Informatik 1

Wintersemester 2026/27

Universität Tübingen

Prof. Dr. Torsten Grust

torsten.grust@uni-tuebingen.de

willkommen!



Info1... damals im WS 20/21



Torsten Grust?

1989–1994	Studium der Informatik an der TU Clausthal
1994–1999	Promotion an der U Konstanz
2000	*Visiting Researcher*, IBM (USA)
2000–2004	Habilitation an der U Konstanz
2004–2005	Professor für Datenbanksysteme an der TU Clausthal
2005–2008	Professor für Datenbanksysteme an der TU München
2008–heute	Professor für Datenbanksysteme an der U Tübingen

Informatik 

- Computer Science
  Die Wissenschaft (digitaler) Rechner**systeme**

  “In der Informatik geht es genauso wenig um Computer wie in der Astronomie um Teleskope.” —Edsger W. Dijkstra

- Computing Science
  Die Wissenschaft computer-unterstützten **Rechnens**

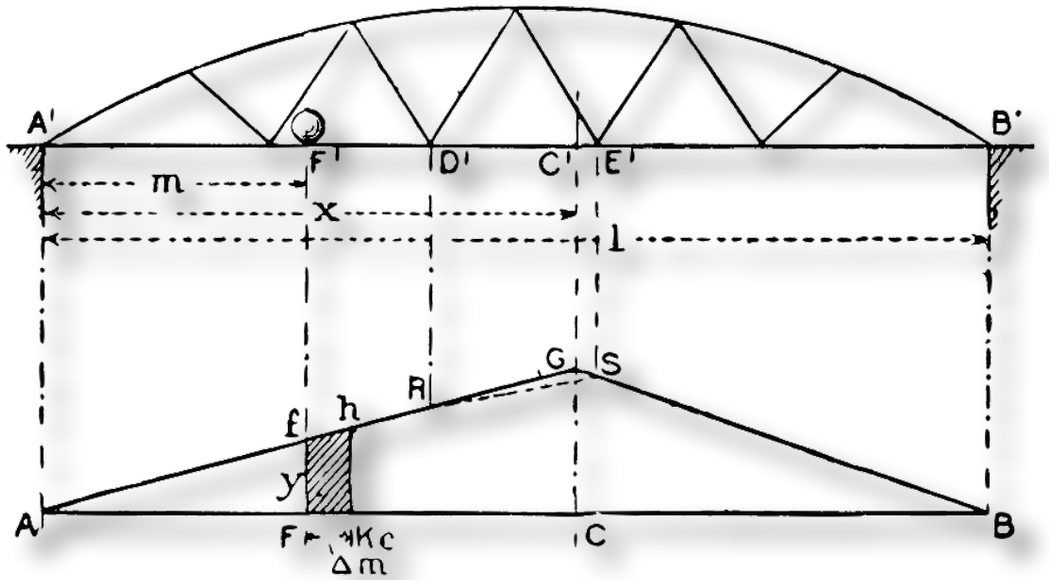
- **Informatik** = Information + Elektrotechnik + Mathematik
  Die Wissenschaft der systematischen, computer-unterstützten Informationsverarbeitung

  (seit 1967; *informatique*, *informatika*, *informatyka*, *informatics*, 计算机科学)

Informatik:



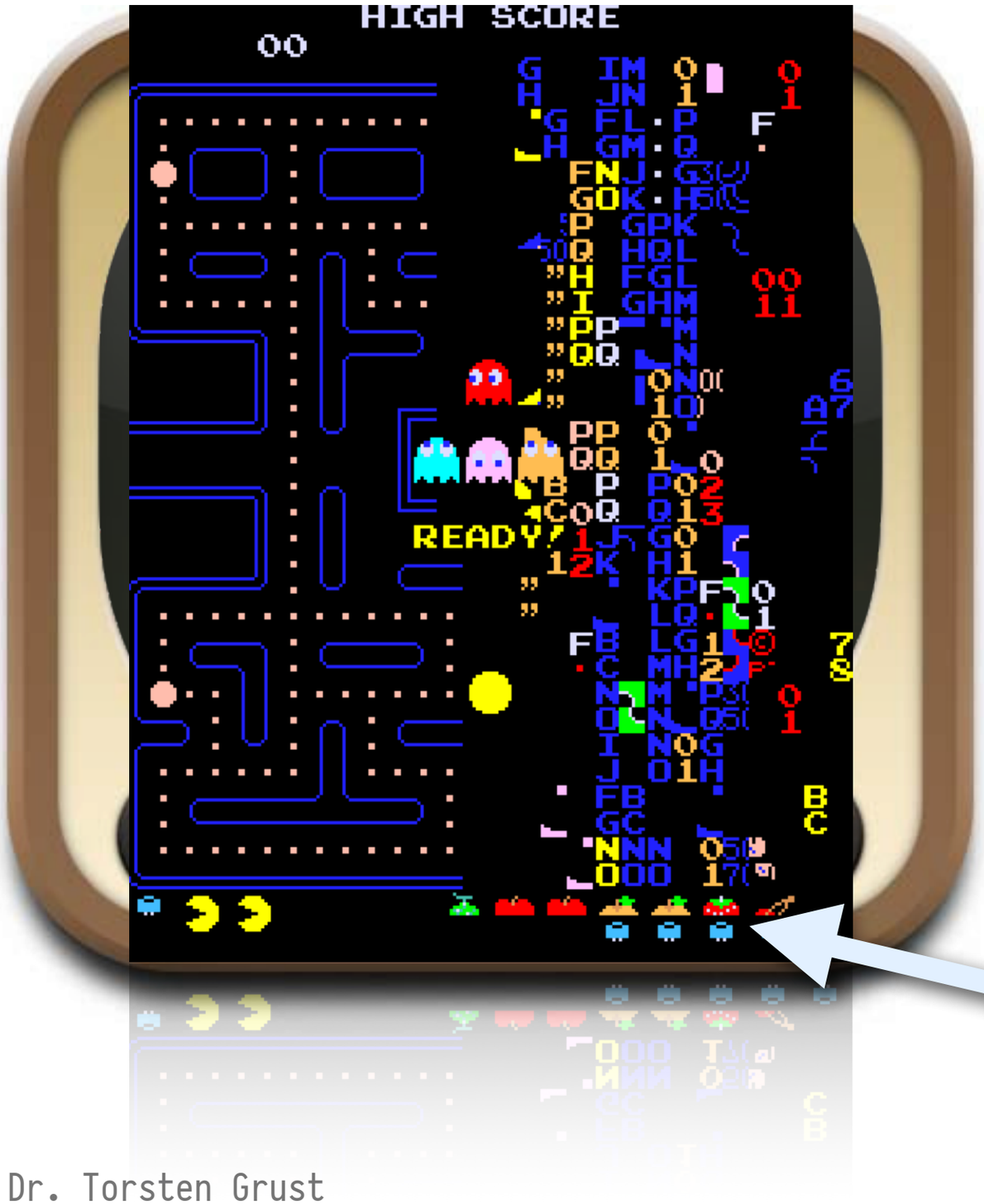
Informatik: (auf dem Weg zur) Ingenieurswissenschaft



