Grundbegriffe der Informatik

Einheit 6: Der Begriff des Algorithmus (erste grundlegende Aspekte)

Prof. Dr. Tanja Schultz

Karlsruher Institut für Technologie, Fakultät für Informatik

Wintersemester 2011/2012

Überblick

Eine kleine Zeitreise

Algorithmusbegriff

- Lösen einer Sorte quadratischer Gleichungen
- Zum informellen Algorithmusbegriff
- Korrektheit des Algorithmus zur Lösung von $x^2 + bx = c$
- Wie geht es weiter?
- Algorithmus zur Multiplikation nichtnegativer ganzer Zahlen
- Multiplikationsalgorithmus mit einer Schleife

bitte alle einsteigen

Türen schließen

anschnallen

Sitzlehnen aufrecht stellen

Schwätzen einstellen

und so weiter und so fort

Wie weit in die Vergangenheit kann man reisen und findet noch etwas, was mit Informatik zu tun hat? (jenseits von Zählen und Zahlen) bitte alle einsteigen

Türen schließen

anschnallen

Sitzlehnen aufrecht stellen

Schwätzen einstellen

und so weiter und so fort

Wie weit in die Vergangenheit kann man reisen und findet noch etwas, was mit Informatik zu tun hat? (jenseits von Zählen und Zahlen)

- Zeit: circa 825–830 n.Chr.
- Ort: Bagdad, Haus der Weisheit
- wir treffen ...

- Muhammad ibn Mūsā al-Khwārizmī
- geboren ca. 780 n.Chr.
  in Chiva (heute Usbekistan)
  oder Qutrubbull (heute Iran)
- gestorben ca. 850 n.Chr.
- Haus der Weisheit, Wirken vermutlich von 813–830 n.Chr.
- Astronom, Mathematiker und Geograph
- Beiname Abu Abdullah

Bildquelle: `http://en.wikipedia.org/wiki/Image:Abu_Abdullah_Muhammad_bin_Musa_al-Khwarizmi.jpg`

. . . da wären wir. . .

- Zeit: circa 825–830 n.Chr.
- Ort: Bagdad, Haus der Weisheit
- wir treffen

- Muhammad ibn Mūsā al-Khwārizmī
- geboren ca. 780 n.Chr.
  in Chiva (heute Usbekistan)
  oder Qutrubbull (heute Iran)
- gestorben ca. 850 n.Chr.

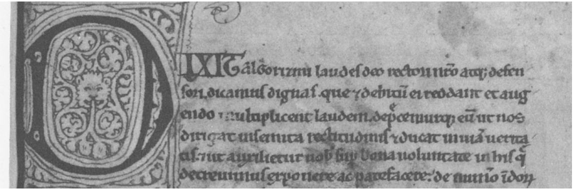
- Haus der Weisheit, Wirken vermutlich von 813–830 n.Chr.
- Astronom, Mathematiker und Geograph
- Beiname Abu Abdullah

Bildquelle: `http://en.wikipedia.org/wiki/Image:Abu_Abdullah_Muhammad_bin_Musa_al-Khwarizmi.jpg`

Zwei wichtige Schriften von al-Khwārizmī (1)

- ca. 830 (?): Buch
  - Titel:
    „Al-Kitāb al-mukhtaṣar fī hīsāb al-ğabr wa'l-muqābala" oder
    „Al-Kitāb al-mukhtaṣar fī ḥisāb al-jabr wa-l-muqābala".
  - auf Deutsch:
    „Das kurzgefasste Buch zum Rechnen durch Ergänzung und Ausgleich"
  - Aus „al-ğabr" bzw. „al-jabr" entstand später das Wort Algebra.
  - Inhalt des Buches unter anderem: Lösen quadratischer Gleichungen mit einer Unbekannten.

- ca. 825 (??)
  - Titel vielleicht „Kitāb al-Jamʿ wa-l-tafrīq bi-ḥisāb al-Hind“
  - auf Deutsch: “Über das Rechnen mit indischen Ziffern“.
  - al-Khwārizmī führt u. a. die aus dem Indischen stammende Zahl Null in das arabische Zahlensystem ein . . . Null [sifr] $\rightarrow$ „Ziffer“
  - nur noch Übersetzungen, z. B. auf Lateinisch, 12. Jhdt. (?):



  - kein Titel bekannt, Vermutung:
    - „Algoritmi de numero Indorum“ oder
    - „Algorismi de numero Indorum“ o. ä.
  - also „Werk des Al-gorismi über die indischen Zahlen“.
  - Das „i“ (Genitiv) von „Algorismi“ wurde später fälschlicherweise als Pluralendung des Wortes Algorithmus

Überblick

Eine kleine Zeitreise

Algorithmusbegriff

- Lösen einer Sorte quadratischer Gleichungen
- Zum informellen Algorithmusbegriff
- Korrektheit des Algorithmus zur Lösung von $x^2 + bx = c$
- Wie geht es weiter?
- Algorithmus zur Multiplikation nichtnegativer ganzer Zahlen
- Multiplikationsalgorithmus mit einer Schleife

Überblick

Lösen einer Sorte quadratischer Gleichungen nach al-Khwārizmī (1)

- gegeben: quadratische Gleichung der Form

$$x^2 + bx = c \qquad \text{mit } b > 0 \text{ und } c > 0$$

- Dann kann man laut al-Khwārizmī die positive Lösung dieser Gleichung bestimmen, indem man nacheinander rechnet:

$$h \leftarrow b/2 \tag{1}$$
$$q \leftarrow h^2 \tag{2}$$
$$s \leftarrow c + q \tag{3}$$
$$w \leftarrow \sqrt{s} \tag{4}$$
$$x \leftarrow w - h \tag{5}$$

Lösen einer Sorte quadratischer Gleichungen nach al-Khwārizmī (2)

- Rechnung:

$$h \leftarrow b/2 \tag{1}$$
$$q \leftarrow h^2 \tag{2}$$
$$s \leftarrow c + q \tag{3}$$
$$w \leftarrow \sqrt{s} \tag{4}$$
$$x \leftarrow w - h \tag{5}$$

- wir schreiben hier *Zuweisungen* in der Form

$$\langle \textit{Variablenname} \rangle \leftarrow \langle \textit{arithmetischer Ausdruck} \rangle$$

- Zuweisungen alle „ausführbar“, da rechts nur Eingaben b und c benutzt und Variablen, die schon einen Wert haben.
- keine Unglücke: s ist nie negativ.
- am Ende hat x immer einen Wert, der die quadratische Gleichung $x^2 + bx = c$ erfüllt.

