



# Einführung in die Programmierung

Prof. Dr. Bertrand Meyer

Vorlesung 1: Übersicht & Willkommen



Unsere Assistenten sprechen verschiedene Sprachen:

- Deutsch (in mehreren Varianten)
- Englisch
- Italienisch
- Chinesisch
- Russisch
- Französisch
- ...

Die Übungsgruppen werden auf deutsch (7) oder auf englisch (3) gehalten.





Nach erfolgreichem Abschluss dieser Vorlesung werden Sie:

- Die Schlüsselkonzepte des Programmierens kennen
- Viele verschiedene Programmierprobleme aus verschiedenen Bereichen lösen können
- Die grundsätzlichen Hardware- und Softwarewerkzeuge kennen
- Eine Programmiersprache beherrschen: Eiffel
- Die Grundkonzepte des Designs, der Implementierung und der Wartung von Softwaresystemen kennen ("software engineering").



- Was ist Software?
- Objekte & Programme
- Schnittstellen und das Klassenkonzept
- Logik und Verträge (contracts)
- Das Laufzeitmodell: Objekterzeugung, Referenzen
- Syntaxbeschreibung
- Kontrollstrukturen
- Vererbung
- Generik
- Rekursion
- Datenstrukturen
- Ereignisgesteuerte Programmierung & Agents
- Topologisches Sortieren
- Einführung ins Software Engineering



- Besuchen Sie alle Vorlesungen
- Lesen Sie die Unterlagen — das Buch und die Folien — jeweils **vor** den Vorlesungen  
(Bem.: Folien werden häufig nach der Vorlesung aktualisiert)
- Nehmen Sie eine Druckversion der Folien mit und machen Sie sich Notizen
- Besuchen Sie alle Übungsstunden
- Machen Sie alle Übungen  
(Sie brauchen sie für das "Testat")
- Falls Sie etwas nicht verstehen, fragen Sie nach  
(es gibt keine dummen Fragen)



Falls Sie **bereits programmiert** haben, nutzen Sie diesen Vorteil, aber seien Sie auch offen für eine neue Sichtweise; erkunden Sie Traffic

Falls Sie noch **nie programmiert** haben, keine Angst; es kann anfangs schwierig sein, aber Sie werden es schaffen.

**Mathematisches Wissen** ist genauso nützlich wie Programmiererfahrung



	2008	Frühere Jahre
Erfahrung mit Computern	<p><math>\leq 2</math> Jahre: 1%</p> <p>2 bis 4 Jahre: 3%</p> <p>5 bis 9 Jahre: 34%</p> <p><math>\geq 10</math> Jahre: 62%</p>	<p>(0%, 0%, 0%, 1%)</p> <p>(1%, 3%, 4%, 1%, 6%)</p> <p>(39%, 35%, 48%, 35%)</p> <p>(61%, 62%, 48%, 63%)</p>
Programmiererfahrung	<p>Keine: 18%</p> <p>Keine OOP: 22%</p> <p><math>\geq 100</math> Klassen: 17%</p>	<p>(12%, 19%, 18%, 14%)</p> <p>(20%, 26%, 33%, 38%)</p> <p>(8%, 11%, 15%, 10%, 5%)</p>





# **Die Industrie der reinen Ideen**



Diese Maschinen kann man nicht berühren, treten oder fallen lassen: sie sind immateriell

Aber es sind trotzdem Maschinen

Wir nennen sie **Programme** oder **Systeme**

Um ein Programm auszuführen, benötigt man eine materielle Maschine: einen Computer

Computer und technische Geräte: **Hardware**

Programme und der damit verbundene intellektuelle Wert: **Software**

# Software, wohin man auch schaut

---



Banken: verwaltet Millionen von Konti

Handel: entscheidet über Kauf und Verkauf

Verkehr: kontrolliert Züge, überwacht Flugzeuge...

- Einige Autos beinhaltet Software mit Millionen von Zeilen Programmcode

Reisen: Flug-, Zug-, und Hotelbuchungen

Kommunikation: Telefonie, Internet, ...

Behörden: verwaltet Steuern, überwacht Gesetze...

Gesundheitswesen: verwaltet Krankenakten, überwacht Geräte

Unterricht

Unterhaltung

Information

usw.

# Computer überall...



Banken

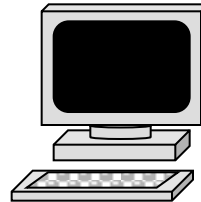
Flugzeuge, Autos...

Waschmaschinen

Mobiltelefone (> 70% des Werts)

Drucker

Morgen: Ihr T-Shirt...





