Rekursion

Vorsemesterkurs Informatik
Wintersemester 2020/21
Ronja Düffel

23. Oktober 2020

Rekursion

Rekursion

Rekursion

Definition

Eine rekursive Funktion ist eine Funktion, die durch sich selbst definiert wird.
Rekursionsanfang: Fall (Fälle) für den (die) die Funktion nicht wieder selbst aufgerufen wird
Rekursionsschritt: rekursiver Aufruf der Funktion.

Beispiel (Fakultätsfunktion)

$f : \mathbb{N}_0 \to \mathbb{N}_0$

$$f(n) := \begin{cases} 1, & \text{falls } n = 0 \\ n \cdot f(n-1), & \text{sonst.} \end{cases}$$

Fakultätsfunktion

$f : \mathbb{N}_0 \to \mathbb{N}_0$

$$f(n) := \begin{cases} 1, & \text{falls } n = 0 \\ n \cdot f(n-1), & \text{sonst.} \end{cases}$$

$$\begin{aligned} f(0) &= 1 \\ f(1) &= 1 \cdot f(0) = 1 \cdot 1 = 1 \\ f(2) &= 2 \cdot f(1) = 2 \cdot (1 \cdot f(0)) = 2 \cdot (1 \cdot 1) = 2 \\ f(3) &= 3 \cdot f(2) = 3 \cdot (2 \cdot (1 \cdot 1)) = 6 \end{aligned}$$

Man schreibt auch $f(n) = n!$

Satz

Für die Fakultätsfunktion $f(n)$ gilt: $f(n) = \prod_{i=1}^{n} i$.

Beweis durch vollständige Induktion

Induktionsanfang: $n = 0$

Behauptung: Der Satz gilt für $n = 0$.

Beweis: Es gilt: $f(0) = 1$. Ferner gilt: $\prod_{i=1}^{0} i = 1$.
Somit gilt: $f(0) = 1 = \prod_{i=1}^{0} i$.

Induktionsschritt: $n \to n + 1$

Induktionsvoraussetzung: Für $n \in \mathbb{N}$ gilt: $f(n) = \prod_{i=1}^{n} i$.

Induktionsbehauptung: Es gilt: $f(n+1) = \prod_{i=1}^{n+1} i$.

Induktionsbehauptung: Es gilt: $f(n+1) = \prod_{i=1}^{n+1} i$.

Beweis:

$$
\begin{aligned}
f(n+1) &= (n+1) \cdot f(n) && \text{Induktionvoraussetzung} \\
&= (n+1) \cdot \prod_{i=1}^{n} i \\
&= (n+1) \cdot n \cdot (n-1) \cdot \dots \cdot 2 \cdot 1 \\
&= \prod_{i=1}^{n+1} i
\end{aligned}
$$

□

Sind wir fertig?

1. Wir haben die Behauptung für $n = 0$ gezeigt.

2. Wir haben gezeigt:

   Wenn die Behauptung für n gilt, **dann** gilt sie auch für $n + 1$.

   wichtig: Verwendung der Induktionsvoraussetzung

ideale Kaninchen

- Ein Kaninchenpaar (m,w)
- ist nach einem Monat geschlechtsreif
- gebären nach einem Monat Tragzeit ein weiteres Kaninchenpaar
- Kaninchen sterben nie
- Wie viele Kaninchenpaare nach n Monaten (K_n)?