Überblick

Algorithmusbegriff informell

Eigenschaften des eben gezeigten Algorithmus:

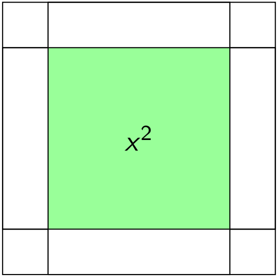
- Algorithmus besitzt endliche Beschreibung
  (ist also ein Wort über einem Alphabet).
- Beschreibung besteht aus elementaren Anweisungen;
  jede offensichtlich effektiv in einem Schritt ausführbar
- Determinismus: nächste elementare Anweisung stets eindeutig festgelegt, nur auf Grund von
  - schon berechneten Ergebnissen und
  - zuletzt ausgeführter elementare Anweisung
- Aus endlicher Eingabe wird endliche Ausgabe berechnet.
- Dabei werden endliche viele Schritte gemacht,
  d. h. nur endlich oft eine elementare Anweisung ausgeführt.
- Der Algorithmus funktioniert für beliebig große Eingaben.
- Die Nachvollziehbarkeit/Verständlichkeit des Algorithmus steht für jeden (mit der Materie vertrauten) außer Frage.

- obige Forderungen sind plausibel aber informell:
  - Was soll z. B. „offensichtlich effektiv ausführbar“ heißen?
  - Für harte Beweise benötigt man einen präziseren Algorithmusbegriff.
- Es hat sich herausgestellt, dass auch Verallgemeinerungen des oben skizzierten Algorithmenbegriffes interessant sind. Dazu gehören zum Beispiel:
  - randomisierte Algorithmen:
    Zufallsereignisse haben Einfluss auf die Fortsetzung eines Algorithmus
  - Online-Algorithmen:
    die Eingaben stehen nicht alle zu Beginn zur Verfügung, sondern erst nach und nach, und
  - nicht terminierende sondern unendlich lange laufende Algorithmen (z. B. Ampelsteuerung)
  - usw. ...

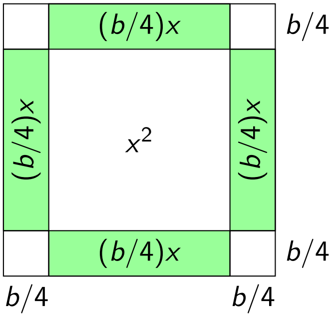
Diskussion des informellen Algorithmusbegriffs

- obige Forderungen sind plausibel aber informell:
  - Was soll z. B. „offensichtlich effektiv ausführbar“ heißen?
  - Für harte Beweise benötigt man einen präziseren Algorithmusbegriff.
- Es hat sich herausgestellt, dass auch Verallgemeinerungen des oben skizzierten Algorithmenbegriffes interessant sind. Dazu gehören zum Beispiel:
  - randomisierte Algorithmen:
    Zufallsereignisse haben Einfluss auf die Fortsetzung eines Algorithmus
  - Online-Algorithmen:
    die Eingaben stehen nicht alle zu Beginn zur Verfügung, sondern erst nach und nach, und
  - nicht terminierende sondern unendlich lange laufende Algorithmen (z. B. Ampelsteuerung)
  - usw. ...

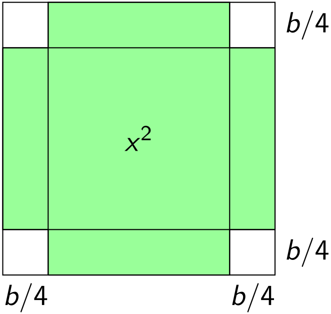
Beweis von al-Khwārizmī



Beweis von al-Khwārizmī

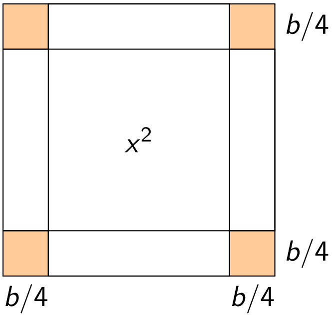


Beweis von al-Khwārizmī



$$x^2 + 4(b/4 \cdot x) = x^2 + bx = c$$

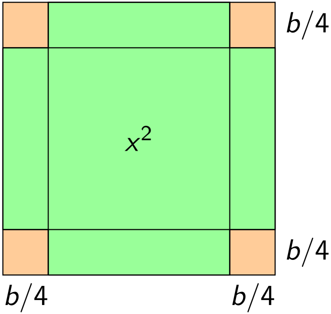
Beweis von al-Khwārizmī



$$x^2 + bx = c$$

$$4 \cdot (b/4)^2 = 4 \cdot b^2/16 = b^2/4 = q$$

Beweis von al-Khwārizmī



$$x^2 + bx = c$$

$$4 \cdot b^2/16 = b^2/4 = q$$

$$\boxed{c + q = (b/4 + x + b/4)^2 = (x + b/2)^2}$$

$$\boxed{\sqrt{c+q} - b/2 = x}$$

Beweis durch Nachrechnen

$/\!/ \quad b > 0 \land c > 0$

$h \leftarrow b/2$

$/\!/ \quad h = b/2$

$q \leftarrow h^2$

$/\!/ \quad q = b^2/4$

$s \leftarrow c + q$

$/\!/ \quad s = c + b^2/4$

$w \leftarrow \sqrt{s}$

$/\!/ \quad w = \sqrt{c + b^2/4}$

$x \leftarrow w - h$

$/\!/ \quad x = \sqrt{c + b^2/4} - b/2$

$/\!/ \quad x^2 + bx = (\sqrt{c + b^2/4} - b/2)^2 + b(\sqrt{c + b^2/4} - b/2)$

$/\!/ \quad x^2 + bx = c + b^2/4 - b\sqrt{c + b^2/4} + b^2/4 + b\sqrt{c + b^2/4} - b^2/2$

$/\!/ \quad x^2 + bx = c$

Beweis durch Nachrechnen

$/\!/ \quad b > 0 \wedge c > 0$

$h \leftarrow b/2$

$/\!/ \quad h = b/2$

$q \leftarrow h^2$

$/\!/ \quad q = b^2/4 \wedge h = b/2$

$s \leftarrow c + q$

$/\!/ \quad s = c + b^2/4 \wedge h = b/2$

$w \leftarrow \sqrt{s}$

$/\!/ \quad w = \sqrt{c + b^2/4} \wedge h = b/2$

$x \leftarrow w - h$

$/\!/ \quad x = \sqrt{c + b^2/4} - b/2$

$/\!/ \quad x^2 + bx = (\sqrt{c + b^2/4} - b/2)^2 + b(\sqrt{c + b^2/4} - b/2)$

$/\!/ \quad x^2 + bx = c + b^2/4 - b\sqrt{c + b^2/4} + b^2/4 + b\sqrt{c + b^2/4} - b^2/2$

$/\!/ \quad x^2 + bx = c$

Überblick

Wie geht es weiter ... in dieser Vorlesung und in anderen ...?