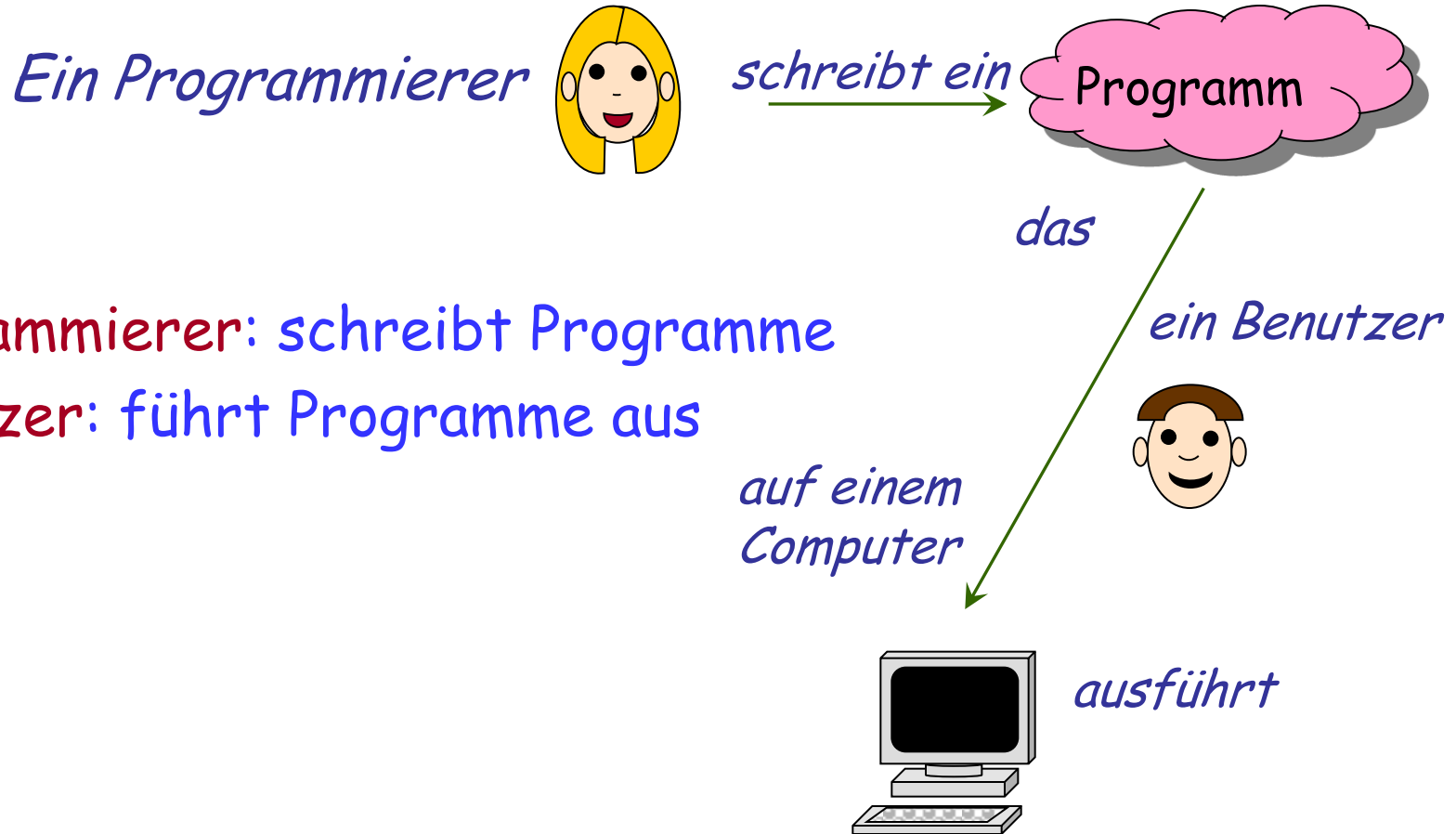
Computer sind universelle Maschinen. Sie führen das Programm aus, das Ihnen gegeben wird.

