

# Beweise: Warum?

Wir möchten ein Phänomen erklären.

- Häufig ist das Phänomen aber so komplex, dass eine vollständige Erklärung nicht zu erwarten ist.
  - ▶ Partielle Erklärungen wie etwa Erfahrungswerte und Ergebnisse experimenteller Arbeit müssen genügen.
- In einigen Fällen sind aber unumstößlich wahre Aussagen nicht nur möglich, sondern werden sogar verlangt:

*Es genügt nicht, dass eine sicherheitssensitive Software funktionieren sollte, sie **muss** funktionieren!*

Wir benutzen die Sprache der Mathematik, um die Korrektheit einer Aussage zweifelsfrei nachzuweisen.

- Beweise eine **existentielle Aussagen** der Form

„Es gibt ein Objekt mit der Eigenschaft  $E$ “

zum Beispiel mit der Methode der **Konstruktion**: Beschreibe ein spezifisches Objekt und weise nach, dass dieses Objekt die Eigenschaft  $E$  besitzt.

- ▶ Die existentielle Aussage „das quadratische Polynom  $x^2 - 1$  besitzt eine reellwertige Nullstelle“ ist zu beweisen: Zeige, dass 1 eine Nullstelle ist.

- Die **All-Aussage**

„Alle fraglichen Objekte besitzen die Eigenschaft  $E$ “

ist zu untersuchen.

- ▶ **Widerlege** die All-Aussage mit einem **Gegenbeispiel**: Die existentielle Aussage „Es gibt ein Objekt mit der Eigenschaft  $\neg E$ “ ist die Negation der All-Aussage.
  - ★ Widerlege die All-Aussage „jedes quadratische Polynom besitzt eine reellwertige Nullstelle“ durch das quadratische Polynom  $x^2 + 1$ .
- ▶ **Beweise** die All-Aussage durch den Nachweis, dass ein **beliebiges** Objekt die Eigenschaft  $E$  besitzt.

# Satz, Behauptung und Beweis

- 1 Ein **Satz** besteht aus **Voraussetzungen** und einer **Behauptung**: Wenn alle Voraussetzungen erfüllt sind, dann muss die Behauptung wahr sein.
- 2 Der **Beweis** eines Satzes muss nachweisen, dass die Behauptung des Satzes wahr ist und kann dabei verwenden:
  - ▶ die Voraussetzungen des Satzes,
  - ▶ Definitionen
  - ▶ bereits bekannte Tatsachen und Sätze,
  - ▶ im Beweis selbst oder anderswo bereits als wahr bewiesene Aussagen,
  - ▶ **logische Schlussregeln**.

(\*) Bisher haben wir semantische Folgerungen  $\Phi \models \phi$  für aussagenlogische Formeln bewiesen.

(\*) Jetzt sind kompliziertere Aussagen

*möglicherweise mit Quantoren, Relationen und Funktionen*

zu beweisen. Aber die Aussagenlogik wird uns helfen!

# Logische Schlussregeln

$\phi$  und  $\psi$  sind beliebige, nicht notwendigerweise aussagenlogische Aussagen.

- 1 Ein **direkter Beweis** geht „ohne Umschweife“ vor.
- 2 Die Beweismethode der **Kontraposition** beruht auf der Äquivalenz

$$(\phi \rightarrow \psi) \equiv (\neg\psi \rightarrow \neg\phi).$$

Um die Implikation  $\phi \rightarrow \psi$  zu beweisen:

- ▶ Nimm an, dass  $\neg\psi$  gilt und
- ▶ zeige die Aussage  $\neg\phi$ .

- 3 Im **Beweis durch Widerspruch** wird die Äquivalenz

$$(\phi \rightarrow \psi) \equiv ((\phi \wedge \neg\psi) \rightarrow \mathbf{0})$$

ausgenutzt: Um  $\phi \rightarrow \psi$  zu beweisen:

- ▶ Nimm an, dass die Voraussetzung  $\phi$  wahr, der Schluss  $\psi$  aber falsch ist und leite einen Widerspruch her, d.h. folgere eine falsche Aussage.

- 4 Später wenden wir die mächtige Methode der **vollständigen Induktion** an.

# Beweise: Typische Fehler

- Wenn eine All-Aussage zu zeigen ist, dann genügt ein „**Beweis durch Beispiel**“ natürlich nicht: Die Aussage ist nicht nur für einige Beispiele zu verifizieren, sondern ist für alle Instanzen zu zeigen.
- Die Notation kann uns einen Streich spielen, wenn gleiche Symbole zur Bezeichnung verschiedener Dinge verwendet werden.
- Die Bedeutung eingeführter Begriffe muss klar sein und darf nicht vom Kontext abhängen.
- Unzulässige Gedankensprünge beim Schlussfolgern bedeuten, dass das Argument unvollständig ist und
- das Ausnutzen von bis dahin noch unbewiesenen Behauptungen ist ein Fehler im Argument.

# Direkte Beweise

# Wir haben bereits direkte Beweise geführt.

Wir haben gezeigt, dass eine endliche Menge  $M$  genau  $2^{|M|}$  Teilmengen besitzt. Wie sahen die Beweisschritte aus?

1. Zuerst haben wir eine bijektive Funktion

$$f : \mathcal{P}(M) \rightarrow \text{Abb}(M, \{0, 1\})$$

von der Potenzmenge  $\mathcal{P}(M)$  nach  $\text{Abb}(M, \{0, 1\})$  konstruiert.