- Anwendung erprobter (teils formal korrekt bewiesener) **Konstruktionsprinzipien** für Software vs. *“Move fast and break things!”*

- Systematisches **Testen** und Verifizierung mathematischer (oft: algebraischer) **Eigenschaften**

The Pac-Man Kill Screen



- Pac-Man: Level l in 8-Bit-Register
- 8 Bit: $0 \leqslant l \leqslant 255$
- Nach Level 255: Überlauf ($255 + 1 \rightsquigarrow 0$)
- Eigentliches Problem: Prozedur zum Zeichnen von $1 \ldots l-1$ Früchten

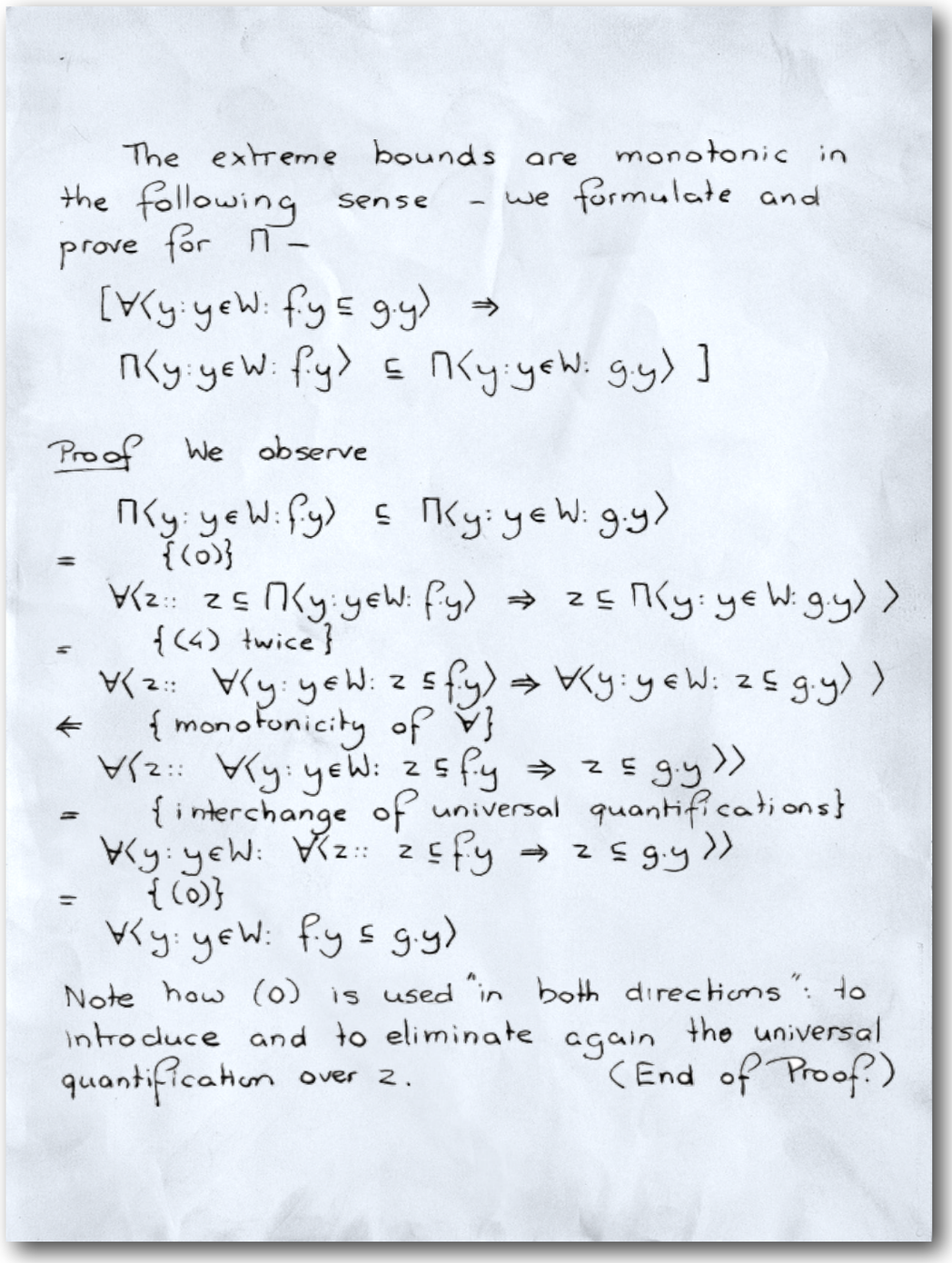
Ariane 5: 10 Jahre, 7×10^9 US$

- Erster “Flug”, Juni 1996

- Umwandlung einer 64-Bit Fliesskommazahl in eine 16-Bit Ganzzahl löst Ausnahmebehandlung aus