Ihre Vorstellungskraft ist die einzige Grenze

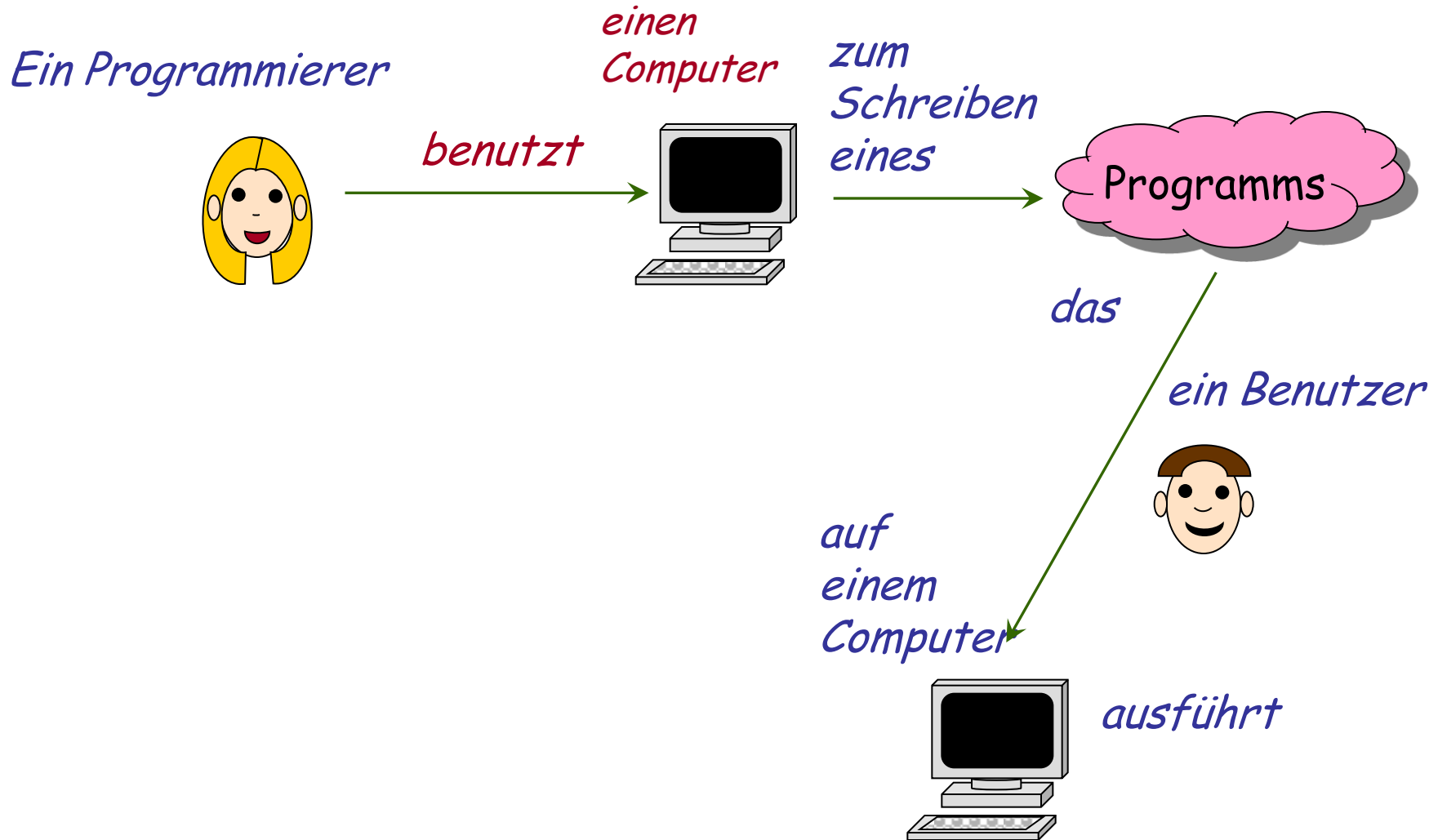
Die guten Nachrichten:

- Ihr Computer tut **genau das**, was in Ihrem Programm steht

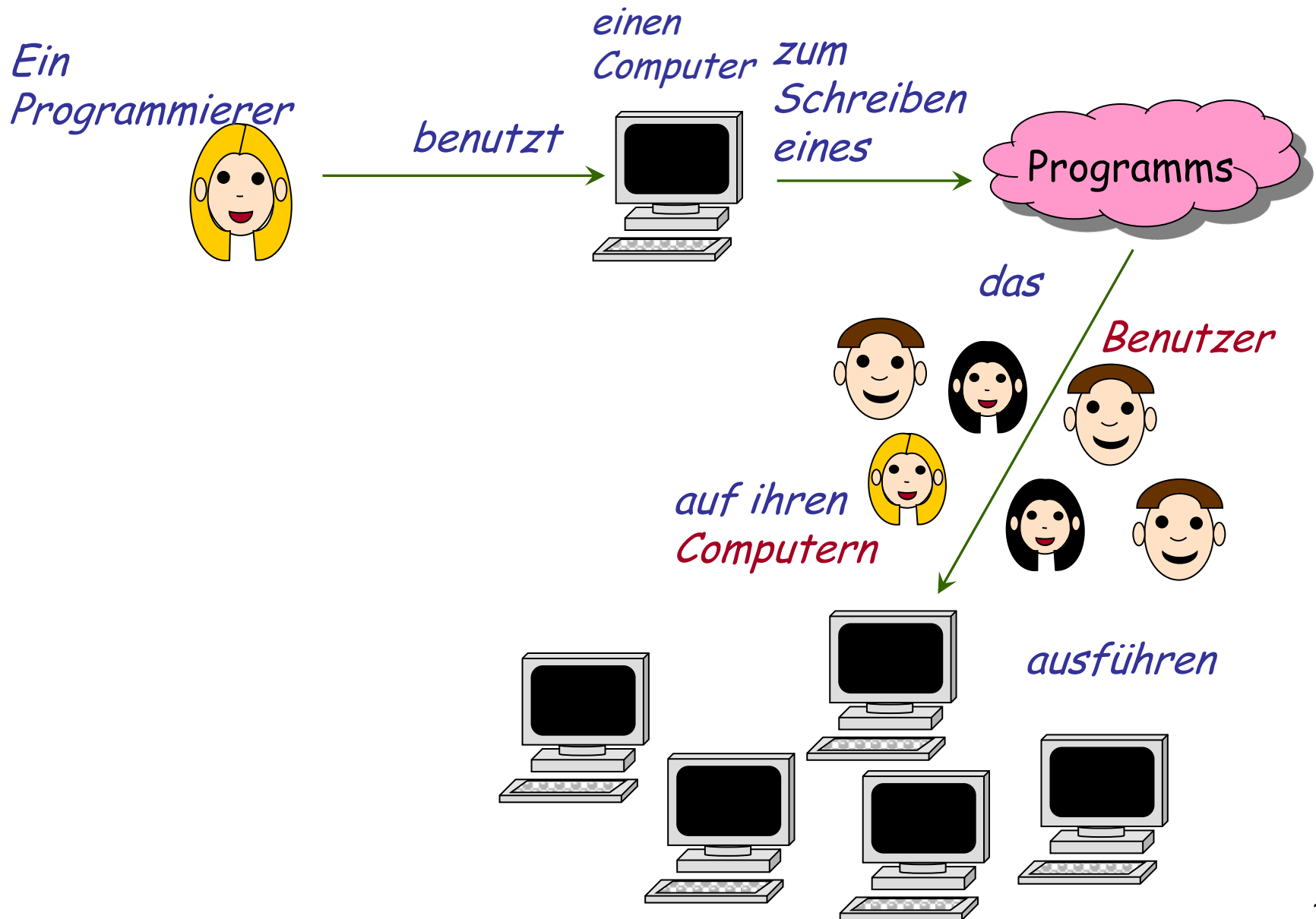
# Programme erstellen und ausführen



# Programme erstellen und ausführen

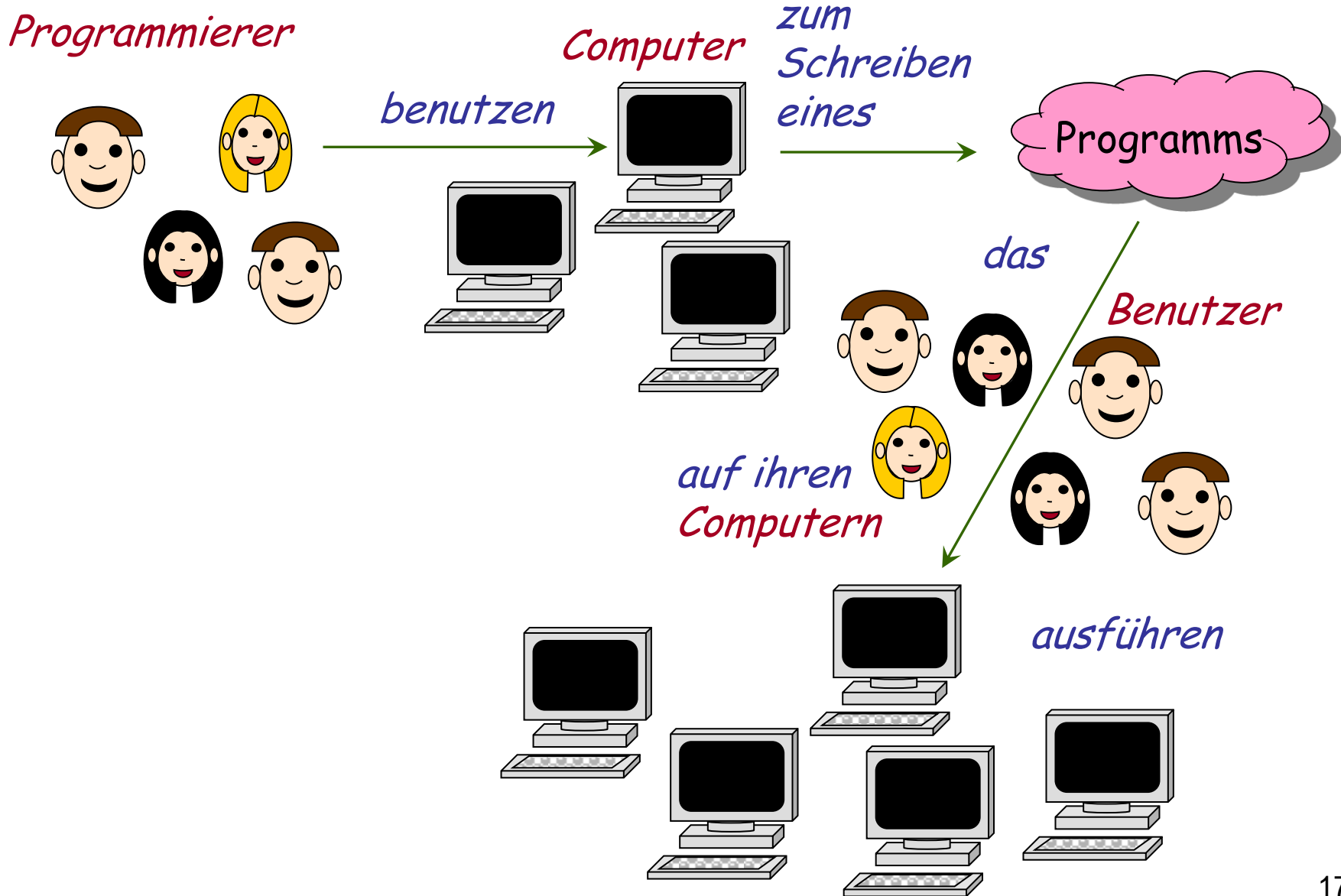


# Programme erstellen und ausführen





# Programme erstellen und ausführen





Computers sind universelle Maschinen. Sie führen das Programm aus, das Ihnen gegeben wird.