- ▶  $\mathcal{P}(M)$  und  $\text{Abb}(M, \{0, 1\})$  sind also gleichgroß.

2. Dann haben wir eine bijektive Funktion

$$g : \text{Abb}(A, B) \rightarrow B^{|A|}$$

sogar für beliebige endliche Mengen  $A, B$  „gebaut“ und damit gezeigt

$$|\text{Abb}(A, B)| = |B|^{|A|}.$$

3. Jetzt müssen wir nur noch  $A = M$  und  $B = \{0, 1\}$  setzen. Es folgt

$$|\mathcal{P}(M)| \stackrel{1.}{=} |\text{Abb}(M, \{0, 1\})| \stackrel{2.}{=} 2^{|M|}.$$

$$\frac{a+b}{2} \geq \sqrt{a \cdot b} \quad \text{für alle reellen Zahlen } a, b \geq 0 \quad (1/2)$$

Zuerst versuchen wir eine Beweisidee zu erhalten.

1. Die Wurzel stört und wir quadrieren:

▶ Statt  $\frac{a+b}{2} \geq \sqrt{a \cdot b}$  zeige die Ungleichung  $(\frac{a+b}{2})^2 \geq a \cdot b$ .

2. Wir multiplizieren aus und erhalten die Ungleichung  $\frac{a^2+2a \cdot b+b^2}{4} \geq a \cdot b$ .

3. Wir multiplizieren mit 4 und erhalten  $a^2 + 2a \cdot b + b^2 \geq 4a \cdot b$ .

4. Wenn wir  $4a \cdot b$  nach links „bringen“, ist  $a^2 - 2a \cdot b + b^2 \geq 0$  zu zeigen.

▶  $a^2 - 2a \cdot b + b^2 = (a - b)^2$  gilt.

▶ Jedes Quadrat ist nicht-negativ und die Ungleichung stimmt!?!)

Das ist leider kein Beweis, weil ....

.... wir **aus** der Ungleichung  $\frac{a+b}{2} \geq \sqrt{a \cdot b}$  eine wahre Aussage folgern :-(((



$$\frac{a+b}{2} \geq \sqrt{a \cdot b} \quad \text{für alle reellen Zahlen } a, b \geq 0 \quad (2/2)$$

Hoffentlich haben wir nur mit äquivalenten Umformungen gearbeitet: Mal sehen.

1.  $a^2 - 2a \cdot b + b^2 = (a - b)^2$  gilt und  $a^2 - 2a \cdot b + b^2 \geq 0$  folgt.
2. Wir addieren  $4a \cdot b$  auf beide Seiten:  $a^2 + 2a \cdot b + b^2 \geq 4a \cdot b$  gilt ebenfalls.
3. Die linke Seite der Ungleichung stimmt mit  $(a + b)^2$  überein: Es gilt also

$$(a + b)^2 \geq 4a \cdot b.$$

4. Wir dividieren beide Seiten durch 4 und ziehen die Wurzel:

$$\frac{a + b}{2} \geq \sqrt{ab}$$

und das war zu zeigen.

Na, geht doch!

# Beweis durch Kontraposition

# Wenn $n \in \mathbb{N}$ und $n^2$ gerade ist, dann ist auch $n$ gerade

Im Beweis durch Kontraposition genügt der Nachweis,

Wenn  $n$  ungerade ist, dann ist auch  $n^2$  ungerade.

Sei also  $n$  ungerade.

1. Eine Zahl  $m \in \mathbb{N}$  ist **genau dann** ungerade, wenn es eine Zahl  $k \in \mathbb{N}$  gibt mit

$$m = 2k + 1.$$

2.  $n$  ist nach Annahme ungerade und es gibt  $k \in \mathbb{N}$  mit  $n = 2k + 1$ .
3. Wir quadrieren  $n$  und erhalten

$$n^2 = (2k + 1)^2 = 4k^2 + 4k + 1 = 2(2k^2 + 2k) + 1$$

und damit ist auch  $n^2$  ungerade.

# Beweis durch Widerspruch

# $\sqrt{2}$ ist keine rationale Zahl.

Wir nehmen an, dass  $\sqrt{2}$  eine rationale Zahl ist und leiten einen Widerspruch her.

1. Wenn  $\sqrt{2}$  eine rationale Zahl ist, dann gibt es **teilerfremde** Zahlen  $p, q \in \mathbb{N}$  mit

$$\sqrt{2} = \frac{p}{q}.$$

2. Wir quadrieren und erhalten die Gleichung

$$p^2 = 2 \cdot q^2.$$

3.  $\implies p^2$  ist eine gerade Zahl.

- ▶ Wir haben aber gerade gezeigt, dass dann auch **p gerade** ist.
- ▶ Wenn  $p = 2r$ , dann folgt also  $p^2 = 4r^2 = 2q^2$  und damit  $2r^2 = q^2$ .

4. Dann ist aber  $q^2$  gerade und damit auch **q**  $\implies$  Die Zahlen  $p$  und  $q$  haben den gemeinsamen Teiler 2 im **Widerspruch** zur Teilerfremdheit ⚡

# Die Grundlage der Public-Key Kryptographie

Eine Primzahl ist eine Zahl größer als Eins, die nur durch die Eins und sich selbst teilbar ist.

Der Satz von Euklid: Es gibt unendlich viele Primzahlen.