⇒ Ausfall der Horizontalsteuerung

Informatik: Formale Wissenschaft



The extreme bounds are monotonic in the following sense – we formulate and prove for Π –

$$[\langle \forall y : y \in W : f.y \sqsubseteq g.y \rangle \Rightarrow \Pi\langle y : y \in W : f.y \rangle \sqsubseteq \Pi\langle y : y \in W : g.y \rangle]$$

Proof We observe

$$\begin{aligned}
&\Pi\langle y : y \in W : f.y \rangle \sqsubseteq \Pi\langle y : y \in W : g.y \rangle \\
=\ &\quad \{(0)\} \\
&\langle \forall z :: z \sqsubseteq \Pi\langle y : y \in W : f.y \rangle \Rightarrow z \sqsubseteq \Pi\langle y : y \in W : g.y \rangle \rangle \\
=\ &\quad \{(4) \text{ twice}\} \\
&\langle \forall z :: \langle \forall y : y \in W : z \sqsubseteq f.y \rangle \Rightarrow \langle \forall y : y \in W : z \sqsubseteq g.y \rangle \rangle \\
\Leftarrow\ &\quad \{\text{monotonicity of } \forall\} \\
&\langle \forall z :: \langle \forall y : y \in W : z \sqsubseteq f.y \Rightarrow z \sqsubseteq g.y \rangle \rangle \\
=\ &\quad \{\text{interchange of universal quantifications}\} \\
&\langle \forall y : y \in W : \langle \forall z :: z \sqsubseteq f.y \Rightarrow z \sqsubseteq g.y \rangle \rangle \\
=\ &\quad \{(0)\} \\
&\langle \forall y : y \in W : f.y \sqsubseteq g.y \rangle
\end{aligned}$$

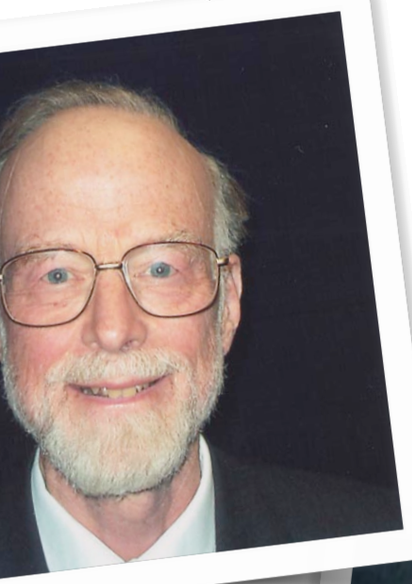
Note how (0) is used "in both directions": to introduce and to eliminate again the universal quantification over z. (End of Proof.)

Excerpt of EWD 1107-0

- Viele Objekte der Informatik (Programme, Daten) zeigen interessante, mathematische Strukturen
- Bereits ein relativ kleiner mathematischer Werkzeug-kasten kann entscheidende Einsichten bringen
- "… $f(f(x)) = f(x)$." Gut zu wissen.

Informatik: Junge Wissenschaft

- Viele Informatik–Legenden sind am Leben!
  (Von den meisten gibt es zumindest Fotos…)



Tony Hoare

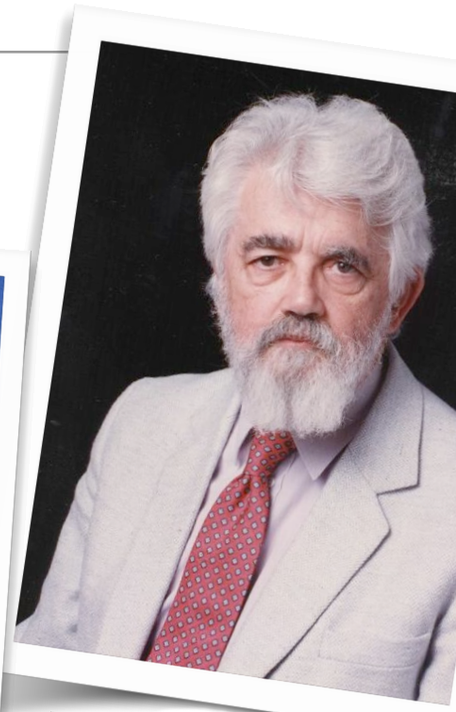


Radia Perlman

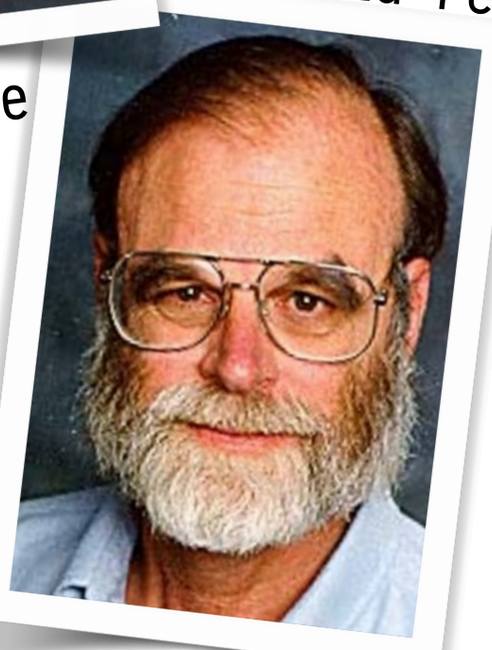


Donald Knuth





John McCarthy



Jim Gray



Alonzo Church



Grace Hopper



Edsger Dijkstra

Alan Turing

Informatik: Gesellschaftswissenschaft



Informatik: Gesellschaftswissenschaft

- Alle Bereiche menschlichen (Zusammen–)Lebens vertrauen heute auf computer–gestützte Technologie

  – Arbeitswelt, Beziehungen, Finanzen, Katastrophen–schutz, Kommunikation, Kriminalistik, Kunst, Künstliche Intelligenz, Medizin, Nachrichtenwesen, Politik, Unterhaltung, Verkehr, Wissenschaft, ...