Ihre Vorstellungskraft ist die einzige Grenze

Die guten Nachrichten:

- Ihr Computer tut **genau das**, was in Ihrem Programm steht
- Er tut es sehr schnell

# Moore's "Gesetz"

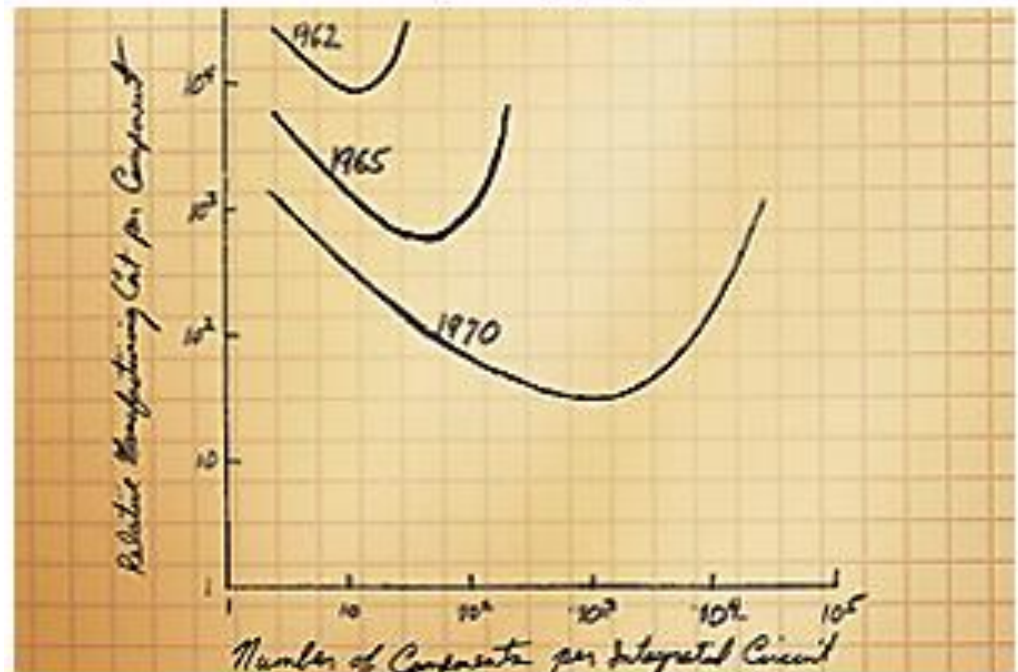


Etwa alle 18 Monate: Verdopplung der Rechenleistung bei gleichbleibendem Preis

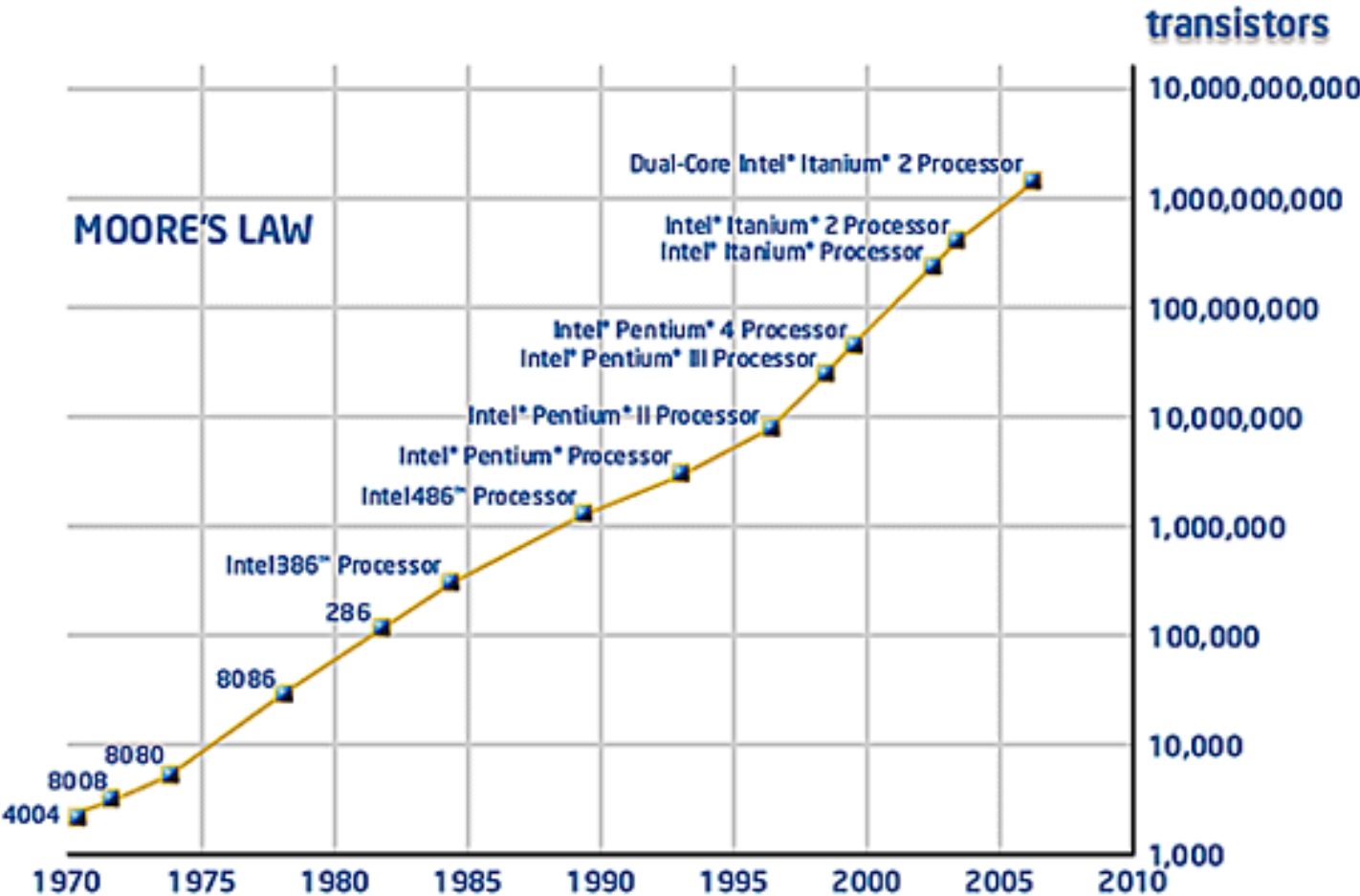
(Ist das die Aussage von Moore's Gesetz?)