Angenommen, es gibt nur endlich viele Primzahlen, nämlich  $p_1, \dots, p_n$ .

1. Betrachte die Zahl  $N = p_1 \cdot \dots \cdot p_n + 1$ .

▶ Alle Primzahlen teilen die Zahl  $N - 1$  und können deshalb  $N$  nicht teilen.

2. Jede natürliche Zahl und damit auch  $N$  ist ein Produkt von Primzahlen.

▶ Alle Primteiler von  $N$  sind von  $p_1, \dots, p_n$  verschieden. ⚡

# Diagonalisierung

# Die Methode der Diagonalisierung

Es gibt **keine** surjektive Funktion von  $\mathbb{N}$  nach  $\mathcal{P}(\mathbb{N})$ .

- Die Potenzmenge von  $\mathbb{N}$  ist „**sehr viel**“ größer als  $\mathbb{N}$ .
- Es gibt „sehr viel mehr“ algorithmische Probleme als Python-Programme!

1. Im **Beweis durch Widerspruch** nehmen wir an, dass die Funktion

$$f: \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$$

surjektiv ist.

2. Die zentrale Idee: Wir definieren die Menge

$$M := \{n \in \mathbb{N} : n \notin f(n)\},$$

die offensichtlich eine Teilmenge von  $\mathbb{N}$  ist: Also gilt  $M \in \mathcal{P}(\mathbb{N})$ .



$$M = \{ n \in \mathbb{N} : n \notin f(n) \}$$

3. Da  $f$  surjektiv ist, muss es ein  $m \in \mathbb{N}$  geben mit  $f(m) = M$ .

▶ Klar: Entweder gilt  $m \in M$  oder es gilt  $m \notin M$ .

4. Fall 1:  $m \notin M$ :

▶ Nach Definition der Menge  $M$  folgt  $m \in f(m)$ .

▶ Es ist  $f(m) = M$  und deshalb folgt  $m \in f(m) = M$ . ⚡

Fall 2:  $m \in M$ :

▶ Nach Definition der Menge  $M$  folgt  $m \notin f(m)$ .

▶ Es ist  $f(m) = M$  und deshalb folgt  $m \notin f(m) = M$ . ⚡

5. In jedem Fall haben wir einen Widerspruch erhalten: Es gibt keine Zahl  $m$  mit  $f(m) = M$  und  $f$  ist im Gegensatz zur Annahme nicht surjektiv.  $\square$

Wir haben sogar mehr gezeigt:

Für **keine** Menge  $M$  gibt es eine surjektive Funktion  $f : M \rightarrow \mathcal{P}(M)$ .

Wie ist Cantor auf die Idee gekommen, die Menge  $M$  so zu definieren?

Eine Funktion  $f : \mathbb{N} \rightarrow \mathcal{P}(\mathbb{N})$  können wir durch folgende Tabelle repräsentieren

	0	1	2	3	4	5	...
0	$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$	$a_{0,4}$	$a_{0,5}$	...
1	$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{1,4}$	$a_{1,5}$	...
2	$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{2,4}$	$a_{2,5}$	...
3	$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$a_{3,4}$	$a_{3,5}$	...
4	$a_{4,0}$	$a_{4,1}$	$a_{4,2}$	$a_{4,3}$	$a_{4,4}$	$a_{4,5}$	...
5	$a_{5,0}$	$a_{5,1}$	$a_{5,2}$	$a_{5,3}$	$a_{5,4}$	$a_{5,5}$	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

wobei  $a_{i,j} := \begin{cases} 1 & \text{falls } j \in f(i) \\ 0 & \text{falls } j \notin f(i). \end{cases}$  der Eintrag in Zeile  $i$  und Spalte  $j$  ist.

Die Teilmenge  $f(i) \subseteq \mathbb{N}$  wird in Zeile  $i$  „Element für Element“ beschrieben.

Es ist  $M = \{n : n \notin f(n)\}$ .

1. Das unendlich lange Tupel  $b = (b_n : n \in \mathbb{N})$  beschreibe die Menge  $M \subseteq \mathbb{N}$ .
  - ▶ Dann ist  $b_n = 1 \iff n \in M \iff n \notin f(n) \iff a_{n,n} = 0$ .
  - ▶ Also ist  $b_n \neq a_{n,n}$ .
2.  $\implies$  das Tupel  $b$  unterscheidet sich von jeder Zeile  $f(n)$  der Tabelle und zwar unterscheidet es sich in der **Diagonalen** ( $b_n \neq a_{n,n}$ ).
  - ▶ Es gilt  $M \neq f(n)$  für jede Zahl  $n \in \mathbb{N}$ .

Jedes Python-Programm  $P$  lässt sich als natürliche Zahl  $ID(P)$  kodieren.

- Definiere zum Beispiel  $ID(P)$  durch die interne Binärdarstellung von  $P$ .

Es gibt kein Python-Programm  $P_0$ , so dass für alle Python-Programme  $P$ , die

(\*) die eine natürliche Zahl als Eingabe erwarten und

(\*) dann *akzeptieren*, *verwerfen* oder *nicht halten*,

gilt

$P_0$  akzeptiert  $ID(P) \iff P$  akzeptiert  $ID(P)$  nicht.