Karnickel

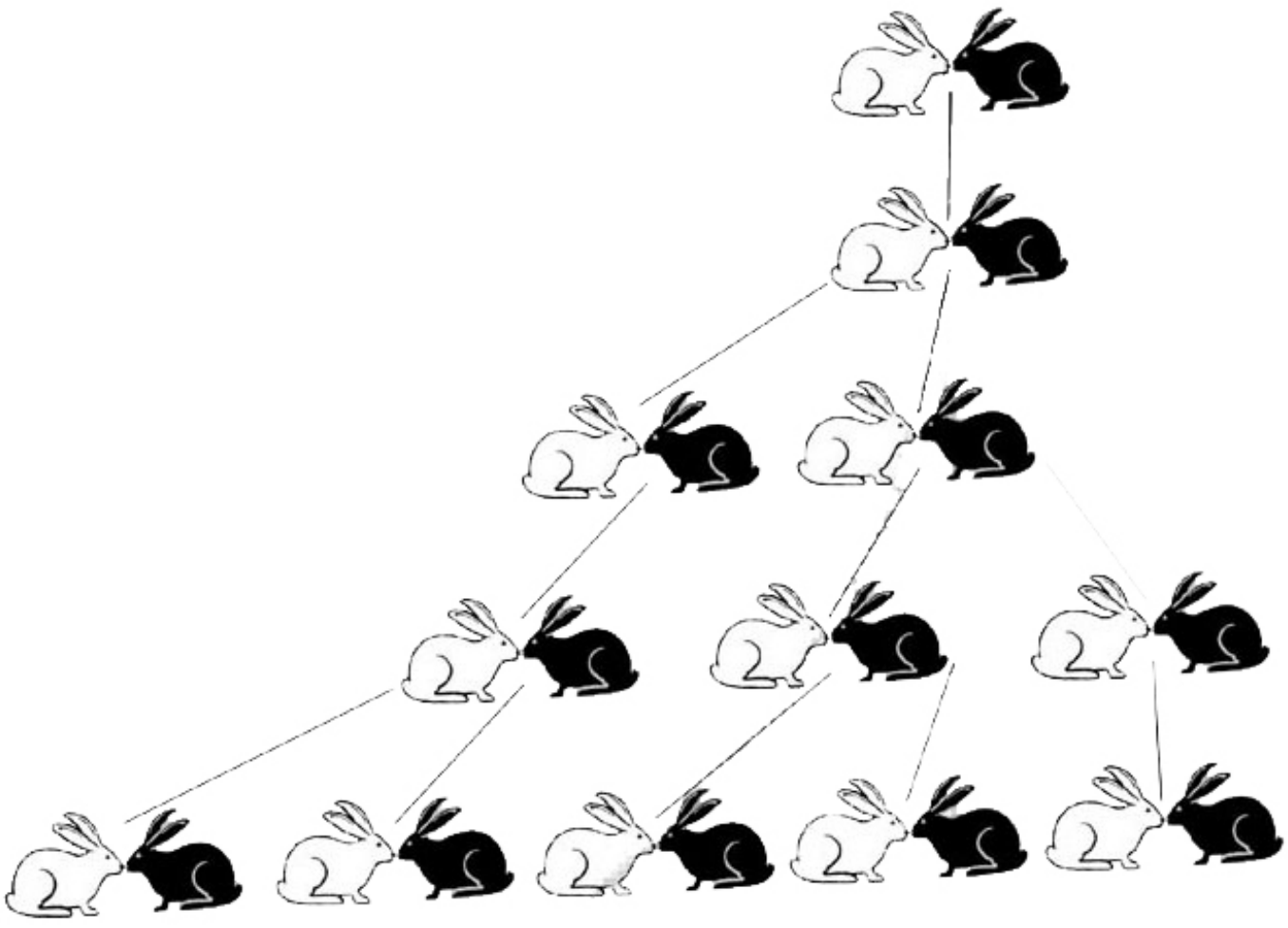
1. Monat $\rightarrow$ 1 Paar

2. Monat $\rightarrow$ 1 Paar

3. Monat $\rightarrow$ 2 Paare

4. Monat $\rightarrow$ 3 Paare

5. Monat $\rightarrow$ 5 Paare



Karnickel

Wie viele (ideale) Kaninchenpaare K_n habe ich nach n Monaten?

