Tagesprogramm

Vollständige Induktion

Einfache rekursive Methoden

Vollständige Induktion als Beweismethode

zwei Schritte um $\mathbf{A(n)}$ für alle $\mathbf{n \geq m}$ zu zeigen:

Induktionsanfang: zeige $\mathbf{A(m)}$

Induktionsschritt: zeige $\mathbf{A(n+1)}$ unter Annahme $\mathbf{A(n)}$ und $\mathbf{n \geq m}$

Gaußsche Summenformel als Beispiel

zu Beweisen: $\text{sum}(n) = 1 + 2 + \cdots + n = \frac{n(n+1)}{2}$

Induktionsanfang: $\text{sum}(1) = \frac{1(1+1)}{2}$ oder $\text{sum}(0) = \frac{0(0+1)}{2}$

Induktionsschritt: $\text{sum}(n+1) = \text{sum}(n) + (n+1) = \frac{n(n+1)}{2} + (n+1)$

$= \frac{n(n+1)+2(n+1)}{2} = \frac{(n+1)(n+2)}{2}$

Verwendung der gaußschen Summenformel

```
public class SummeGauss {
    public static void main(String[] args) {
        System.out.println(sum(3));
    }

    private static int sum(int n) {
        if (n <= 0) {
            return 0;
        } else {
            return  (n * (n + 1)) / 2;
    }
}
```

Aufgabe: Geeigneter Induktionsanfang?

Wir wollen für alle **n** $\geq$ **m** zeigen, dass **n*n** $>$ **n** gilt.
Der Induktionsschritt ist klar:

wenn **n*n** $>$ **n** gilt, dann gilt auch **(n+1)*(n+1)** $>$ **n+1**

(n $>$ 1 (n+1)*n+n+1 $>$ n+1 n*n+n+n $>$ n)

Welches **m** ist als Induktionsanfang geeignet?