(In der „Theoretischen Informatik 1“ wird der Beweis gezeigt.) )

Es gibt also **keinen „Super-Compiler“**  $P_0$ , der voraussagt, ob ein Programm  $P$  eine Eingabe  $n \in \mathbb{N}$  akzeptiert oder verwirft!

*Denn sonst könnte  $P_0$  voraussagen, ob ein Anwenderprogramm  $P$  seine Kodierung  $ID(P)$  akzeptiert.*

# Vollständige Induktion

# Die Grundidee der vollständigen Induktion

$A(n)$  sei eine Aussage über die natürliche Zahl  $n$ .

**Zeige:**  $A(n)$  ist für jedes  $n \in \mathbb{N}$  wahr.

(a) **INDUKTIONSANFANG** bzw. **INDUKTIONSBASIS:**

Zuerst zeige, dass die Aussage  $A(n)$  für die Zahl  $n = 0$  gilt.

(b) **INDUKTIONSSCHRITT:** Für jede beliebige natürliche Zahl  $n \in \mathbb{N}$  zeige die Aussage  $A(n+1)$ , falls die Induktionsannahme  $A(n)$  wahr ist.

Wenn (a) und (b) bewiesen sind, dann wird eine **Lawine** losgetreten:

1.  $A(0)$  ist wahr gemäß Induktionsanfang (a).
2.  $A(1)$  ist wahr gemäß 1. und Induktionsschritt (b) für  $n = 0$ ,
3.  $A(2)$  ist wahr gemäß 2. und Induktionsschritt (b) für  $n = 1$ ,
4.  $A(3)$  ist wahr gemäß 3. und Induktionsschritt (b) für  $n = 2$ ,
5.  $A(4)$  ist wahr gemäß 4. und Induktionsschritt (b) für  $n = 3$ ,
- .....

Insgesamt hat man damit gezeigt, dass die Aussage  $A(n)$  **für alle**  $n \in \mathbb{N}$  wahr ist.

# Es ist richtig dunkel

Wir befinden uns in einem stockdunklen Gang, der nach einer Seite unbeschränkt lang ist. Den Gang können wir nur über die andere Seite verlassen.

- ? Was tun, wir kennen noch nicht einmal die Länge  $n$  des Weges bis zum Ende des Ganges!?!

Wie können wir mit möglichst wenigen Schritten den Gang verlassen?

- Wie wär's mit: Ein Schritt nach „vorn“, zwei zurück, drei nach vorn, vier zurück, ...
- Und wieviele Schritte sind das?

# Die Summe der ersten $n$ Zahlen

$$\sum_{i=1}^n i = 1 + 2 + \dots + n = \frac{n \cdot (n+1)}{2}.$$

- Vollständige Induktion nach  $n$ :

- ▶ **INDUKTIONSANFANG** für  $n = 0$ :  $\sum_{i=1}^0 i = 0$  und  $\frac{0 \cdot (0+1)}{2} = 0$ . ✓

- ▶ **INDUKTIONSSCHRITT** von  $n$  auf  $n+1$ : Sei  $n \in \mathbb{N}$  beliebig.

- ★ Wir können die Induktionsannahme  $\sum_{i=1}^n i = \frac{n \cdot (n+1)}{2}$  voraussetzen und müssen zeigen  $\sum_{i=1}^{n+1} i = (n+1) \cdot (n+2)/2$ .

- ★  $\sum_{i=1}^{n+1} i = \sum_{i=1}^n i + (n+1) \stackrel{IA}{=} \frac{n \cdot (n+1)}{2} + (n+1) = \frac{n \cdot (n+1) + 2 \cdot (n+1)}{2} = \frac{(n+2) \cdot (n+1)}{2} = \frac{(n+1) \cdot (n+2)}{2}$ . ✓

- Ein direkter Beweis:

- ▶ Betrachte ein Gitter mit  $n$  Zeilen und  $n$  Spalten:  $n^2$  Gitterpunkte.

- ▶ Wir müssen die Gitterpunkte unterhalb der Hauptdiagonale und auf der Hauptdiagonale zählen.

- ▶ Die Hauptdiagonale besitzt  $n$  Gitterpunkte und unterhalb der Hauptdiagonale befindet sich die Hälfte der verbleibenden  $n^2 - n$  Gitterpunkte. Also folgt

$$\sum_{i=1}^n i = n + \frac{n^2 - n}{2} = \frac{n \cdot (n+1)}{2}. \quad \checkmark$$



# Wie viele Schritte?

$n$  sei die Länge des Weges bis zum Ende des Gangs.

1. Nach  $2k$  Wiederholungen sind wir insgesamt

$$(1 - 2) + (3 - 4) + \cdots + (2k - 1 - 2k) = -k$$

Schritte nach vorn, also  $k$  Schritte zurückgegangen.

- Nach  $2k + 1 = 2(k + 1) - 1$  Wiederholungen haben wir also  $k + 1$  Schritte nach vorn geschafft.
- Um den Gang zu verlassen, müssen wir insgesamt

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

Schritte zurücklegen.

Bei quadratisch vielen Schritten werden wir mächtig erschöpft sein!

# Sind quadratisch viele Schritte wirklich notwendig?

- Alles auf eine Karte zu setzen, also nur in eine Richtung zu marschieren, ist Unfug.
- Aber können wir etwas mutiger sein, als immer nur einen weiteren Schritt zu wagen?
  - ▶ Zum Beispiel, Ein Schritt nach vorn, zwei zurück, vier nach vorn, acht zurück, ... ,
  - ▶ Wieviele Schritte brauchen wir diesmal?