- Die Kaninchen sterben nie

$$K_n = K_{n-1} + \text{ die Neugeborenen}$$

- ab dem zweiten Lebensmonat, jeden Monat ein neues Paar

  $\rightarrow$ alle Paare, die mindestens 2 Monate alt sind, bekommen Nachwuchs

  $\rightarrow$ Anzahl der Neugeborenen Paare in Monat n: K_{n-2}

$$K_n = K_{n-1} + K_{n-2}$$

Türme von Hanoi

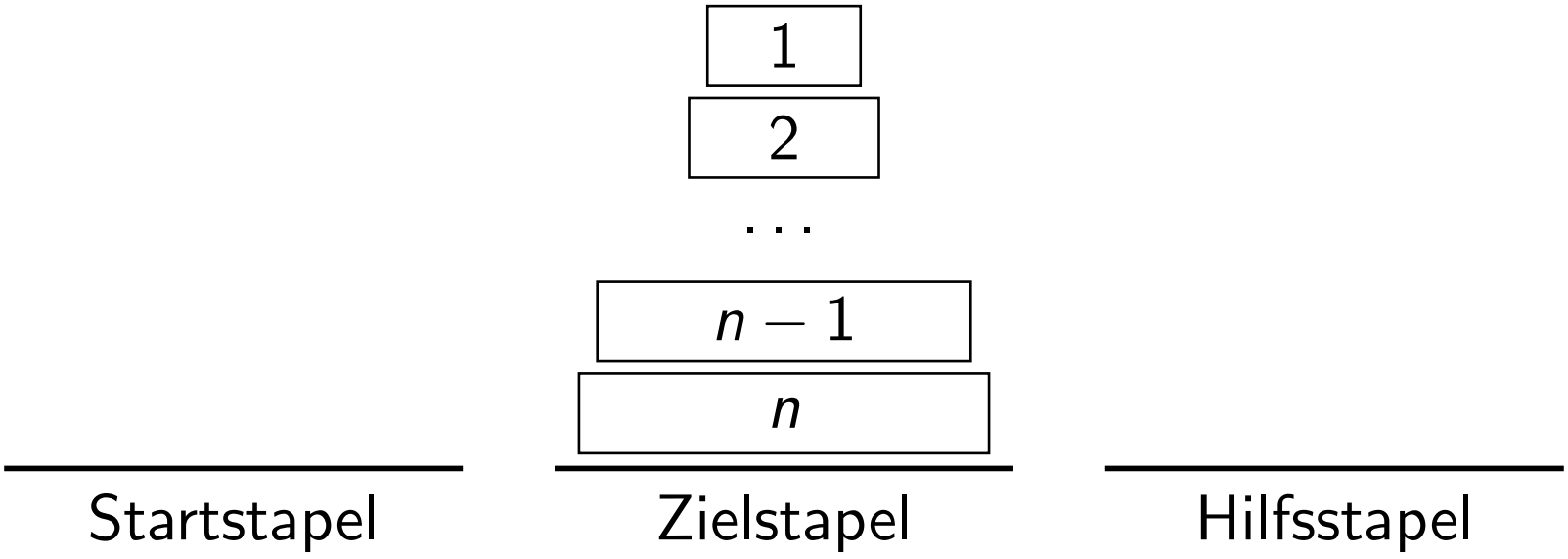
Regeln:

- nur eine Scheibe bewegen
- niemals eine größere auf eine kleinere Scheibe legen.