- in dieser Vorlesung nur andeutungsweise z. B.: Wie kann man sich *allgemein* von der Richtigkeit solcher Folgen von Rechnungen überzeugen?
  - Ansätze für die *Verifikation* von Algorithmen
- Dafür braucht man aber
  - einen präzisen Algorithmenbegriff,
  - eine präzise Spezifikation von „Verhalten" eines Algorithmus
    - Schlagwort *Semantik*
  - eine präzise Spezifikation von „was ist richtig"
  - präzise Methoden, um z. B. zu beweisen, dass das Verhalten eines Algorithmus der Spezifikation genügt.
  - Dazu kommt in allen Fällen auch eine präzise Notation, die zumindest bei der Verarbeitung durch Rechner nötig ist.

Wie geht es weiter . . . ? (2)

Weitere Punkte, die zum Teil schon in dieser Vorlesung:

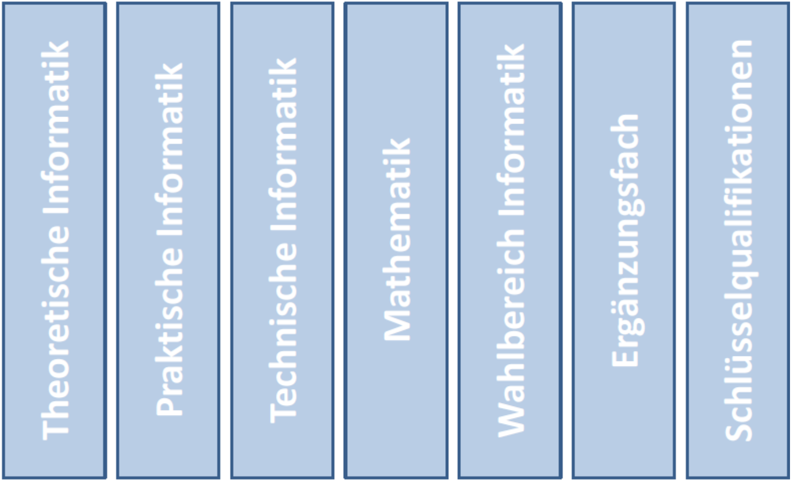
- Präzisierungen des Algorithmenbegriffes:
  - Sie kennen inzwischen z. B. Grundzüge einer Programmiersprache.
    - praktisch, wenn man tatsächlich Algorithmen so aufschreiben will, dass sie ein Rechner ausführen können soll.
    - unpraktisch, wenn man z. B. beweisen will, dass ein bestimmtes Problem durch keinen Algorithmus gelöst werden kann.
  - einfachere Modelle wie Registermaschinen oder Turingmaschinen

Wie geht es weiter . . . ? (3)

Weitere Punkte, die Sie später sehen werden:

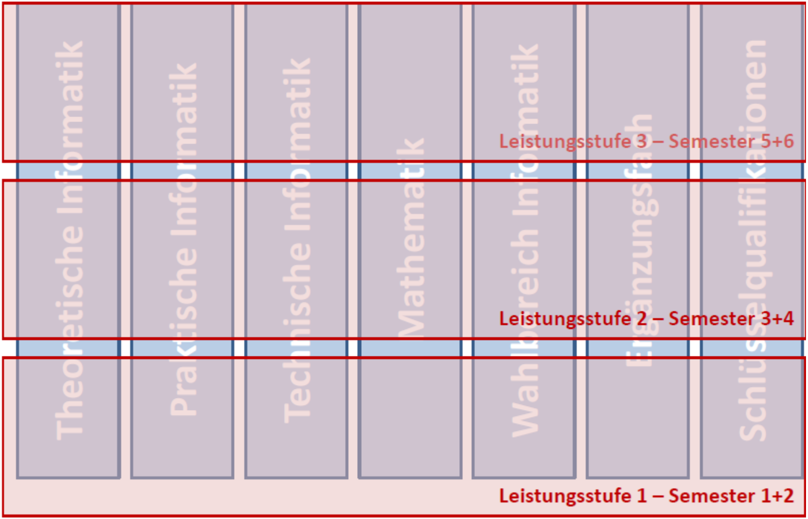
- präzise Notationen für „das richtige Verhalten“
  - *Zusicherungen*: logische Formeln, die Aussagen über (Zusammenhänge zwischen) Variablen machen
- präzise Methoden, um zu beweisen, dass ein Algorithmus „das Richtige tut“
  - Schlagworte *schwächste Vorbedingung*, *Schleifeninvarianten*
- präzise Notationsmöglichkeiten, um Aufgaben dem Rechner übertragen zu können
  - Wie legt man fest, was syntaktisch korrekt ist?
  - Wie stellt man fest, ob etwas syntaktisch korrekt ist?
  - Schlagwort *formale Sprachen*

Modulhandbuch Informatik B.Sc. (WS 2011/2012)
www.informatik.kit.edu/downloads/studium/mhb_ws11_12_info_bsc_de_lang.pdf



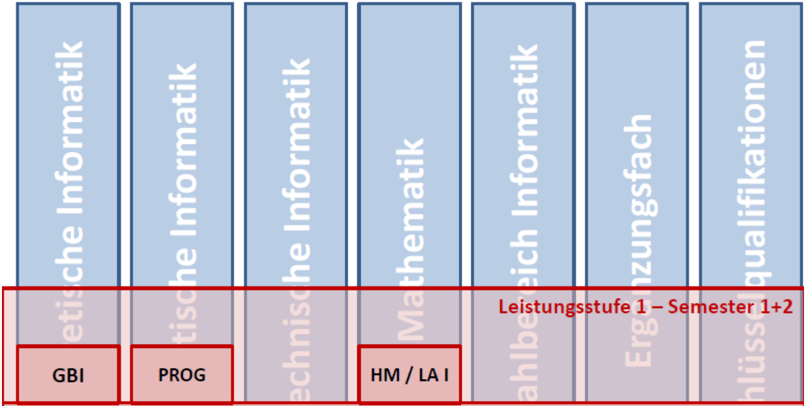
Modulhandbuch Informatik B.Sc. (2)

Modulhandbuch Informatik B.Sc. (WS 2011/2012)
www.informatik.kit.edu/downloads/studium/mhb_ws11_12_info_bsc_de_lang.pdf