# Die geometrische Reihe

# Die geometrische Reihe

$$\sum_{i=0}^n a^i = \frac{a^{n+1} - 1}{a - 1} \text{ falls } a \neq 1.$$

- Vollständige Induktion nach  $n$ :

- ▶ **INDUKTIONSANFANG** für  $n = 0$ :  $\sum_{i=0}^0 a^i = 1$  und  $\frac{a^{0+1}-1}{a-1} = 1$ . ✓

- ▶ **INDUKTIONSSCHRITT** von  $n$  auf  $n + 1$ . Sei  $n \in \mathbb{N}$  beliebig.

- ★ Wir können die Induktionsannahme  $\sum_{i=0}^n a^i = \frac{a^{n+1}-1}{a-1}$  voraussetzen und müssen zeigen  $\sum_{i=0}^{n+1} a^i = \frac{a^{n+2}-1}{a-1}$ . Dann ist

- ★  $\sum_{i=0}^{n+1} a^i = \sum_{i=0}^n a^i + a^{n+1} \stackrel{IA}{=} \frac{a^{n+1}-1}{a-1} + a^{n+1} = \frac{a^{n+1}-1+a^{n+2}-a^{n+1}}{a-1} = \frac{a^{n+2}-1}{a-1}$  ✓

- Ein direkter Beweis:

$$\begin{aligned} (a-1) \cdot \sum_{i=0}^n a^i &= a \cdot \sum_{i=0}^n a^i - \sum_{i=0}^n a^i \\ &= \sum_{i=1}^{n+1} a^i - \sum_{i=0}^n a^i = a^{n+1} - a^0 = a^{n+1} - 1 \end{aligned}$$

und das war zu zeigen. ✓

# Und wie viele Schritte diesmal?

Der stockdunkle Gang: Vorher quadratisch viele Schritte, jetzt linear viele!

Warum?

- Es gelte  $2^{k-1} < n \leq 2^k$ .
- Nach höchstens

$$(1 + 2) + \cdots + (2^k + 2^{k+1}) + 2^{k+2} = 2^{k+3} - 1 \leq 16 \cdot n - 1$$

Schritten haben wir eine Distanz von

$$(1 - 2) + \cdots + (2^k - 2^{k+1}) + 2^{k+2} \geq -2^{k+1} + 2^{k+2} = 2^{k+1} > n$$

in der richtigen Richtung zurückgelegt und den rettenden Ausgang erreicht.

# Rekursiv definierte Funktionen

Wir können das Induktionsprinzip auch benutzen, um Funktionen

$$f: \mathbb{N} \rightarrow M$$

für eine beliebige Menge  $M$  zu definieren. Die Zahlen  $k, n_0 \in \mathbb{N}$  seien beliebig.

- (1) **REKURSIONSANFANG:** Definiere  $f(n_0), f(n_0 + 1), \dots, f(n_0 + k)$ .
- (2) **REKURSIONSSCHRITT:** Definiere  $f(n + 1)$  für  $n \geq n_0 + k$  unter Verwendung der Werte  $f(n_0), f(n_0 + 1), \dots, f(n)$ .

Um eine Aussage  $A$  über die Funktion  $f$  herzuleiten, benutze eine „entsprechende“ Variante der vollständigen Induktion:

- (1) **INDUKTIONSANFANG** für  $n_0, \dots, n_0 + k$ :  
Zeige, dass die Aussagen  $A(n_0), A(n_0 + 1), \dots, A(n_0 + k)$  wahr sind.
- (2) **INDUKTIONSSCHRITT** von  $n$  auf  $n + 1$ : Zeige, dass  $A(n + 1)$  wahr ist, falls die Aussagen  $A(n_0), A(n_0 + 1), \dots, A(n)$  wahr sind.



Der Brahmane Sissa ibn Dahir lebte angeblich im dritten oder vierten Jahrhundert n. Chr. in Indien.

- Der indische Herrscher Shihram tyrannisierte damals seine Untertanen und stürzte sein Land in Not und Elend.
- Sissa erfand das Schachspiel (bzw. seine indische Urform Tschaturanga), um die Aufmerksamkeit von Shihram auf seine Fehler zu lenken, ohne ihn zu erzürnen:

*Der König als wichtigste Figur kann ohne Hilfe der anderen Figuren nichts ausrichten.*

- Als Dank für die anschauliche, aber zugleich unterhaltsame Lehre gewährte Shihram dem Brahmanen einen freien Wunsch.

Sissa wünschte sich Weizenkörner:

- Auf das erste Feld eines Schachbretts wollte er ein Korn,
- auf das zweite Feld das doppelte, also zwei,
- auf das dritte wiederum die doppelte Menge, also vier und **so weiter**.

Shihram lachte und war gleichzeitig erbost über die vermeintliche Bescheidenheit des Brahmanen.

- Als sich Shihram einige Tage später erkundigte, ob Sissa seine Belohnung in Empfang genommen habe, hatten die Rechenmeister die Menge der Weizenkörner noch nicht berechnet.
- Der Vorsteher der Kornkammer meldete nach mehreren Tagen ununterbrochener Arbeit, dass er diese Menge Getreidekörner im ganzen Reich nicht aufbringen könne.
  - ▶ Auf allen Feldern eines Schachbretts zusammen wären es 18.446.744.073.709.551.615 ( $\approx 18,45$  Trillionen) Weizenkörner.
- Wie sollte Shihram das Versprechen einlösen?
  - ▶ Der Rechenmeister half dem Herrscher aus der Verlegenheit: Er empfahl, Sissa ibn Dahir solle das Getreide Korn für Korn zählen!