Rekursion: Türme von Hanoi

Startsituation

Rekursion: Türme von Hanoi



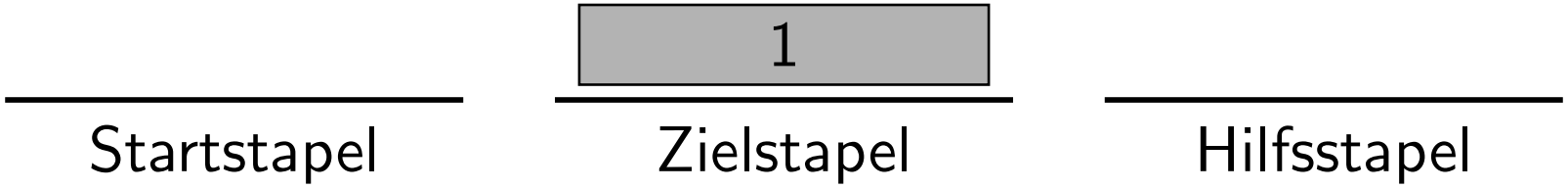
Zielsituation

Beispiel $n = 3$

Lösen durch Rekursion: Rekursionanfang

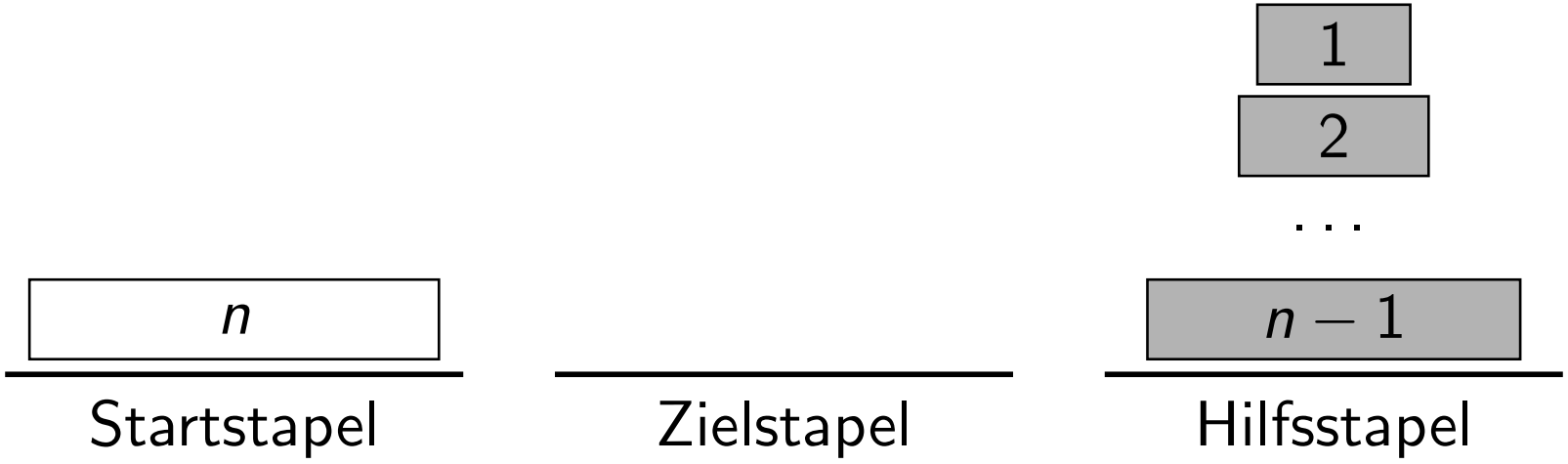
$n = 1$: Verschiebe Scheibe von Startstapel auf Zielstapel