-1

0

1

2

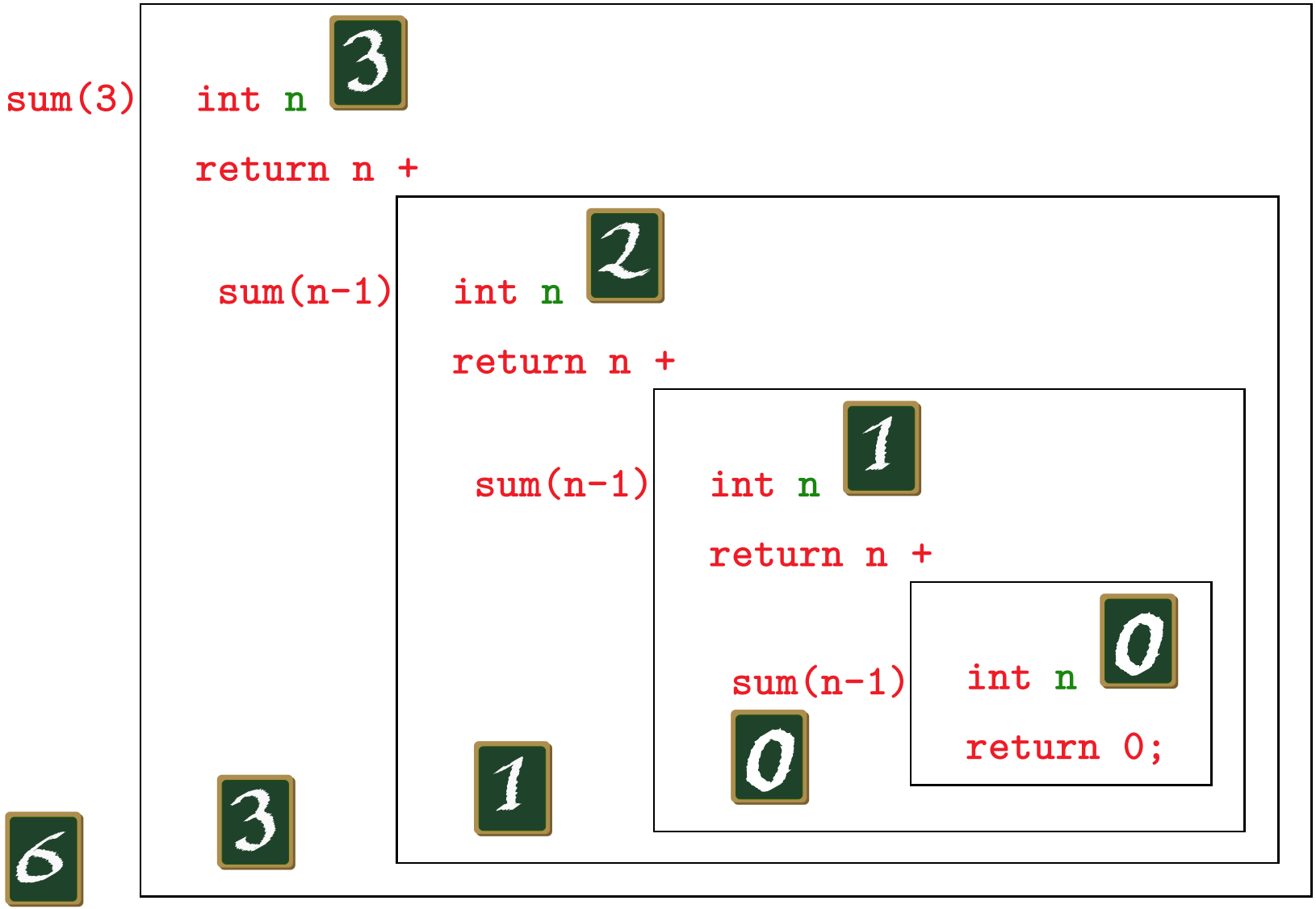
3

Verwendung des Konstruktionsprinzips

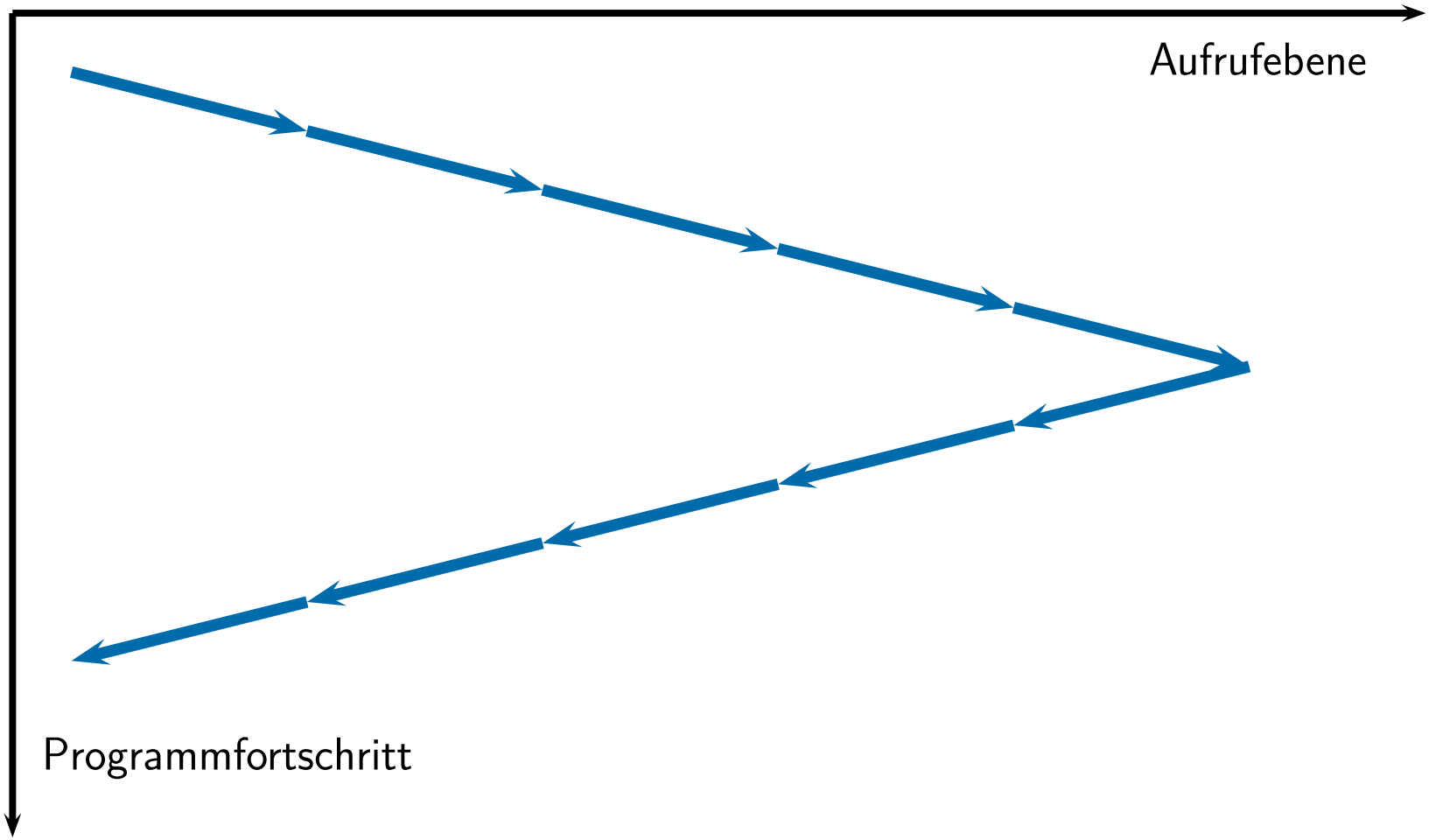
```
public class SummeRekursiv {
    public static void main(String[] args) {
        System.out.println(sum(3));
    }

    private static int sum(int n) {
        if (n <= 0) {                    // Abbruchbedingung
            return 0;
        } else {
            return  n + sum(n - 1);      // Rekursionsschritt
    }
}
```