Wir definieren die Funktion  $f : \mathbb{N}_{\geq 1} \rightarrow \mathbb{N}$  durch

$$f(n) = \text{Anzahl der Weizenkörner auf dem } n\text{'ten Feld}$$

- Wir geben eine rekursive Definition von  $f$ .
  - ▶ **REKURSIONSANFANG:** Es ist  $f(1) = 1$  und
  - ▶ **REKURSIONSSCHRITT:**  $f(n+1) = 2 \cdot f(n)$ .
- Wir können einen einfachen Ausdruck für  $f(n)$  finden:
  - ▶ Die Anzahl der Weizenkörner am  $n$ 'ten Tag erhalten wir durch  $(n-1)$ -maliges Verdoppeln.
  - ▶ Es „sollte“  $f(n) = 2^{n-1}$  gelten!
- Wir verifizieren „ $f(n) = 2^{n-1}$ “ durch vollständige Induktion.
  - ▶ **INDUKTIONSANFANG:**  $f(1) = 1$  und  $2^{1-1} = 1$ . ✓
  - ▶ **INDUKTIONSSCHRITT** von  $n$  auf  $n+1$ . Sei  $n \in \mathbb{N}$  beliebig.  
Induktionsannahme:  $f(n) = 2^{n-1}$ , zeige  $f(n+1) = 2^n$ .  
 $f(n+1) \stackrel{\text{Rekursionsschritt}}{=} 2 \cdot f(n) \stackrel{\text{Induktionsannahme}}{=} 2 \cdot 2^{n-1} = 2^n$ . ✓

Die Anzahl der Sissa zustehenden Weizenkörner ist  $1 + 2 + 4 + \dots + 2^{63} = 2^{64} - 1$ .

- (a) **REKURSIONSANFANG**: Die boolesche Funktion  $p_1 : \{0, 1\} \rightarrow \{0, 1\}$  wird definiert durch  $p_1(x_1) := x_1$ .
- (b) **REKURSIONSSCHRITT**: Definiere die boolesche Funktion  $p_{n+1} : \{0, 1\}^{n+1} \rightarrow \{0, 1\}$  durch  $p_{n+1}(x_1, \dots, x_{n+1}) := p_n(x_1, \dots, x_n) \oplus x_{n+1}$ .

Für alle  $n \in \mathbb{N}_{>0}$  und alle  $x = (x_1, \dots, x_n) \in \{0, 1\}^n$  gilt

$$p_n(x) = 1 \iff x \text{ hat ungerade viele Einsen.}$$

Die Parität wird in fehler-korrigierenden Codes eingesetzt, weil das „Flippen“ irgendeines Bits die Parität ändert.

Sei  $x = (x_1, \dots, x_n) \in \{0, 1\}^n$  beliebig.

**INDUKTIONSANFANG** für  $n = 1$ : Es gilt  $p_1(x) = 1$  genau dann, wenn  $x_1 = 1$ , d.h. genau dann wenn  $x = (x_1)$  eine ungerade Anzahl von Einsen hat. ✓

**INDUKTIONSSCHRITT** von  $n$  nach  $n+1$ : Sei  $n \in \mathbb{N}_{>0}$  beliebig. Zeige

$$p_{n+1}(x) = 1 \iff x \text{ hat ungerade viele Einsen}$$

für alle Tupel  $x \in \{0, 1\}^{n+1}$ .

- Nach Induktionsannahme gilt für alle Tupel  $(y_1, \dots, y_n) \in \{0, 1\}^n$

$$p_n(y_1, \dots, y_n) = 1 \iff (y_1, \dots, y_n) \text{ hat ungerade viele Einsen.}$$

Wann gilt  $p_{n+1}(x) = 1$ ?

- Es ist  $p_{n+1}(x_1, \dots, x_{n+1}) := p_n(x_1, \dots, x_n) \oplus x_{n+1}$ . Also folgt

$$p_{n+1}(x) = 1 \iff \begin{aligned} & (p_n(x_1, \dots, x_n) = 1 \text{ und } x_{n+1} = 0) \text{ oder} \\ & (p_n(x_1, \dots, x_n) = 0 \text{ und } x_{n+1} = 1) \end{aligned}$$

$$\stackrel{\text{Ind.annahme}}{\iff} \begin{aligned} & (x_1, \dots, x_n) \text{ hat ungerade viele Einsen und } x_{n+1} = 0 \\ & \text{oder} \end{aligned}$$

$$\begin{aligned} & (x_1, \dots, x_n) \text{ hat gerade viele Einsen und } x_{n+1} = 1 \\ & \iff x = (x_1, x_2, \dots, x_{n+1}) \text{ hat ungerade viele Einsen} \end{aligned}$$

und das war zu zeigen. ✓

Ein Bauer züchtet Kaninchen. Jedes weibliche Kaninchen bringt im Alter von zwei Monaten ein weibliches Kaninchen zur Welt und danach jeden Monat ein weiteres.