⇒ Informatikerinnen und Informatiker können eine Vielzahl gesellschaftlicher Probleme lösen — oder solche schaffen

Auch nur Menschen... Informatiker:innen



How it all started — Female Computers



Female Computers

- Die Programmierung war die Domäne der Frauen...
- ...bis die Männer entdeckten wie lukrativ (und spaßig) die Sache ist.
- **Liebe Kommilitoninnen, holt euch die Programmierung zurück!**





Informatik 1

Ein Parkplatz–Puzzle



*“Auf einem Parkplatz stehen PKWs und Motorräder (ohne Beiwagen). Zusammen seien es **n** Fahrzeuge mit insgesamt **r** Rädern.*
*Bestimme die Anzahl **P** der PKWs.”*

- Dies beschreibt gleich eine ganze Menge von Puzzles, für jede Wahl von n und r.

- Also haben wir es mit folgenden Funktionen zu tun:

  $P(n,r)$ Anzahl der PKWs
  $M(n,r)$ Anzahl der Motorräder

Ein Parkplatz–Puzzle



- PKWs und Motorräder machen alle n Fahrzeuge aus. PKWs steuern jeweils 4, Motorräder jeweils 2 Räder zu den r Rädern bei:

$$P + M = n \qquad M = n - P$$

$$4P + 2M = r$$

$$4P + 2(n - P) = r$$

$$4P + 2n - 2P = r$$

$$2P = r - 2n$$

$$P = 1/2\,(r - 2n)$$

Not Parkplatz—puzzled anymore?

- Eine Funktion als Lösung: $P(n,r) = 1/2\ (r - 2n)$. Sieht gut aus.

- Oder?

  - $P(3,9) = 1.5$ Halbe PKWs?
  - $P(5,2) = -4$ Noch 4 PKWs einparken, dann ist der Parkplatz leer...
  - $P(2,10) = 3$ Subtil (mehr PKWs als Fahrzeuge).

Verträge!

- Mit der Funktion $P(n,r)$ haben wir soweit abstrahiert und uns damit vom tatsächlichen Parkplatz entfernt, bis wir in der Realität geltende Einschränkungen aus den Augen verloren haben.

- $P(n,r)$ erfüllt seine Aufgabe. Aber nur, sofern wir uns **bei der Anwendung einschränken:**

$$r \text{ gerade} \quad \text{und} \quad 2n \leqslant r \leqslant 4n$$

- Funktion $P(n,r)$ kommt quasi mit **Vertragsbedingungen.**

Programmieren mit Funktionen

- **Funktionen** gehören zu den zentralen Bausteinen der Mathematik:

$$f(x_1, x_2, \ldots, x_n)$$

- In **Scheme** adoptieren wir diese Idee und nutzen **Funktionen als Bausteine**, aus denen wir ganze **Programme konstruieren**:

```
(f x1 x2 ... xn)
```

Programmieren mit Funktionen

- Diese einfache Idee des *Programmierens mit Funktionen* begleitet uns durch das gesamte Semester:

1. **Konstruiere** (oder nutze die eingebaute) Funktion *f*, dann
2. **wende** *f* auf die benötigte Anzahl von **Argumenten an:**

```
(+ 40 2)
```

```
(odd? 42)
```

Das Leibniz–Prinzip

- Aber hinter Funktionen steckt mehr als nur ihre kompakte Notation: ein nachvollziehbares, verlässliches Verhalten!

- In der Mathematik verlassen wir uns auf das folgende (wähle Funktion f und die Argumente x, y beliebig):

$$x = y \quad \text{und} \quad f(x) = z \quad \Rightarrow \quad f(y) = z$$

In Worten: um das Verhalten von f vorherzusagen, benötigen wir **ausschliesslich** das Argument (hier: x, y).

Gottfried Wilhelm von Leibniz

- 1646—1716
- Wesentliche Beiträge zur Mathematik, Paläontologie, Philosophie, Physik, Politik, Rechtswesen, ...
- 1673: Rechenmaschine



Programmieren “ohne Leibniz”?

- Sicher, das ist schon möglich…
- Vielleicht erstaunlich: die weitaus meisten Programmiersprachen sichern den Entwicklern das Leibniz–Prinzip **nicht** zu.
- Die Vorhersagbarkeit des Verhaltens einer Funktion allein mittels ihres Argument geht verloren:

$$x = y \text{ und } f(x) = z \not\Rightarrow f(y) = z$$



“Ohne Leibniz”: Konsequenzen

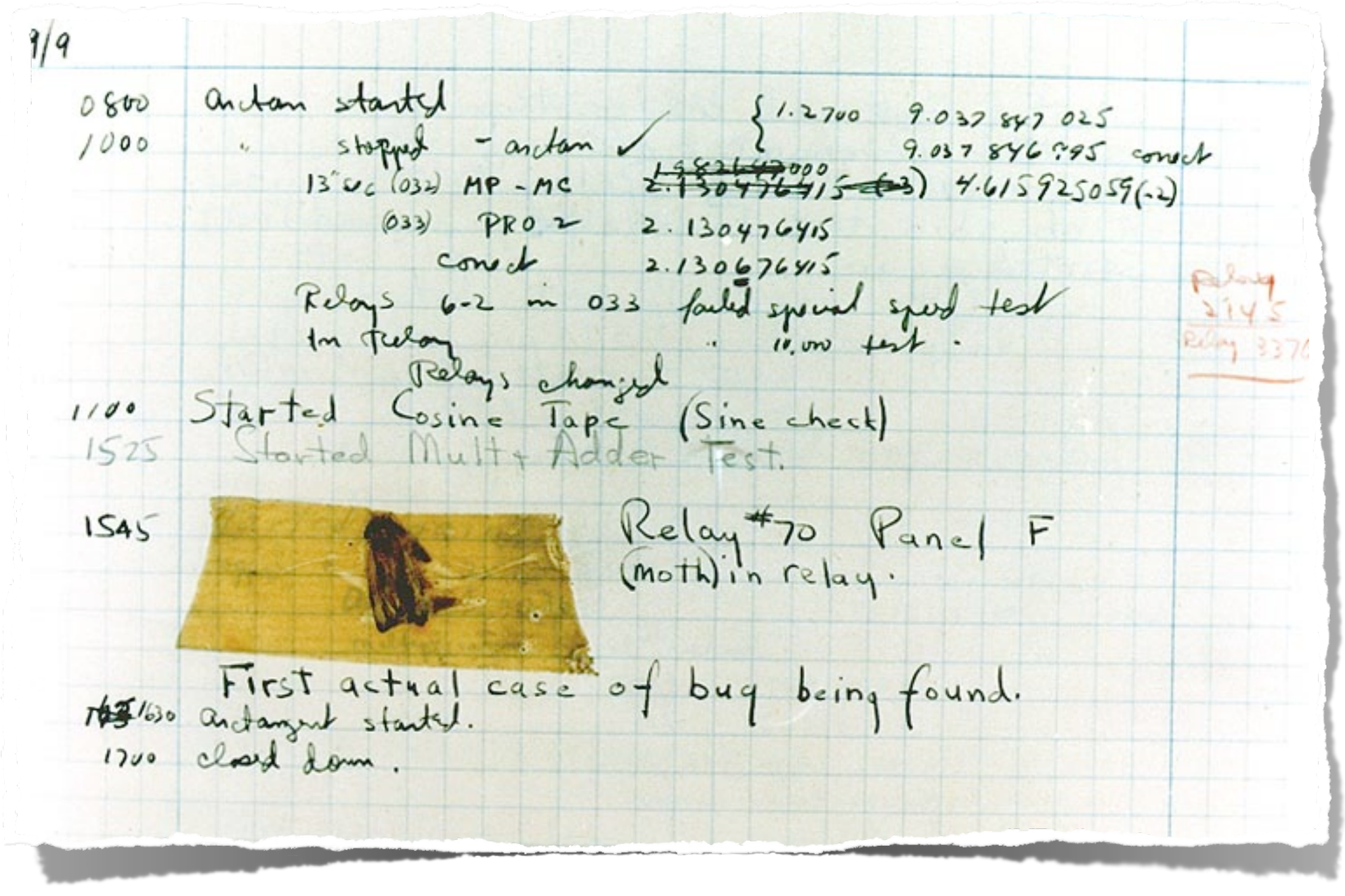
- Viele intuitive Annahmen gelten bei der Programmkonstruktion i.A. nicht mehr. Etwa:

$$f(x) + f(x) \neq 2 \times f(x)$$

- Das Verhalten von f kann von einer unübersichtlichen **Vielzahl von** (**impliziten**) **Parametern** abhängen:

  - vorhergegangene Anwendungen weiterer Funktionen (inkl. f), Systemzustand, Zeit, Eingaben des Users, Mondphase, …

Programmfehler (Bugs)



9/9

0800 Arctan started
1000 " stopped - arctan ✓ { 1.2700 9.037 847 025
9.037 846 ?95 correct
~~1.9821647000~~
13"UC (032) MP - MC ~~2.13047615 (033)~~ 4.615925059(-2)
(033) PRO 2 2.13047615
correct 2.13067615
Relays 6-2 in 033 failed special speed test
In relay " 11.00 test.
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Mult + Adder Test.
1545 Relay #70 Panel F
(moth) in relay.

First actual case of bug being found.

1700 closed down.