Modulhandbuch Informatik B.Sc. (WS 2011/2012)

www.informatik.kit.edu/downloads/studium/mhb_ws11_12_info_bsc_de_lang.pdf

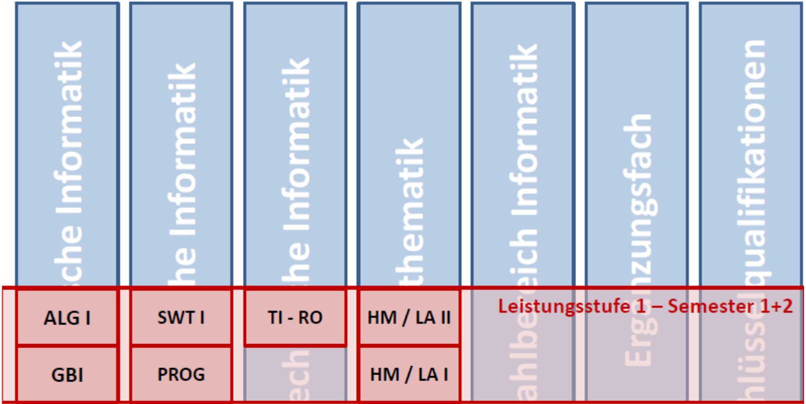


Orientierungsprüfung (muss bis Ende 3.Sem bestanden sein): Kontrolle der für das weitere Studium relevanten Grundkenntnisse

1. Grundbegriffe der Informatik (GBI)
2. Programmieren (PROG) – Java Programmiersprache, Prof. Pretschner
3. Höhere Mathematik oder Lineare Algebra (HM / LA – Teil I)

Modulhandbuch Informatik B.Sc. (WS 2011/2012)

www.informatik.kit.edu/downloads/studium/mhb_ws11_12_info_bsc_de_lang.pdf

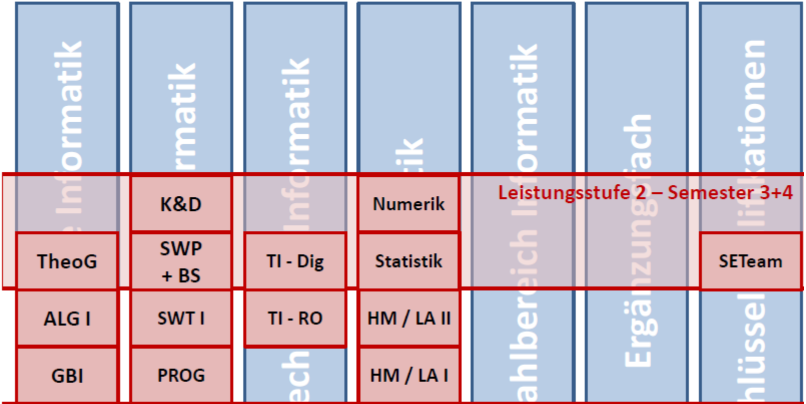


Grundbegriffe der Informatik (GBI) sind wichtige Grundlagen für:

1. **Algorithmen I** (Entwurf, Korrektheit, Effizienz, Datenstrukturen …) **Prof. Zitterbart**
2. **Softwaretechnik I** (Implementierung, Validation, Verifikation) **Prof. Tichy**

Modulhandbuch Informatik B.Sc. (WS 2011/2012)

www.informatik.kit.edu/downloads/studium/mhb_ws11_12_info_bsc_de_lang.pdf

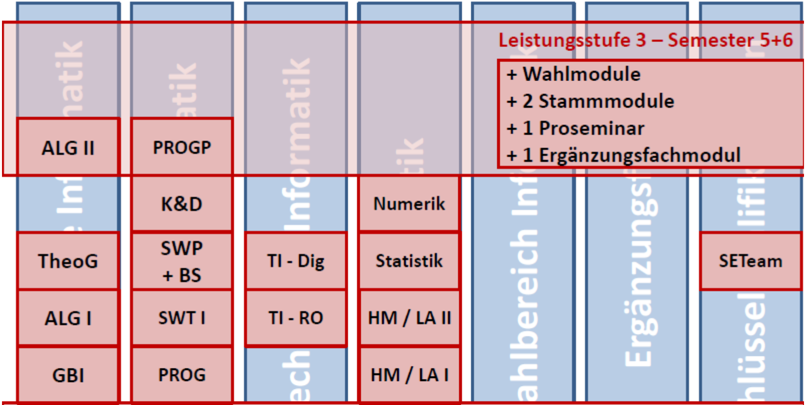


Grundbegriffe der Informatik (GBI) sind wichtige Grundlage für:

1. **Theor. Grundlagen der Informatik** (Berechenbarkeit, Lösbarkeit) **Prof. Wagner**
2. **Praxis der SWE** (Pflichtenheft, Spezifikation, Implementierung) **Prof. Snelting**
3. **Betriebssysteme** (Systemarchitekturen, Speicher ...) **Prof. Bellosa**
4. **Kommunikation und Datenhaltung** (Telekomm., Netze, DB) **Prof. Böhm/Zitterbart**

Modulhandbuch Informatik B.Sc. (6)

Modulhandbuch Informatik B.Sc. (WS 2011/2012)
www.informatik.kit.edu/downloads/studium/mhb_ws11_12_info_bsc_de_lang.pdf



Grundbegriffe der Informatik (GBI) wichtige Grundlage für:

1. **Algorithmen II** (Approximations-, Lineare Programmierung, randomisierte, parallele, und parametrisierte Algorithmen) **Profs. Wagner / Sanders**
2. **Programmierparadigmen** (rek. Fkt, Lambda-Kalkül, Log. Program. - Terme, Hornklauseln, Unifikation, Parallele Prog., Compilerbau) **Profs. Snelting / Reussner**

Mathematisches Vorgeplänkel

Definiere zwei binäre Operationen **div** und **mod** für $x, y \in \mathbb{N}_0$:

- x **mod** y
  der Rest der ganzzahligen Division von x durch y
  also stets: $0 \leq x \textbf{ mod } y < y$
  - Beispiel: $11 \textbf{ mod } 3 = 2$ (beachte: $0 \leq 2 < 3$)
- x **div** y
  das Ergebnis der ganzzahligen Division von x durch y.
  - Beipiel: $11 \textbf{ div } 3 = 3$
- Daher gilt für $x, y \in \mathbb{N}_0$ stets:

$$x = y \cdot (x \textbf{ div } y) + (x \textbf{ mod } y)$$

  - In unserem Beispiel mit $(x = 11, y = 3)$:
  - $11 = 3 \cdot (11 \textbf{ div } 3) + (11 \textbf{ mod } 3) = 3 \cdot 3 + 2 = 11$