Wie groß ist die Anzahl

$\text{fib}(n)$

der weiblichen Kaninchen, die der Bauer am Ende des  $n$ -ten Monats hat, wenn er mit einem neu geborenen weiblichen Kaninchen im ersten Monat startet?

*Antwort:*  $\text{fib}(n)$  ist rekursiv wie folgt definiert:

- **REKURSIONSANFANG:**  $\text{fib}(1) := 1$  und  $\text{fib}(2) := 1$ ,
- **REKURSIONSSCHRITT:**  $\text{fib}(n+1) := \text{fib}(n) + \text{fib}(n-1)$  f.a.  $n \in \mathbb{N}$  mit  $n \geq 2$ .

Die Funktion  $\text{fib}$  wird auch **Fibonacci-Folge**<sup>a</sup> genannt, die Zahl  $\text{fib}(n)$  heißt auch  $n$ -te Fibonacci-Zahl.

---

<sup>a</sup>Zu Ehren von Leonardo Fibonacci (13. Jh.), einem italienischen Mathematiker

Zur Erinnerung: Es ist

- $\text{fib}(1) := 1$  und  $\text{fib}(2) := 1$  und
- $\text{fib}(n+1) := \text{fib}(n) + \text{fib}(n-1)$  f.a.  $n \in \mathbb{N}$  mit  $n \geq 2$ .

Somit gilt:

$n$	1	2	3	4	5	6	7	8	9	10	11	12
$\text{fib}(n)$	1	1	2	3	5	8	13	21	34	55	89	144

Wie stark wächst die Folge  $\text{fib}(n)$ ?

- Zeige  $\text{fib}(n) \leq 2^n$  mit Induktionsanfang für  $n_0 = 1$  und  $n_0 + 1$  und dem Induktionsschritt von  $n-1$  und  $n$  auf  $n+1$ .
- Zeige, dass

$$2^{n/2} \leq \text{fib}(n)$$

für  $n \geq 6$  gilt. Verwende diesmal den Induktionsanfang für  $n_0 = 6$  und  $n_0 + 1$ .

# Rekursive Programme



# Wir berechnen $\text{fib}(n)$ , die $n$ te Fibonacci-Zahl

$\text{Algo}(n)$ :

1. Falls  $n = 1$ , dann "return"  $\text{Algo}(1) := 1$ .
2. Falls  $n = 2$ , dann "return"  $\text{Algo}(2) := 1$ .
3. Falls  $n \geq 3$ , dann "return"  $\text{Algo}(n) := \text{Algo}(n-1) + \text{Algo}(n-2)$ .

Wir zählen jede Addition, jeden Vergleich, und jede Ergebnis-Rückgabe als einen Schritt. Wir beschreiben die Anzahl  $g(n)$  benötigter Schritte durch

$$\begin{aligned}g(1) &= 2, & g(2) &= 3 \quad \text{und} \\g(n) &= 5 + g(n-1) + g(n-2).\end{aligned}$$

**Um Himmels willen**, für  $n \geq 6$  ist

$$g(n) \geq \text{fib}(n) \geq 2^{n/2},$$

die Laufzeit ist **exponentiell** ⚡ **Das geht doch viel, viel schneller!**

# Der Euklidische Algorithmus

```
def euklid(a,b):                                     # Es gelte  $a, b \in \mathbb{N}$  mit  $a \geq b$ .
    if b == 0:
        return a
    else:
        return euklid(b, a % b)
```

Zeige durch vollständige Induktion nach  $b$ , dass *für alle* natürlichen Zahlen  $a$  gilt  
$$\text{euklid}(a, b) = \text{ggT}(a, b).$$

**INDUKTIONSANFANG** für  $b = 0$ :  $\text{euklid}(a, 0) = a = \text{ggT}(a, 0)$ . ✓

**INDUKTIONSSCHRITT** für  $b > 0$ : Nach Induktionsannahme gilt  
 $\text{euklid}(a^*, b^*) = \text{ggT}(a^*, b^*)$  für alle  $a^*, b^* \in \mathbb{N}$  falls  $b^* < b$ .

Zeige  $\text{euklid}(a, b) = \text{ggT}(a, b)$ . Es ist

$$\text{ggT}(a, b) = \text{ggT}(b, a \% b)$$

Siehe Tafel.  $\implies$  Setze  $a^* := b, b^* := a \% b$ .

Da  $b^* < b$ :  $\text{euklid}(a, b) = \text{euklid}(b, a \% b) \stackrel{\text{IA}}{=} \text{ggT}(b, a \% b) = \text{ggT}(a, b)$  ✓

## 1 Mit Quantoren arbeiten:

- ▶ Zeige eine Existenz-Aussage

Es gibt ein Objekt mit Eigenschaft  $E$

z. B. durch die **Konstruktion** eines Objekts mit der geforderten Eigenschaft.

- ▶ Widerlege eine All-Aussage

Alle Objekte haben die Eigenschaft  $E$

durch ein **Gegenbeispiel**. Um die All-Aussage zu beweisen, zeige für ein *beliebiges* Objekt  $x$ , dass  $x$  die Eigenschaft  $E$  hat.

## 2 Beweismethoden:

- ▶ **Direkter** Beweis

$$\star |\mathcal{P}(M)| = 2^{|M|}, \quad \frac{a+b}{2} \geq \sqrt{a \cdot b},$$

- ▶ Beweis durch **Kontraposition**,

- ▶ Beweis durch **Widerspruch**

$\star \sqrt{2}$  ist irrational, es gibt unendlich viele Primzahlen, die Menge der reellen Zahlen ist überabzählbar groß,

- ▶ Beweis durch **vollständige Induktion**

$\star$  die Summe der ersten  $n$  Zahlen, die geometrische Reihe, die Parität, Rekursionsgleichungen, die Verifikation rekursiver Programme.