Relay 2145
Relay 3370

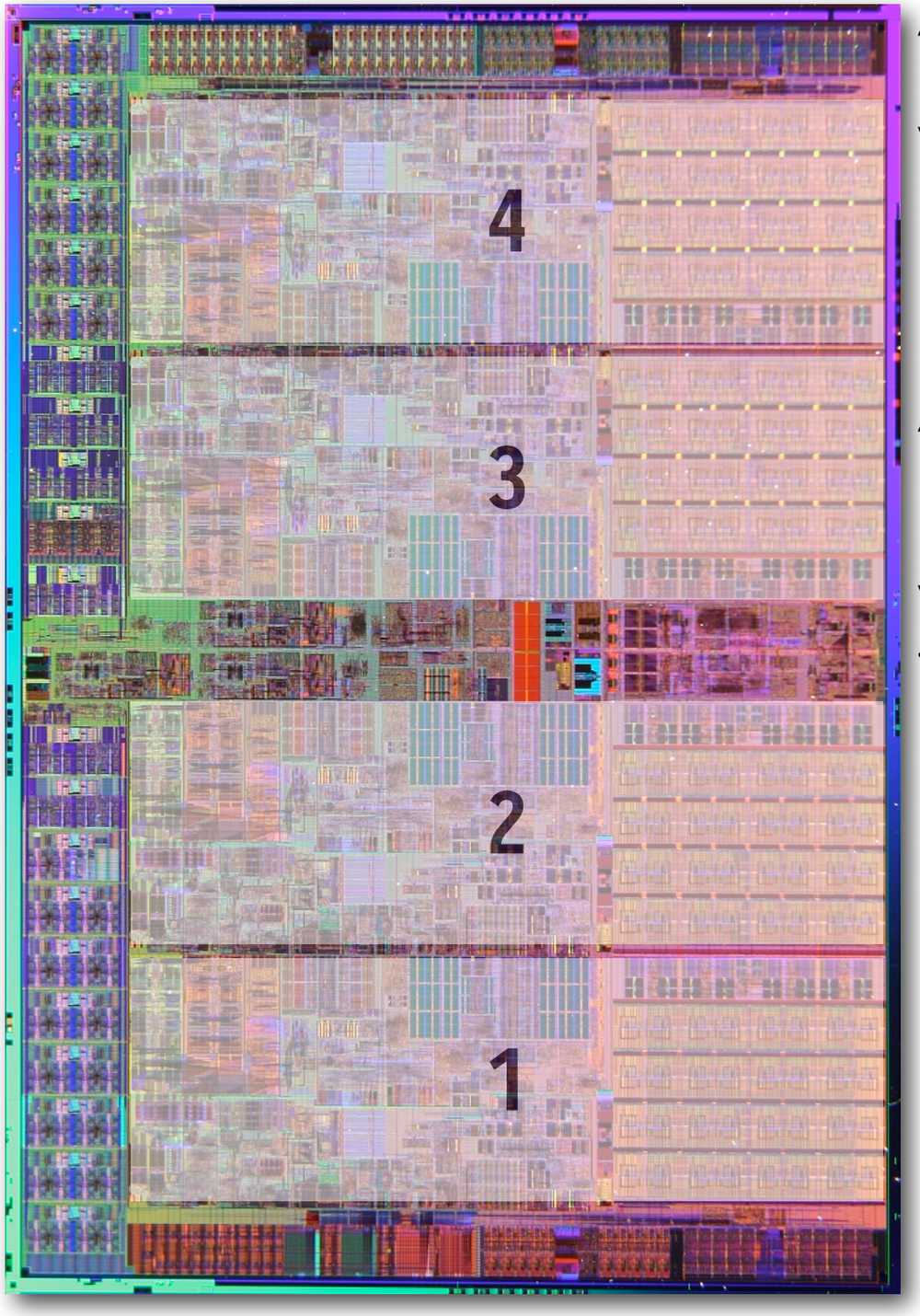
Motte im Relay 70 eines Mark II Aiken,
entdeckt durch Grace Murray Hopper, 1945

Funktionen und moderne Rechner

- Sei (f (g x) (h y))
  Teil eines Programms.

- Laut Leibniz **beeinflussen** sich die Berechnungen von (g x) und (h y) **nicht** gegenseitig.

- Warum also nicht (g x) und (h y) **parallel** berechnen?



Intel Core i7 (Nehalem), 4-core CPU (2008)

All the big guys are doing it…

- Programmieren mit Funktionen (FP oder *functional programming*) hat seit Jahren besonderen Einfluss:
  - Industrielle Forschung & Entwicklung baut auf FP:

    Verizon, Bloomberg LP, Comcast, S&P Capital, Facebook, Netflix, Twitter, Jane Street, Standard Chartered, Citrix, Nokia, Alcatel Lucent, Google, Credit Suisse, Deutsche Bank, McGraw-Hill, ING, …
  - Ericsson’s AXD301 ATM Switch basiert auf Erlang:

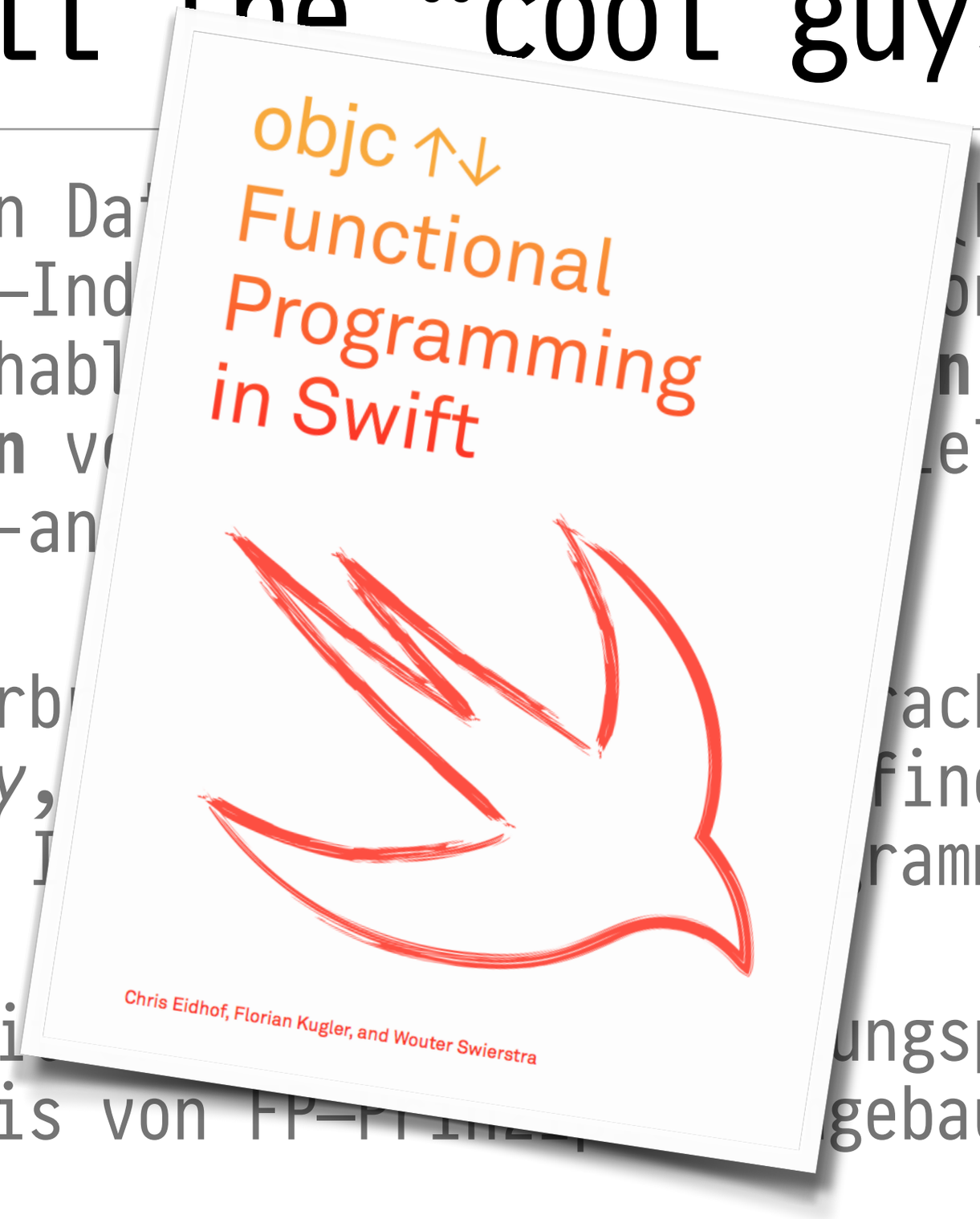
    The measured reliability is quoted as being 99.9999999% (9 nines) corresponding to a downtime of 31 ms/year.