// Eingaben: $a \in \mathbb{G}_8, b \in \mathbb{N}_0$

$P_0 \leftarrow 0$

$X_0 \leftarrow a$

$Y_0 \leftarrow b$

$x_0 \leftarrow X_0$ **mod** 2

// — Algorithmusstelle — $i = 0$

$P_1 \leftarrow P_0 + x_0 \cdot Y_0$

$X_1 \leftarrow X_0$ **div** 2

$Y_1 \leftarrow 2 \cdot Y_0$

$x_1 \leftarrow X_1$ **mod** 2

// — Algorithmusstelle — $i = 1$

// — Algorithmusstelle — $i = 1$

$P_2 \leftarrow P_1 + x_1 \cdot Y_1$

$X_2 \leftarrow X_1$ **div** 2

$Y_2 \leftarrow 2 \cdot Y_1$

$x_2 \leftarrow X_2$ **mod** 2

// — Algorithmusstelle — $i = 2$

$P_3 \leftarrow P_2 + x_2 \cdot Y_2$

$X_3 \leftarrow X_2$ **div** 2

$Y_3 \leftarrow 2 \cdot Y_2$

$x_3 \leftarrow X_3$ **mod** 2

// — Algorithmusstelle — $i = 3$

Beispielrechnung

Es sei $a = 6$ und $b = 9$

$$P_i \leftarrow P_{i-1} + x_{i-1} \cdot Y_{i-1}$$
$$X_i \leftarrow X_{i-1} \textbf{ div } 2$$
$$Y_i \leftarrow 2 \cdot Y_{i-1}$$
$$x_i \leftarrow X_i \textbf{ mod } 2$$

	P_i	X_i	Y_i	x_i
$i = 0$	0	6	9	0
$i = 1$	0	3	18	1
$i = 2$	18	1	36	1
$i = 3$	54	0	72	0

- Am Ende ist $P_3 = 54 = a \cdot b$
- Wir wollen beweisen: Das klappt immer!

Beispielrechnung

Es sei $a = 6$ und $b = 9$

$$P_i \leftarrow P_{i-1} + x_{i-1} \cdot Y_{i-1}$$
$$X_i \leftarrow X_{i-1} \textbf{ div } 2$$
$$Y_i \leftarrow 2 \cdot Y_{i-1}$$
$$x_i \leftarrow X_i \textbf{ mod } 2$$

	P_i	X_i	Y_i	x_i
$i = 0$	0	6	9	0
$i = 1$	0	3	18	1
$i = 2$	18	1	36	1
$i = 3$	54	0	72	0

- Am Ende ist $P_3 = 54 = a \cdot b$
- Wir wollen beweisen: Das klappt immer!

Annäherung an einen Korrektheitsbeweis

- P_3 wird mit Hilfe von P_2 ausgerechnet.
- Also sollte man auch etwas über P_2 wissen.
- und über P_1 auch, usw.
- Analog sollte man am besten etwas über alle X_i und Y_i wissen.

- Angenommen, es gelingt uns, „etwas Passendes“ hinzuschreiben,
- d. h. logische Formeln $\mathcal{A}_i$, die Aussagen über die interessierenden Werte P_i, x_i, X_i und Y_i machen.
- Was dann?

- vollständige Induktion

Annäherung an einen Korrektheitsbeweis

- P_3 wird mit Hilfe von P_2 ausgerechnet.
- Also sollte man auch etwas über P_2 wissen.
- und über P_1 auch, usw.
- Analog sollte man am besten etwas über alle X_i und Y_i wissen.

- Angenommen, es gelingt uns, „etwas Passendes“ hinzuschreiben,
- d. h. logische Formeln $\mathcal{A}_i$, die Aussagen über die interessierenden Werte P_i, x_i, X_i und Y_i machen.
- Was dann?

- vollständige Induktion

Annäherung an einen Korrektheitsbeweis

- P_3 wird mit Hilfe von P_2 ausgerechnet.
- Also sollte man auch etwas über P_2 wissen.
- und über P_1 auch, usw.
- Analog sollte man am besten etwas über alle X_i und Y_i wissen.

- Angenommen, es gelingt uns, „etwas Passendes“ hinzuschreiben,
- d. h. logische Formeln $\mathcal{A}_i$, die Aussagen über die interessierenden Werte P_i, x_i, X_i und Y_i machen.
- Was dann?

- vollständige Induktion

Annäherung an einen Korrektheitsbeweis (2)

- Problem:
  Induktionsbeweise sind am Anfang schon schwer genug.
- Aber wir müssen auch erst noch Aussagen $\mathcal{A}_i$ finden,
  - die wir erstens beweisen können, und
  - die uns zweitens zum gewünschten Ziel führen.
- Passende Aussagen zu finden ist nicht immer ganz einfach und Übung ist sehr(!) hilfreich.
- Hinweise durch Wertetabelle:

	P_i	X_i	Y_i	x_i
$i = 0$	0	6	9	0
$i = 1$	0	3	18	1
$i = 2$	18	1	36	1
$i = 3$	54	0	72	0

Annäherung an einen Korrektheitsbeweis (2)

- Problem:
  Induktionsbeweise sind am Anfang schon schwer genug.
- Aber wir müssen auch erst noch Aussagen $\mathcal{A}_i$ finden,
  - die wir erstens beweisen können, und
  - die uns zweitens zum gewünschten Ziel führen.
- Passende Aussagen zu finden ist nicht immer ganz einfach und Übung ist sehr(!) hilfreich.
- Hinweise durch Wertetabelle:

	P_i	X_i	Y_i	x_i
$i = 0$	0	6	9	0
$i = 1$	0	3	18	1
$i = 2$	18	1	36	1
$i = 3$	54	0	72	0

- Herumspielen liefert, dass jedenfalls im Beispiel für jedes $i \in \mathbb{G}_4$ die folgende Aussage wahr ist:

$$\forall i \in \mathbb{G}_4 : X_i \cdot Y_i + P_i = a \cdot b$$

	P_i	X_i	Y_i	x_i	$X_i \cdot Y_i + P_i$
$i = 0$	0	6	9	0	$6 \cdot 9 + 0 = 54$
$i = 1$	0	3	18	1	$3 \cdot 18 + 0 = 54$
$i = 2$	18	1	36	1	$1 \cdot 36 + 18 = 54$
$i = 3$	54	0	72	0	$0 \cdot 72 + 54 = 54$

Annäherung an einen Korrektheitsbeweis (4)