(Nein: Verdopplung der Anzahl Transistoren)

Gordon Moore's original graph from 1965



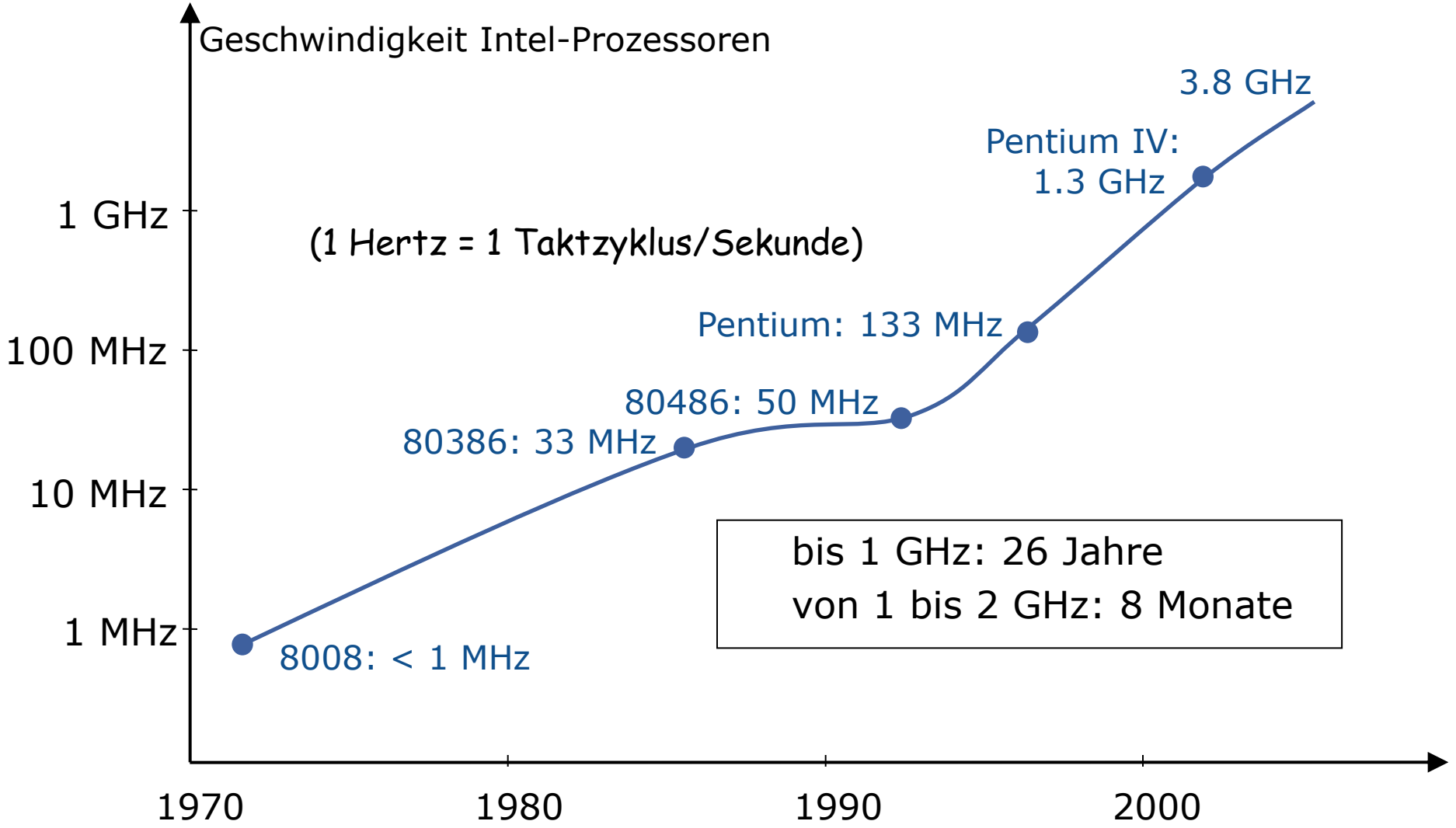
# Moore's Gesetz (Quelle: Intel)