...and all the “cool guys”, too

- Die grössten Da… …bspw. Googles Web-Ind… …on *MapReduce* —einer “Schabl… …n **Konstruktion** v… …elen Programmen—an…

- In allen verb… …achen (*C#*, *Python*, *Ruby*, … …finden sich wesentliche I… …rammierung.

- Apple hat mi… …ungsplattform auf der Basis von FP-Prinzi… …gebaut.



Das Info1–Team



Björn Bamberg, Nico Faden, Ludwig Kolesch, Tim Fischer

Wissenschaftliche Mitarbeiterinnen und Mitarbeiter (auch: “*Assis*”)

Organisation des gesamten Übungsbetriebes

Die Info1–Tutorinnen und -Tutoren

Mikaela Brauchler, Levi Drieling, Mara Germann, Julius Grupp, Ali Karim Güzel,
Oskar Hirsch, Eva Junge, Marinela Kolaj, Benedict Konhäuser, Luisa Kraus, Elayo Kunze,
Emily Meyer, Jonas Molz, Shota Oniani, Ludwig Patzold, Johannes Philippin, Ole Preiss,
Jannik Prütz, Fabio Valente, Dejan Vukelic, Lara Walter, Jannik Weberruß,
Finn Felix Weiser, Muni Zhong, Johannes von Schmettow

Tutorien, Korrektur, Racket-Party — ansprechbar und generell **hi**lfs**wi**llig

Der Info1–Übungsbetrieb

Übungsblätter + Tutorien

- Ausgabe wöchentlich (ÜB #0 am 14.10.)
- Bearbeitung: Mittwoch—Mittwoch
- Diskussion von ÜBs, Quizzes (ca. 4×)
- Bildung von 2er-Teams innerhalb einer Übungsgruppe

“Racket-Party”

- Montags in 2.033/2.034 (erste RP: 19.10.)
- Flexibler Zeitrahmen (14:15–17:00 Uhr)
- Hinweise zum letzten und kommenden ÜB
- Individuelles Arbeiten mit Unterstützung der Tutorinnen und Tutoren

❶ Forum

- Neueste Informationen und Diskussion zu V und Ü (unter Studierenden/Tutoren/Dozenten),
  Tips + Tricks, Downloads (ÜBs + Slides + Code)

https://forum-db.cs.uni-tuebingen.de

https://handin-db.cs.uni-tuebingen.de

- Registrierung zur Info1 und
  Anmeldung zu den Tutorien **bis Freitag, 9:00 Uhr** ⚠️