Lösen durch Rekursion: Rekursionanfang



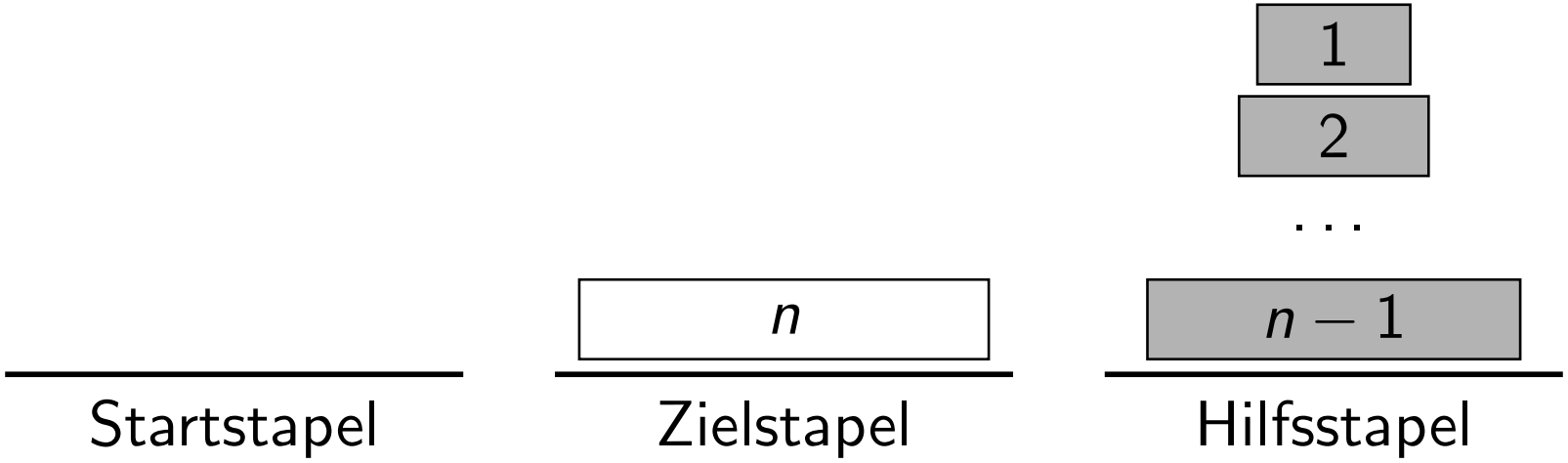
$n = 1$: Verschiebe Scheibe von Startstapel auf Zielstapel

Lösen durch Rekursion: Rekursionsschritt



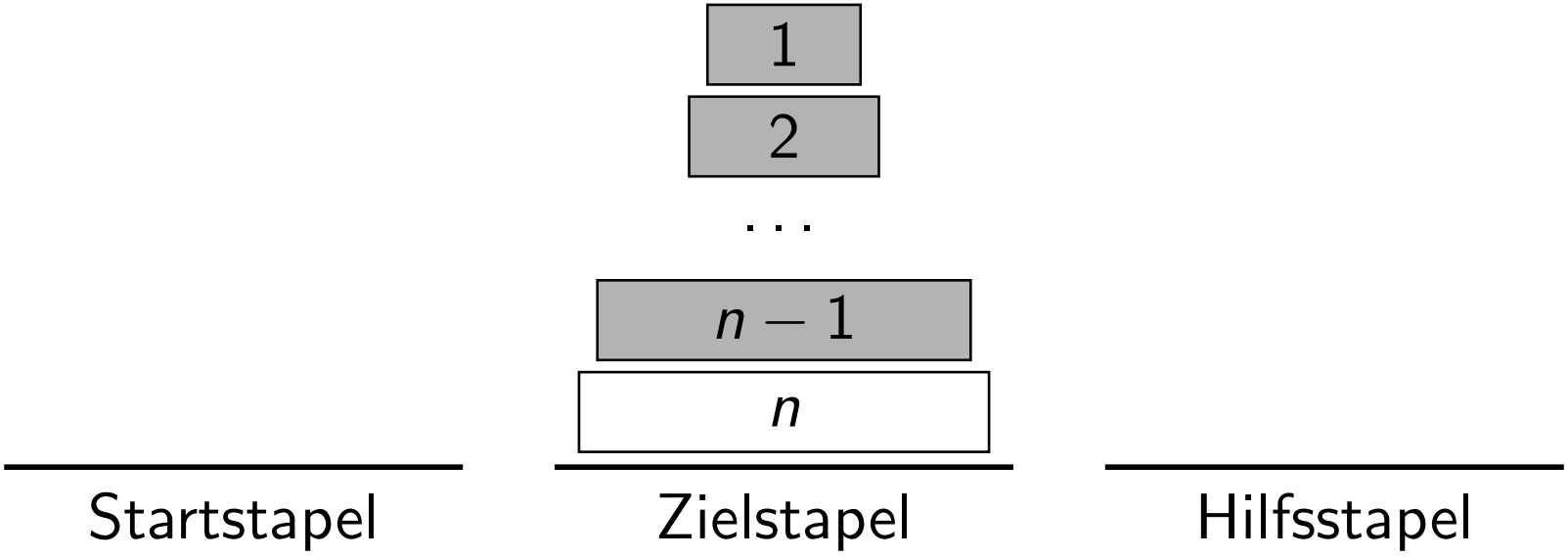
1. Verschiebe den Turm der Höhe $n-1$ **rekursiv** auf den Hilfsstapel