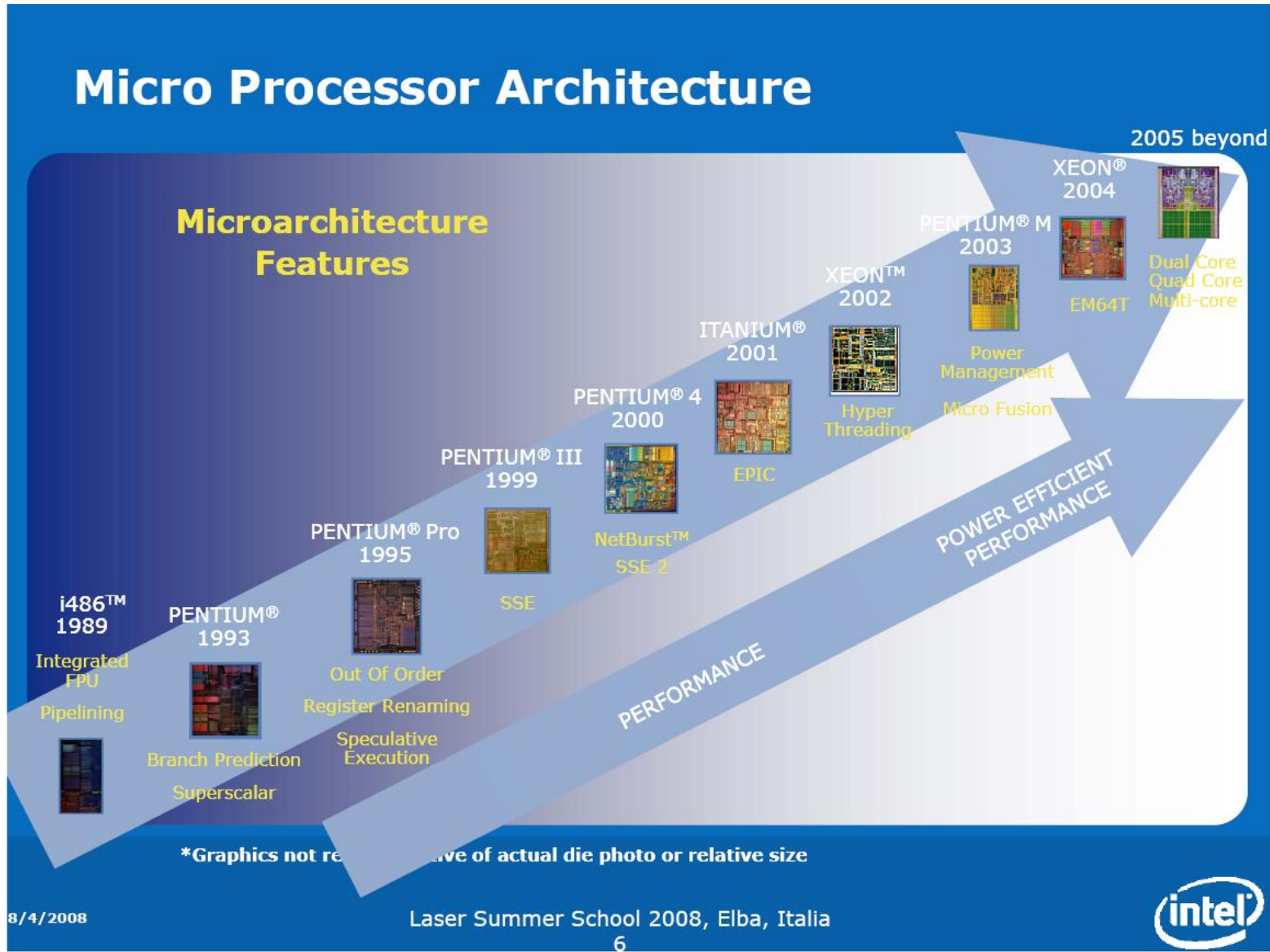


# Moore's "Gesetz"



Etwa alle 18 Monate: Verdopplung der Rechenleistung bei gleichbleibendem Preis







“Computer sind intelligent”

*Tatsache: Computer sind weder intelligent noch dumm. Sie führen Programme aus, die von Menschen entwickelt wurden. Diese Programme spiegeln die Intelligenz ihrer Autoren wieder. Die Grundoperationen eines Computers sind elementar (speichern eines Wertes, addieren zweier Zahlen...).*

“Der Computer ist abgestürzt”

“Der Computer erlaubt das nicht”

“Der Computer hat Ihre Datei verloren”

“Der Computer hat Ihre Datei kaputt gemacht”

# Computer machen keine Fehler \* ....

---



- Programme machen auch keine Fehler
- Programmierer **machen** die Fehler

\* Hardware kann zwar defekt sein, aber das ist viel seltener der Fall als Fehler in der Software





Computers sind universelle Maschinen. Sie führen das Programm aus, das Ihnen gegeben wird.

