {
    if (n <= 0) {
        return 0;
    }
    return n + sum(n-1);
}
```

Endrekursion

Spezialfall der linearen Rekursion: im Rekursionsfall ist der (einzige) rekursive Aufruf die letzte Aktion. Endrekursive Funktionen lassen sich entrekursivieren.

```
public static int ggt(int a, int b) {
    if (a == b)
        return a;

    if (a < b)
        return ggt(a, b-a);

    return ggt(a-b, b);
}
```

Kaskadenförmige Rekursion

Die Funktion ruft sich im Rekursionsfall unter Umständen mehrfach selbst auf.

```
public static int fib(int n) {
    if (n == 0 | n == 1) {
        return 1;
    }

    return fib(n-1) + fib(n-2);
}
```

Verschachtelte Rekursion

Rekursiver Aufruf der Funktion zur Bestimmung der Parameter des rekursiven Aufrufs.
Kommt in der Praxis quasi nie vor...

```
public static int foo(int i) {
    // ...
    return foo(foo(i));
}
```

Verschränkte Rekursion

Zwei verschiedene Funktionen rufen sich im Rekursionsfall gegenseitig auf.

```
public static int foo(int i) {
    // ...
    return bar(i-1);
}

public static void bar(int i) {
    // ...
    return foo(i*2);
}
```

Vollständige Induktion

Gegeben ist folgende Funktion

```
static long sum(int n) {
    if (n == 0) {
        return 0;
    } else {
        return n + sum(n-1);
    }
}
```

zu zeigen ist folgende Aussage wahr

$$\forall n \in \mathbb{N}, n \geq 0: \quad \texttt{sum(n)} \equiv \sum_{i=0}^{n} i =: S_n$$

Induktionsanfang Basisfall n=0:

$$\texttt{sum(n)} \overset{n=0}{\equiv} \texttt{sum(0)} \overset{if-then}{\equiv} 0 \equiv \sum_{i=0}^{0} i \equiv S_0$$

Induktionsvorraussetzung (n-1):

$$\texttt{sum(n-1)} \equiv S_{n-1} \equiv \sum_{i=0}^{n-1} i$$

Induktionsschritt (n-1→n):

$$\texttt{sum(n)} \overset{if-else}{\equiv} \texttt{n + sum(n-1)} \overset{IV}{\equiv} n + \sum_{i=0}^{n-1} i \equiv \sum_{i=0}^{n} i \equiv S_n \quad \blacksquare$$

Bei Induktion mit mehreren Induktionsanfängen

```
static long lf(int n) { // returns the left factorial !n
    if      (n <= 0) return 0;
    else if (n == 1) return 1;
    else             return n * lf(n - 1) - (n - 1) * lf(n - 2);
}
```

Zeigen Sie, dass folgende Aussage gilt:

$$\forall n \geq 0: \quad \texttt{lf(n)} \equiv \sum_{k=0}^{n-1} k!$$

Induktionsanfang 2 Basisfälle n=0 und n=1:

IA_0 $(n = 0)$:

$$\sum_{k=0}^{0-1} k! = 0 \equiv 0 = \texttt{lf(0)}$$

IA_1 $(n = 1)$:

$$\sum_{k=0}^{1-1} k! = 0! = 1 \equiv 1 = \texttt{lf(1)}$$

Induktionsvorraussetzungen:

IV_{n-1} $(n-1)$:

$$\texttt{lf(n-1)} \equiv \sum_{k=0}^{(n-1)-1} k! = \sum_{k=0}^{n-2} k!$$

IV_n (n):

$$\texttt{lf(n)} \equiv \sum_{k=0}^{n-1} k!$$

$$\texttt{lf(n+1)} = (n+1) \cdot \texttt{lf((n+1) - 1)} - ((n+1) - 1) \cdot \texttt{lf((n+1) - 2)}$$

$$= (n+1) \cdot \texttt{lf(n)} - n \cdot \texttt{lf(n - 1)} \overset{IV_{n-1}, IV_n}{\equiv}$$

$$\equiv (n+1) \cdot \sum_{k=0}^{n-1} k! - n \cdot \sum_{k=0}^{n-2} k! =$$

$$= n \cdot \sum_{k=0}^{n-1} k! + \sum_{k=0}^{n-1} k! - n \cdot \sum_{k=0}^{n-2} k! =$$

$$= n \cdot (n-1)! + n \cdot \sum_{k=0}^{n-2} k! + \sum_{k=0}^{n-1} k! - n \cdot \sum_{k=0}^{n-2} k! =$$

$$= n! + \sum_{k=0}^{n-1} k! = \sum_{k=0}^{(n+1)-1} k! \quad \blacksquare$$

Totale Korrektheit

- Terminierung gehört zur (totalen) Korrektheit einer rekursiven Methode dazu
  - Rekursion muss nach endlich vielen Schritten fertig sein
- Gesucht wird eine Terminierungsfunktion T(n) mit den folgenden Eigenschaften
  - Werte von T(n) sind ganzzahlig
  - Die Folge der T(n) ist streng monoton fallend
  - T(n) ist nach unten beschränkt meist >=0

```java
static long sum(int n) {
    if (n == 0) {
        return 0;
    } else {
        return n + sum(n-1);
    }
}
```

$$\forall n \in \mathbb{N}, n \geq 0 : \ \texttt{sum(n)} \ \equiv \sum_{i=0}^{n} i \ =: \ S_n$$

Eine passende Terminierungsfunktion ist z.B.: $T(n) = n$

- Werte sind ganzzahlig (`int` bzw. $n \in \mathbb{N}$),
- streng monoton fallend (rekursiver Aufruf mit $n - 1$) und
- nach unten beschränkt (Basisfall `n==0` bzw. $n \geq 0$)