Lösen durch Rekursion: Rekursionsschritt



2. Verschiebe Scheibe n auf den Zielstapel

Lösen durch Rekursion: Rekursionsschritt



3. Verschiebe den Turm der Höhe $n-1$ **rekursiv** auf den Zielstapel

Pseudo-Algorithmus

verschiebe(n,start,ziel,hilf)

1. Wenn $n > 1$, dann verschiebe(n-1,start,hilf,ziel)
2. Schiebe Scheibe n von start auf ziel
3. Wenn $n > 1$, dann verschiebe(n-1,hilf,ziel,start)

- Rekursionanfang ist bei $n = 1$: keine rekursiven Aufrufe
- Beachte: Zwei rekursive Aufrufe pro Rekursionsschritt

Algorithmus Türme von Hanoi

Algorithmus

```
1 function hanoi(n, start, ziel, hilf){
2     if (n > 1){
3         hanoi(n-1,start,hilf,ziel)
4     }
5     verschiebe Scheibe n von start auf ziel
6     if (n > 1){
7         hanoi(n-1,hilf,ziel,start)
8     }
9 }
```

Anzahl der Spielzüge

$n = 1$	Schiebe 1 von A nach B	1 Zug
$n = 2$	$1\curvearrowright$ C, $2\curvearrowright$ B, $1\curvearrowright$ B	3 Züge
$n = 3$	$1\curvearrowright$ B, $2\curvearrowright$ C ,$1\curvearrowright$ C, $3\curvearrowright$ B, $1\curvearrowright$ A, $2\curvearrowright$ B, $1\curvearrowright$ B	7 Züge

Satz

Um n Scheiben von einem Stapel zu einem anderen zu transportieren, werden mindestens $2^n - 1$ Spielzüge benötigt.

Beweis Anzahl der Spielzüge

Induktionsanfang: $n = 1$

Behauptung: Um eine Scheibe von A nach B zu setzen, werden $2^1 - 1 = 1$ Spielzüge benötigt.

Beweis: offensichtlich korrekt, wir setzen die Scheibe von A nach B.

Induktionsschritt: $n \to n + 1$

Induktionsvoraussetzung: Um n Scheiben von einem zu einem anderen Stapel zu transportieren, werden $2^n - 1$ Spielzüge benötigt.

Induktionsbehauptung: Um $n + 1$ Scheiben von einem zu einem anderen Stapel zu transportieren, werden $2^{n+1} - 1$ Spielzüge benötigt.

Induktionsbehauptung: Um $n+1$ Scheiben von einem zu einem anderen Stapel zu transportieren, werden $2^{n+1}-1$ Spielzüge benötigt.

Beweis:

1. transportiere n Scheiben von start auf hilf
2. transportiere Scheibe $n+1$ von start auf ziel
3. transportiere n Scheiben von hilf auf ziel

$$2^n - 1 + 1 + 2^n - 1 = 2^n + 2^n - 1 + 1 - 1 = 2 \cdot 2^n - 1 = 2^{n+1} - 1$$

□

Turm von Benares

- bei 64 Scheiben, benötigen die Mönche $2^{64} - 1$ Züge
- bei 1 Zug pro Sekunde sind das 18 446 744 073 709 551 615 Sekunden
- das entspricht 584 942 417 400 Jahren

Fragen?