- Für jedes $i \in \mathbb{G}_4$ ist die folgende Aussage wahr ist:

$$\forall i \in \mathbb{G}_4 : X_i \cdot Y_i + P_i = a \cdot b$$

- Das formen wir noch in eine Aussage für alle nichtnegativen ganzen Zahlen um:

$$\forall i \in \mathbb{N}_0 : i < 4 \Longrightarrow X_i \cdot Y_i + P_i = a \cdot b$$

  Wir beweisen nun durch vollständige Induktion die Formel

$$\forall i \in \mathbb{N}_0 : \mathcal{A}_i$$

  wobei $\mathcal{A}_i$ die Aussage ist:

$$i < 4 \Longrightarrow X_i \cdot Y_i + P_i = a \cdot b \ .$$

Korrektheitsbeweis (1)

Induktionsanfang $i = 0$:

Aufgrund der Initialisierungen der Variablen ist klar:

$$X_0 Y_0 + P_0 = ab + 0 = ab \ .$$

Also gilt: $0 < 4 \Longrightarrow X_0 Y_0 + P_0 = ab$

Induktionsvoraussetzung:

für ein beliebiges aber festes i gelte

$$i < 4 \Longrightarrow X_i \cdot Y_i + P_i = a \cdot b$$

Induktionsschluss $i \to i + 1$: zu zeigen:

$$i + 1 < 4 \Longrightarrow X_{i+1} \cdot Y_{i+1} + P_{i+1} = a \cdot b$$

Korrektheitsbeweis (2)

Induktionsvoraussetzung:

$$i < 4 \Longrightarrow X_i \cdot Y_i + P_i = a \cdot b$$

Induktionsschluss $i \to i + 1$:

Zeige: $i + 1 < 4 \Longrightarrow X_{i+1} \cdot Y_{i+1} + P_{i+1} = a \cdot b$.

- Wenn $i + 1 < 4$, dann auch $i < 4$ und nach Ind.vor. gilt: $X_i \cdot Y_i + P_i = a \cdot b$.
- Wir rechnen nun:

$$\begin{aligned}
X_{i+1} \cdot Y_{i+1} + P_{i+1} &= (X_i \textbf{ div } 2) \cdot 2Y_i + P_i + x_i Y_i \\
&= (X_i \textbf{ div } 2) \cdot 2Y_i + P_i + (X_i \textbf{ mod } 2) Y_i \\
&= (2(X_i \textbf{ div } 2) + (X_i \textbf{ mod } 2)) Y_i + P_i \\
&= X_i Y_i + P_i \\
&= ab \; .
\end{aligned}$$

- erste beiden Gleichheiten wegen Zuweisungen im Algorithmus,
- vierte wegen Gleichung aus „mathematischem Vorgeplänkel“ ($x = y \cdot (x \textbf{ div } y) + (x \textbf{ mod } y)$, mit $X_i = x \wedge y = 2$),
- die letzte nach Induktionsvoraussetzung.

Korrektheitsbeweis (2)

Induktionsvoraussetzung:

$$i < 4 \Longrightarrow X_i \cdot Y_i + P_i = a \cdot b$$

Induktionsschluss $i \to i + 1$:

Zeige: $i + 1 < 4 \Longrightarrow X_{i+1} \cdot Y_{i+1} + P_{i+1} = a \cdot b$.

- Wenn $i + 1 < 4$, dann auch $i < 4$ und nach Ind.vor. gilt: $X_i \cdot Y_i + P_i = a \cdot b$.
- Wir rechnen nun:

$$\begin{aligned} X_{i+1} \cdot Y_{i+1} + P_{i+1} &= (X_i \textbf{ div } 2) \cdot 2Y_i + P_i + x_i Y_i \\ &= (X_i \textbf{ div } 2) \cdot 2Y_i + P_i + (X_i \textbf{ mod } 2)Y_i \\ &= (2(X_i \textbf{ div } 2) + (X_i \textbf{ mod } 2))Y_i + P_i \\ &= X_i Y_i + P_i \\ &= ab . \end{aligned}$$

- erste beiden Gleichheiten wegen Zuweisungen im Algorithmus,
- vierte wegen Gleichung aus „mathematischem Vorgeplänkel“ ($x = y \cdot (x \textbf{ div } y) + (x \textbf{ mod } y)$, mit $X_i = x \wedge y = 2$),
- die letzte nach Induktionsvoraussetzung.

Korrektheitsbeweis (2)

Induktionsvoraussetzung:

$$i < 4 \Longrightarrow X_i \cdot Y_i + P_i = a \cdot b$$

Induktionsschluss $i \to i + 1$:

Zeige: $i + 1 < 4 \Longrightarrow X_{i+1} \cdot Y_{i+1} + P_{i+1} = a \cdot b$.

- Wenn $i + 1 < 4$, dann auch $i < 4$ und nach Ind.vor. gilt: $X_i \cdot Y_i + P_i = a \cdot b$.
- Wir rechnen nun:

$$\begin{aligned}
X_{i+1} \cdot Y_{i+1} + P_{i+1} &= (X_i \textbf{ div } 2) \cdot 2Y_i + P_i + x_i Y_i \\
&= (X_i \textbf{ div } 2) \cdot 2Y_i + P_i + (X_i \textbf{ mod } 2) Y_i \\
&= (2(X_i \textbf{ div } 2) + (X_i \textbf{ mod } 2)) Y_i + P_i \\
&= X_i Y_i + P_i \\
&= ab\ .
\end{aligned}$$

- erste beiden Gleichheiten wegen Zuweisungen im Algorithmus,
- vierte wegen Gleichung aus „mathematischem Vorgeplänkel“ $(x = y \cdot (x \textbf{ div } y) + (x \textbf{ mod } y)$, mit $X_i = x \wedge y = 2)$,
- die letzte nach Induktionsvoraussetzung.

Induktionsvoraussetzung:

$$i < 4 \Longrightarrow X_i \cdot Y_i + P_i = a \cdot b$$

Induktionsschluss $i \to i+1$:

Zeige: $i+1 < 4 \Longrightarrow X_{i+1} \cdot Y_{i+1} + P_{i+1} = a \cdot b.$

- Wenn $i+1 < 4$, dann auch $i < 4$ und nach Ind.vor. gilt: $X_i \cdot Y_i + P_i = a \cdot b.$
- Wir rechnen nun:

$$\begin{aligned}
X_{i+1} \cdot Y_{i+1} + P_{i+1} &= (X_i \ \mathbf{div}\ 2) \cdot 2Y_i + P_i + x_i Y_i \\
&= (X_i \ \mathbf{div}\ 2) \cdot 2Y_i + P_i + (X_i \ \mathbf{mod}\ 2) Y_i \\
&= (2(X_i \ \mathbf{div}\ 2) + (X_i \ \mathbf{mod}\ 2)) Y_i + P_i \\
&= X_i Y_i + P_i \\
&= ab\ .
\end{aligned}$$

- erste beiden Gleichheiten wegen Zuweisungen im Algorithmus,
- vierte wegen Gleichung aus „mathematischem Vorgeplänkel“ $(x = y \cdot (x \ \mathbf{div}\ y) + (x \ \mathbf{mod}\ y)$, mit $X_i = x \wedge y = 2)$,
- die letzte nach Induktionsvoraussetzung.