Schematische Darstellung rekursiver Aufrufe



Abstrakte schematische Darstellung



Tabellarische Darstellung

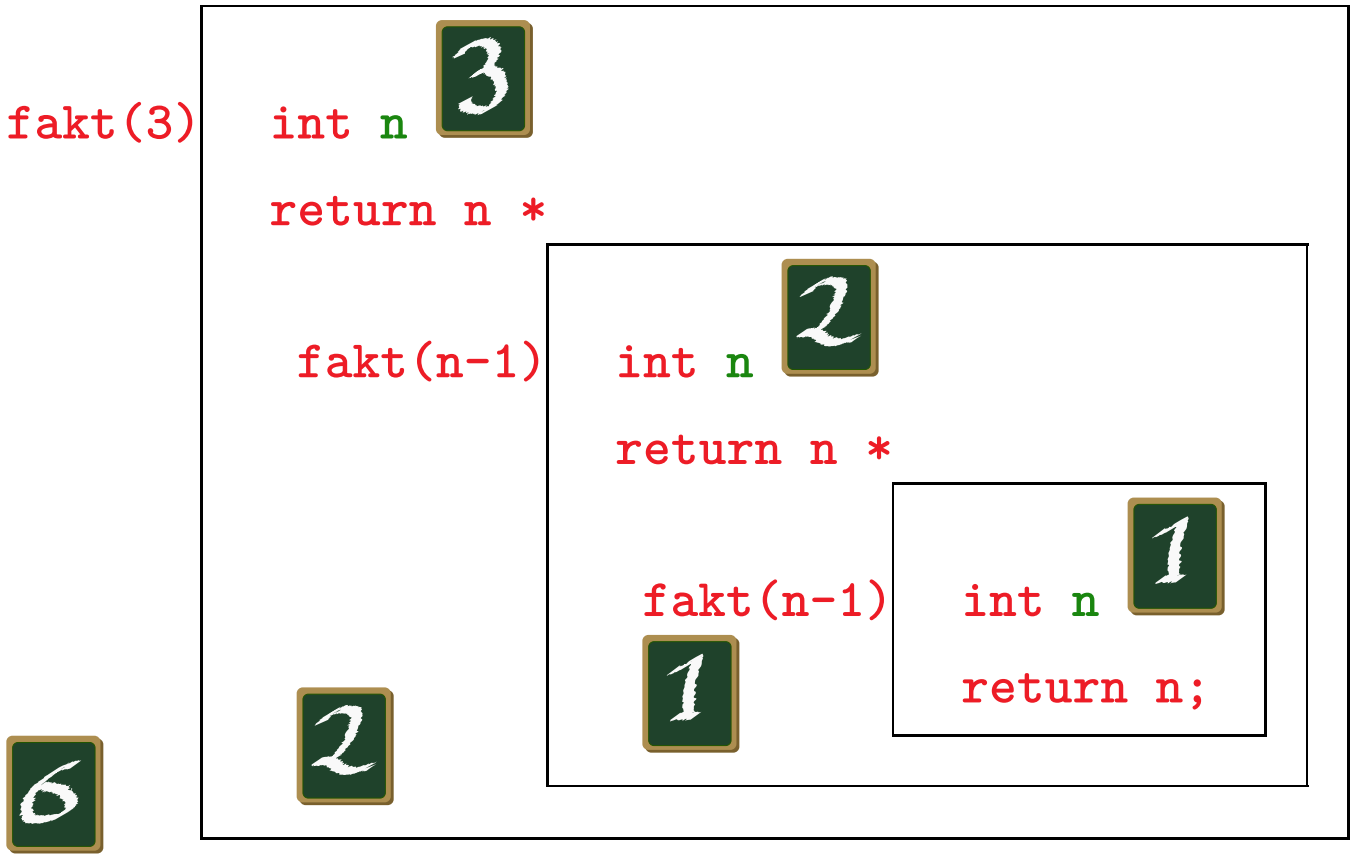
Methode	Aufrufebene	Argumente	Ergebnis
main	0	-	-
sum	1	3	6
sum	2	2	3
sum	3	1	1
sum	4	0	0
println	1	6	-

Faktorielle rekursiv berechnet

```
public class FaktorelleRekursiv {
    public static void main(String[] args) {
        System.out.println(fakt(3L));
    }

    private static long fakt(long n) {
        if (n > 1) {
            return  n * fakt(n - 1);
        } else {
            return n;
    }
}
```

Schematische Darstellung für Faktorielle



Aufgabe: Tabellarische Darstellung

Erstellen Sie in Gruppen zu 2 bis 3 Personen eine tabellarische Aufstellung aller Methodenaufrufe in einer Ausführung von `FaktorelleRekursiv`:

Methode	Aufrufebene	Argumente	Ergebnis
main	0	-	-
...	...	...	...

```
public class FaktorelleRekursiv {
    public static void main(String[] args) {
        System.out.println(fakt(3L));
    }

    private static long fakt(long n) {
        if (n > 1) {
            return  n * fakt(n - 1);
        } else {
            return n;
    }
}
```

Faktorielle rekursiv als Ausdruck

```
public class FaktorelleRekursivAusdruck {
    public static void main(String[] args) {
        System.out.println(fakt(5));
    }

    private static long fakt(long n) {
        return  n > 1  ?  n * fakt(n - 1)  :  n;
    }
}
```

Aufgabe: Rekusion und Schleifen

Offensichtlich kann man Rekursion manchmal anstelle von Schleifen einsetzen.

Suchen Sie in Gruppen zu 2 bis 3 Personen Antworten auf diese Fragen:

1. Kann man Schleifen **stets** durch Rekursion ersetzen?
   Wo könnte dies schwierig sein?

2. Kann man Rekursion **stets** durch Schleifen ersetzen?
   Wo könnte dies schwierig sein?