$\star$  **Rekursive Definitionen** (und ihre Analyse mit Hilfe der vollständigen Induktion).

Sei  $a$  eine positive reelle Zahl. Dann gilt  $a^n = 1$  für alle natürlichen Zahlen  $n$ .

Wir führen eine vollständige Induktion nach  $n$  aus.

**INDUKTIONSANFANG:**  $n = 0$ . Nach Definition der Potenzierung ist  $a^0 = 1$ . ✓.

**INDUKTIONSSCHRITT:**  $n \rightarrow n + 1$ . Sei  $n \in \mathbb{N}$  beliebig.

Induktionsannahme: Es gilt  $a^{n-1} = 1$  und  $a^n = 1$ . Zeige  $a^{n+1} = 1$ .

Beachte  $a^{n+1} = \frac{a^{2n}}{a^{n-1}} = \frac{a^n \cdot a^n}{a^{n-1}} = \frac{1 \cdot 1}{1} = 1$  und das war zu zeigen. ✓

Für alle natürlichen Zahlen  $a$  und  $b$  gilt  $a = b$ .

Setze  $k = \max\{a, b\}$ . Wir führen eine vollständige Induktion nach  $k$  aus.

**INDUKTIONSANFANG:** Es ist  $k = \max\{a, b\} = 0$  und  $a = 0 = b$  folgt. ✓.

**INDUKTIONSSCHRITT:**  $k \rightarrow k + 1$ . Sei  $k \in \mathbb{N}$  beliebig.

Induktionsannahme: Wenn  $\max\{c, d\} = k$ , dann folgt  $c = d$ .

Es gelte also  $\max\{a, b\} := k + 1$ : Zeige  $a = b$ .

- Da  $\max\{a, b\} = k + 1$ , folgt  $\max\{a - 1, b - 1\} = k$ .
- $\xrightarrow{\text{Induktionsannahme}} a - 1 = b - 1 \implies a = b \checkmark$

## Unfug: Teil 3 (Induktionsanfang)

F.a.  $n \in \mathbb{N}$  mit  $n \geq 1$  gilt: Ist  $M$  eine Menge von Menschen mit  $|M| = n$ , so haben alle Menschen in  $M$  die gleiche Größe.

Wir führen einen „Beweis“ durch Induktion nach  $n$ :

INDUKTIONSANFANG:  $n = 1$

*Behauptung:*

Ist  $M$  eine Menge von Menschen mit  $|M| = 1$ , so haben alle Menschen in  $M$  die gleiche Größe.

*Beweis:*

Sei  $M$  eine Menge von Menschen mit  $|M| = 1$ . D.h.  $M$  besteht aus genau einem Menschen. Daher haben offensichtlich alle Menschen in  $M$  die gleiche Größe.

# Unfug: Teil 3 (Induktionsschritt)

INDUKTIONSSCHRITT:  $n \rightarrow n + 1$

Sei  $n \in \mathbb{N}$  mit  $n \geq 1$  beliebig.

*Induktionsannahme:* Ist  $M'$  eine Menge von Menschen mit  $|M'| = n$ , so haben alle Menschen in  $M'$  die gleiche Größe.

*Behauptung:* Ist  $M$  eine Menge von Menschen mit  $|M| = n + 1$ , so haben alle Menschen in  $M$  die gleiche Größe.

*Beweis:* Sei  $M = \{a_1, a_2, \dots, a_n, a_{n+1}\}$  eine Menge von  $n + 1$  Menschen mit den Personen  $a_1, a_2, \dots, a_n, a_{n+1}$ . Wir können die Induktionsannahme auf

$$M' := \{a_1, a_2, \dots, a_n\} \quad \text{und} \quad M'' := \{a_2, \dots, a_n, a_{n+1}\}.$$

anwenden, denn  $M'$  und  $M''$  sind Mengen von  $n$  Menschen. Wir erhalten

- (1) Alle Menschen in  $M'$  haben die gleiche Größe  $g'$ , und
- (2) alle Menschen in  $M''$  haben die gleiche Größe  $g''$ .

Aber Person  $a_2$  gehört sowohl zu  $M'$  wie auch zu  $M''$ : Also folgt

$$g' = g''. \quad \square$$

# Ein unkonventioneller Induktionsbeweis

Gegeben sei die folgende rekursive Funktion  $f : \mathbb{Z} \rightarrow \mathbb{Z}$ .

```
def f(z):  
    if z > 11:  
        return z-2  
    else:  
        return f(f(z+3))
```

Zeigen Sie mit vollständiger Induktion:

*Für alle  $z \in \mathbb{Z}$  mit  $z \leq 12$  gibt die Funktion  $f$  den Wert 10 aus.*

*Hinweis:* Verwenden Sie als Induktionsschritt  $z \rightarrow z - 3$ . Was müssen Sie dann beim Induktionsanfang beachten?

*Kommentar:* Bei der obigen Funktion handelt es sich um eine Variante der *McCarthy-91-Funktion*, die als schwieriger Testfall bei der formalen Verifikation von Programmen eingesetzt wird.