- Wissen wir nun, dass am Ende des Algorithmus $P_3 = ab$ ist?
- Nein: bisher nur bewiesen, dass $P_3 + X_3 Y_3 = ab$
- Wertetabelle zeigt, dass im Beispiel aber $X_3 = 0$.
- Beweisen wir, dass auch das für *alle* Eingaben $a \in \mathbb{G}_8$ und $b \in \mathbb{N}_0$ gilt. Wie?
- Beobachtung: Die X_i werden der Reihe nach immer kleiner.
  - Und zwar immer um mindestens die Hälfte, denn X_i **div** $2 \leq X_i/2$.
  - Mit anderen Worten:

$$
\begin{aligned}
X_0 &\leq a \\
X_1 &\leq X_0/2 \leq a/2 \\
X_2 &\leq X_1/2 \leq a/4 \\
&\vdots
\end{aligned}
$$

- Wie man *die* Pünktchen weg bekommt wissen wir schon: vollständige Induktion

Korrektheitsbeweis (3): das fehlende Puzzlestück

- Wissen wir nun, dass am Ende des Algorithmus $P_3 = ab$ ist?
- Nein: bisher nur bewiesen, dass $P_3 + X_3 Y_3 = ab$
- Wertetabelle zeigt, dass im Beispiel aber $X_3 = 0$.
- Beweisen wir, dass auch das für *alle* Eingaben $a \in \mathbb{G}_8$ und $b \in \mathbb{N}_0$ gilt. Wie?
- Beobachtung: Die X_i werden der Reihe nach immer kleiner.
  - Und zwar immer um mindestens die Hälfte, denn X_i **div** $2 \leq X_i/2$.
  - Mit anderen Worten:

$$
\begin{aligned}
X_0 &\leq a \\
X_1 &\leq X_0/2 \leq a/2 \\
X_2 &\leq X_1/2 \leq a/4 \\
&\vdots
\end{aligned}
$$

- Wie man *die* Pünktchen weg bekommt wissen wir schon: vollständige Induktion

Korrektheitsbeweis (3): das fehlende Puzzlestück

- Wissen wir nun, dass am Ende des Algorithmus $P_3 = ab$ ist?
- Nein: bisher nur bewiesen, dass $P_3 + X_3 Y_3 = ab$
- Wertetabelle zeigt, dass im Beispiel aber $X_3 = 0$.
- Beweisen wir, dass auch das für *alle* Eingaben $a \in \mathbb{G}_8$ und $b \in \mathbb{N}_0$ gilt. Wie?
- Beobachtung: Die X_i werden der Reihe nach immer kleiner.
  - Und zwar immer um mindestens die Hälfte, denn X_i **div** $2 \leq X_i/2$.
  - Mit anderen Worten:

$$
\begin{aligned}
X_0 &\leq a \\
X_1 &\leq X_0/2 \leq a/2 \\
X_2 &\leq X_1/2 \leq a/4 \\
&\vdots
\end{aligned}
$$

- Wie man *die* Pünktchen weg bekommt wissen wir schon: vollständige Induktion

Korrektheitsbeweis (3): das fehlende Puzzlestück

- Wissen wir nun, dass am Ende des Algorithmus $P_3 = ab$ ist?
- Nein: bisher nur bewiesen, dass $P_3 + X_3 Y_3 = ab$
- Wertetabelle zeigt, dass im Beispiel aber $X_3 = 0$.
- Beweisen wir, dass auch das für *alle* Eingaben $a \in \mathbb{G}_8$ und $b \in \mathbb{N}_0$ gilt. Wie?
- Beobachtung: Die X_i werden der Reihe nach immer kleiner.
  - Und zwar immer um mindestens die Hälfte, denn X_i **div** $2 \leq X_i/2$.
  - Mit anderen Worten:

$$\begin{aligned} X_0 &\leq a \\ X_1 &\leq X_0/2 \leq a/2 \\ X_2 &\leq X_1/2 \leq a/4 \\ &\vdots \end{aligned}$$

- Wie man *die* Pünktchen weg bekommt wissen wir schon: vollständige Induktion

Korrektheitsbeweis (4): das fehlende Puzzlestück

- Die Induktion ist so einfach ist, dass wir sie hier schon nicht mehr im Detail durchführen müssen.
- Ergebnis

$$\forall i \in \mathbb{N}_0 : i < 4 \Longrightarrow X_i \leq a/2^i \ .$$

- Insbesondere ist also $X_2 \leq a/4$.
- Nach Voraussetzung ist $a < 8$
- folglich $X_2 < 8/4 = 2$.
- Da X_2 eine ganze Zahl ist, ist $X_2 \leq 1$.
- Und daher ist *immer* das zuletzt berechnete

$$X_3 = X_2 \ \mathbf{div} \ 2 = 0 \ .$$

- ... und damit gilt am Ende des Algorithmus *immer*: $P_3 = ab$

Eine kleine Zeitreise

Algorithmusbegriff

- Lösen einer Sorte quadratischer Gleichungen
- Zum informellen Algorithmusbegriff
- Korrektheit des Algorithmus zur Lösung von $x^2 + bx = c$
- Wie geht es weiter?
- Algorithmus zur Multiplikation nichtnegativer ganzer Zahlen
- Multiplikationsalgorithmus mit einer Schleife

Der Algorithmus, anders aufgeschrieben

Die Indizes sind gar nicht wichtig:

// Eingaben: $a \in \mathbb{G}_8, b \in \mathbb{N}_0$

$P \leftarrow 0$

$X \leftarrow a$

$Y \leftarrow b$

$x \leftarrow X$ **mod** 2

// — Algorithmusstelle — $i = 0$

$P \leftarrow P + x \cdot Y$

$X \leftarrow X$ **div** 2

$Y \leftarrow 2 \cdot Y$

$x \leftarrow X$ **mod** 2

// — Algorithmusstelle — $i = 1$

// — Algorithmusstelle — $i = 1$

$P \leftarrow P + x \cdot Y$

$X \leftarrow X$ **div** 2

$Y \leftarrow 2 \cdot Y$

$x \leftarrow X$ **mod** 2

// — Algorithmusstelle — $i = 2$

$P \leftarrow P + x \cdot Y$

$X \leftarrow X$ **div** 2

$Y \leftarrow 2 \cdot Y$

$x \leftarrow X$ **mod** 2

// — Algorithmusstelle — $i = 3$

- dreimal exakt der gleiche Algorithmustext
- das kürzen wir ab:

  for ⟨*Schleifenvariable*⟩ ← ⟨*Startwert*⟩ **to** ⟨*Endwert*⟩ **do**

  ⟨*sogenannter Schleifenrumpf, der*⟩

  ⟨*aus mehreren Anweisungen bestehen darf*⟩

  od

- Bedeutung:
  - Schleifenrumpf wird nacheinander für jeden Wert der ⟨*Schleifenvariable*⟩ durchlaufen
  - als erstes für den ⟨*Startwert*⟩
  - Bei jedem weiteren Durchlauf wird die ⟨*Schleifenvariable*⟩ um 1 erhöht.
  - Der letzte Durchlauf findet für den ⟨*Endwert*⟩ statt.
  - Falls ⟨*Endwert*⟩ $<$ ⟨*Anfangswert*⟩, wird der Schleifenrumpf überhaupt nicht durchlaufen.

$/\!/$ Eingaben: $a \in \mathbb{G}_8, b \in \mathbb{N}_0$

$X \leftarrow a$

$Y \leftarrow b$

$P \leftarrow 0$

$x \leftarrow X$ **mod** 2

for $i \leftarrow 0$ **to** 2 **do**

  $/\!/$ — Algorithmusstelle — i

  $P \leftarrow P + x \cdot Y$

  $X \leftarrow X$ **div** 2

  $Y \leftarrow 2 \cdot Y$

  $x \leftarrow X$ **mod** 2

  $/\!/$ — Algorithmusstelle — $i + 1$

od

$/\!/$ Ergebnis: $P = a \cdot b$

Bedeutung des Induktionsbeweises für die Schleifenschreibweise

- Entfernen der Indizes aus den Aussagen

$$\mathcal{A}_i : \qquad P_i + X_i Y_i = ab$$

- liefert

$$P + XY = ab$$

- Alle sehen gleich aus!
- $\mathcal{A}_i$ war Aussage darüber, was „an Algorithmusstelle i" gilt.
- Das bedeutet nun: nach i Schleifendurchläufen bzw. vor dem $i+1$-ten Schleifendurchlauf.
- Bewiesen: Aus der Gültigkeit von $\mathcal{A}_i$ folgt die von $\mathcal{A}_{i+1}$.
- Wir haben also gezeigt:

  Wenn die Aussage $P + XY = ab$ vor dem einmaligen Durchlaufen des Schleifenrumpfes gilt, dann gilt sie auch hinterher wieder.

- Diese Aussage ist eine sogenannte Schleifeninvariante.

Schleifeninvariante

- Der Induktionsanfang war nichts anderes als der Nachweis, dass die Schleifeninvariante vor dem ersten Betreten der Schleife stets gilt.
- Der Induktionsschritt war der Nachweis, dass die Wahrheit der Schleifeninvariante bei jedem Durchlauf erhalten bleibt.

- Also:
  - Wenn die Schleife jemals zu einem Ende kommt
  - und etwas anderes ist bei einer **for**-Schleife wie eben beschrieben gar nicht möglich,
  - dann gilt die Schleifeninvariante auch zum Schluss.

Schleifeninvariante

- Der Induktionsanfang war nichts anderes als der Nachweis, dass die Schleifeninvariante vor dem ersten Betreten der Schleife stets gilt.
- Der Induktionsschritt war der Nachweis, dass die Wahrheit der Schleifeninvariante bei jedem Durchlauf erhalten bleibt.

- Also:
  - Wenn die Schleife jemals zu einem Ende kommt
  - und etwas anderes ist bei einer **for**-Schleife wie eben beschrieben gar nicht möglich,
  - dann gilt die Schleifeninvariante auch zum Schluss.

Verallgemeinerung des Algorithmus für beliebig große Eingaben a

- Wir wollen, dass der Algorithmus für alle $a \in \mathbb{N}_0$ funktioniert.
- Wo wurde die Bedingung $a < 8$ verwendet?
- nur an einer Stelle: beim Nachweis, dass $X_3 = 0$ ist.
- Für z. B. $a = 4711$ ist man natürlich nach drei Schleifendurchläufen noch nicht bei $X = 0$.
- Dafür muss man öfter den Wert X halbieren.
- Es ist also eine größere Anzahl n von Schleifendurchläufen notwendig.
- Wieviele? Man betrachte noch einmal Ungleichung

$$X_i \leq a/2^i$$

- Man ist fertig, wenn vor dem letzten Durchlauf gilt: $X_{n-1} \leq 1$.

Verallgemeinerung des Algorithmus für beliebig große Eingaben a (2)

- $X_{n-1} \leq 1$ gilt, wenn

$$
\begin{aligned}
& & a/2^{n-1} &\leq 1 \\
&\text{also} & a &\leq 2^{n-1} \\
&\text{also} & n-1 &\geq \log_2 a \\
&\text{also} & n &\geq 1 + \log_2 a
\end{aligned}
$$

Verallgemeinerung des Algorithmus für beliebig große Eingaben a (3)

$/\!/$ Eingaben: $a \in \mathbb{N}_0, b \in \mathbb{N}_0$

$X \leftarrow a$

$Y \leftarrow b$

$P \leftarrow 0$

$x \leftarrow X \textbf{ mod } 2$

$n \leftarrow 1 + \lceil \log_2 a \rceil$

for $i \leftarrow 0$ **to** $n - 1$ **do**

$\quad P \leftarrow P + x \cdot Y$

$\quad X \leftarrow X \textbf{ div } 2$

$\quad Y \leftarrow 2 \cdot Y$

$\quad x \leftarrow X \textbf{ mod } 2$

od

$/\!/$ Ergebnis: $P = a \cdot b$

Was ist wichtig

Das sollten Sie mitnehmen:

- informeller *Algorithmusbegriff*
- Andeutungen von Fortsetzungen der Vorlesung und des Studiums
- *Schleifeninvarianten*
- Induktionsbeweise nützlich für Korrektheitsnachweis von Schleifen

Das sollten Sie üben:

- Schleifeninvarianten finden
  - Wertetabellen können helfen
- Korrektheitsbeweise finden