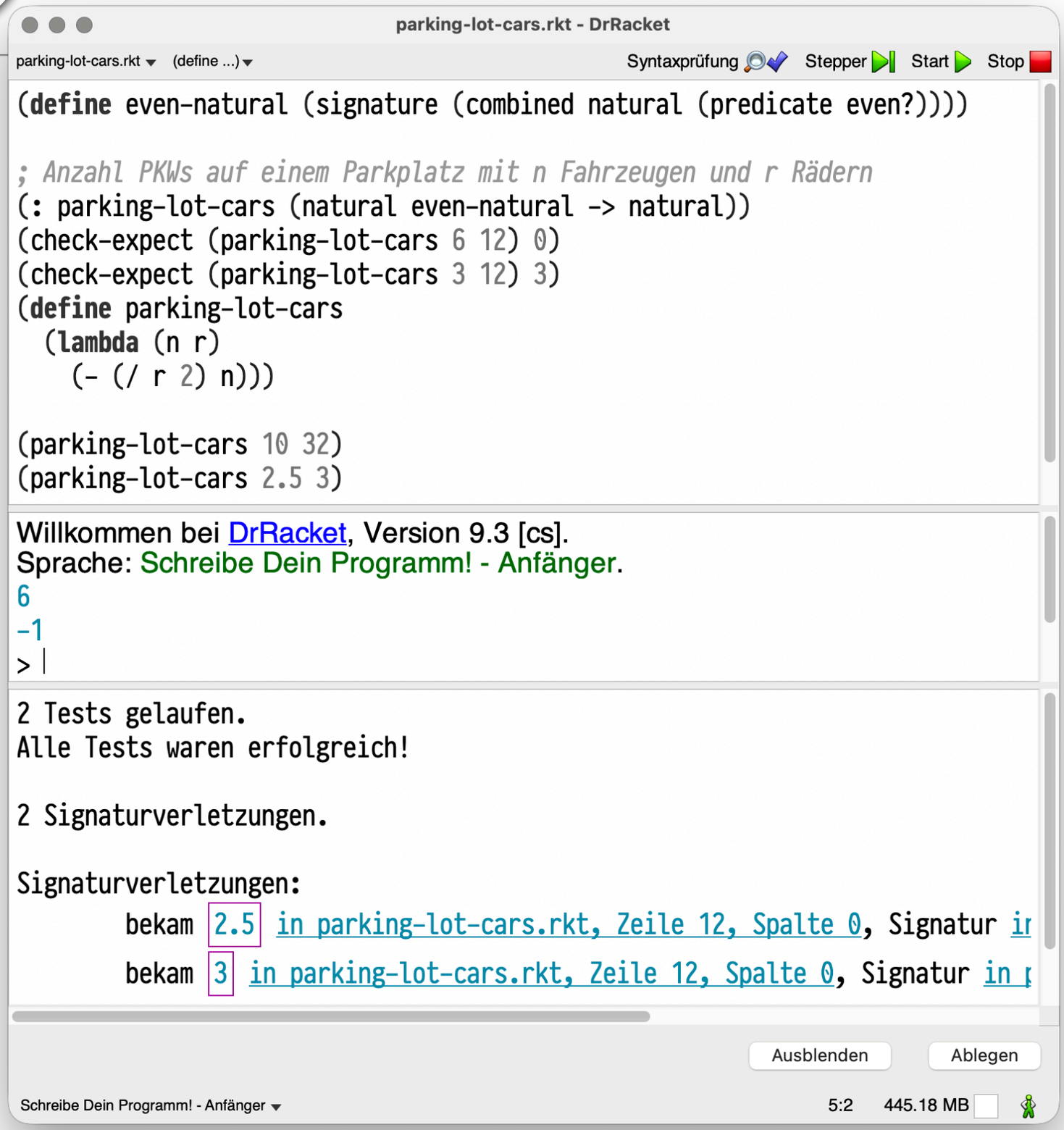
❷ Hand-In

DrRacket

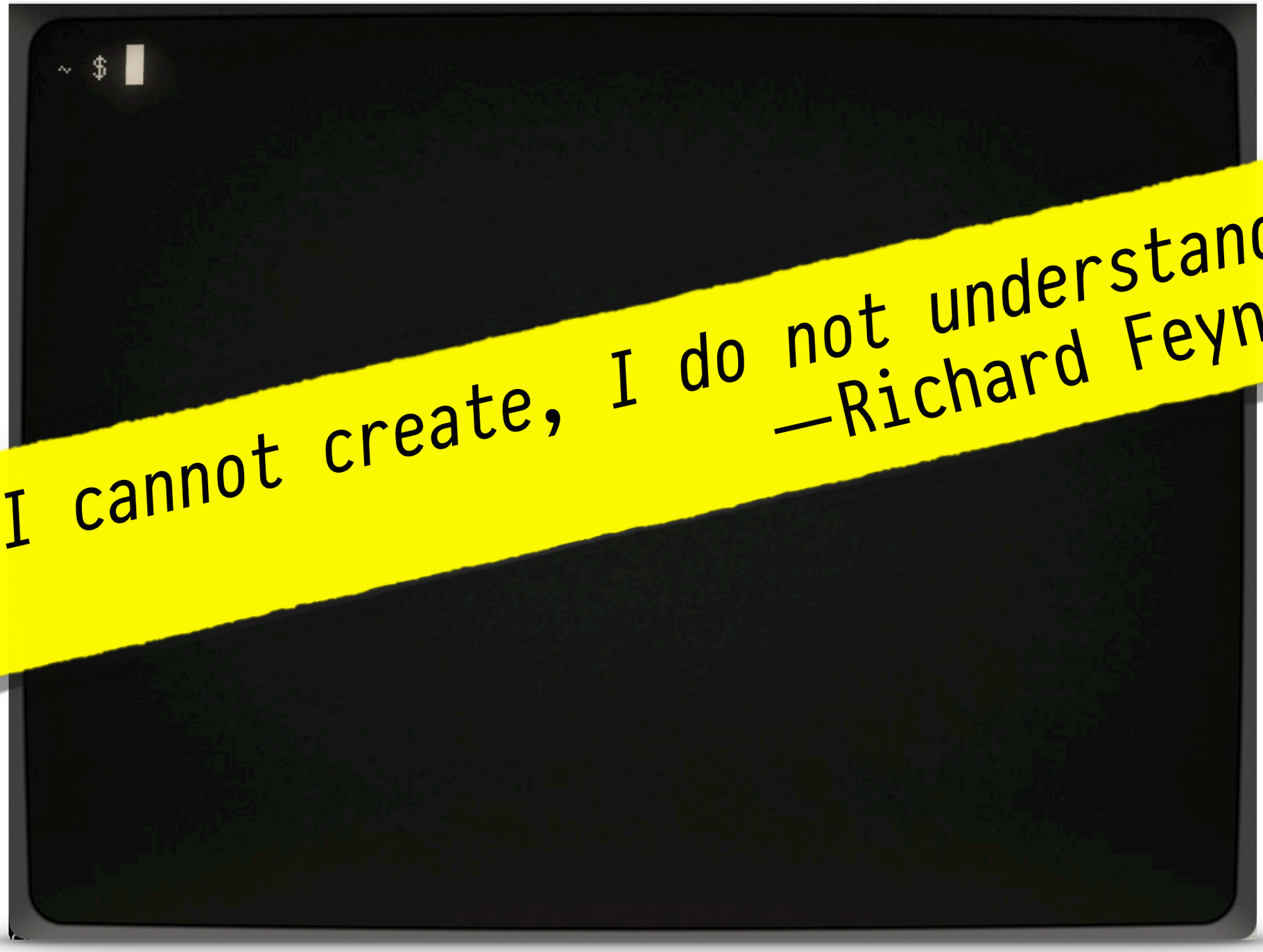


- Programmierumgebung für **Scheme**, einer Sprache die funktionales Programmieren (und mehr) unterstützt.

- Interaktiv, hilfsbereit, konfigurierbar, kostenfrei, quasi überall verfügbar.



Die *Big Averaging Machines* kennen Scheme...



$$f(x,y)$$

STAR WARS

Informatik 1
FAQs