Die einzige Grenze ist Ihre Vorstellungskraft und Ihre Sorgfalt

Die guten Nachrichten:

- Ihr Computer tut genau das, was in Ihrem Programm steht
- Er tut es sehr schnell

Die schlechten Nachrichten:

- Ihr Computer tut genau das, was in Ihrem Programm steht
- Er tut es sehr schnell

*"To err is human, but to really mess things up takes a computer"*



**WARNING!**

The system is either busy or has become unstable. You can wait and see if it becomes available again, or you can restart your computer.

- \* Press any key to return to Windows and wait.
- \* Press CTRL+ALT+DEL again to restart your computer. You will lose unsaved information in any programs that are running.

Press any key to continue \_



Programme "stürzen ab"

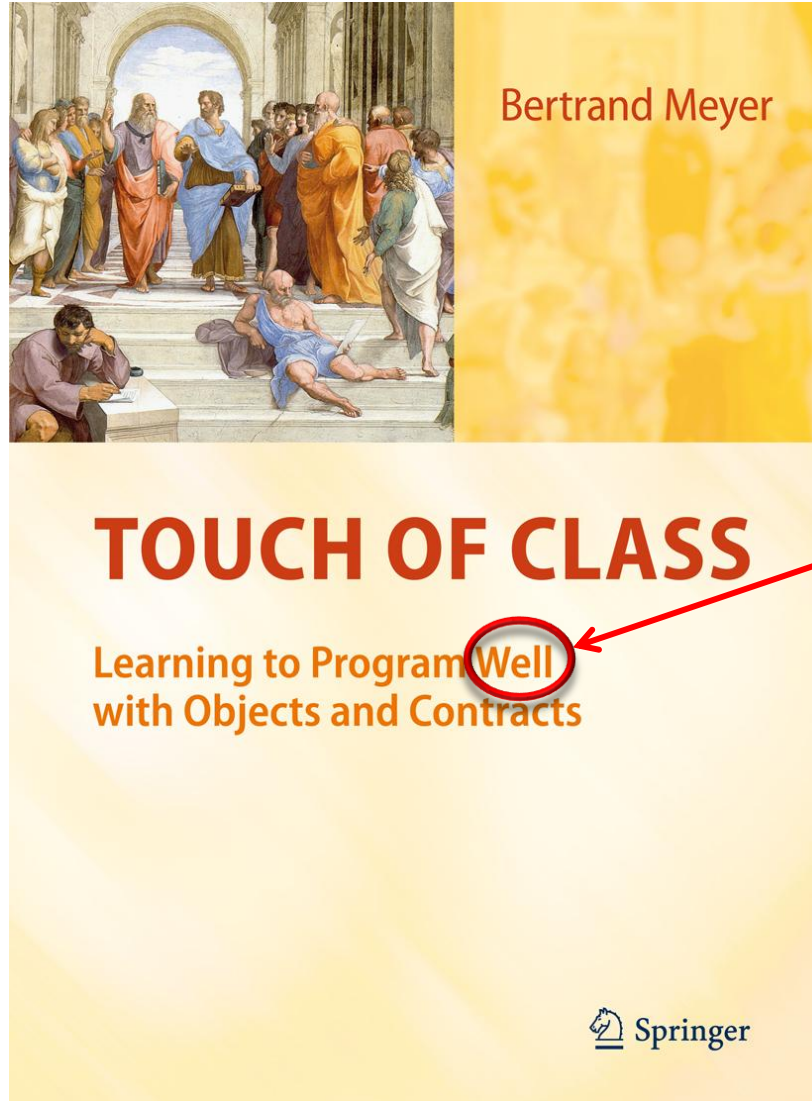
Wenn Programme nicht abstürzen, bedeutet das nicht unbedingt, dass sie richtig funktionieren

Fehlerhafte Programme haben Menschen umgebracht, z.B. medizinische Geräte

Ariane 5 Rakete, 1996: \$10 Mia. verloren, wegen einem einfachen Programmfehler

Programmierer sind für die korrekte Funktionsweise ihrer Programme verantwortlich

Der Zweck dieser Vorlesung ist es nicht nur Ihnen programmieren beizubringen, sondern dass Sie **gut** programmieren lernen.



## Vorlesungen:

- Montags, 13:15 - 15:00, HG E3
- Dienstags, 8:15 - 10:00, HG F1

## Übungsstunden:

- 10 Gruppen
  - Montags, 15:15 - 17:00, in verschiedenen Räumen
  - Dienstags, 13:15 - 15:00, in verschiedenen Räumen



Diese Information wurde Ihnen per E-Mail zugeschickt.  
Sie wählen zwischen

- Drei Niveaus
- Zwei Sprachen

Sie werden einer Gruppe zugewiesen.

Falls Sie gute Gründe für einen  
Gruppenwechsel haben: **fragen Sie Scott**

# Das Team der Assistenten

---



Nadia Polikarpova  
(Qualitätssicherung/  
Back Office)

Scott West  
(Koordinator)

Marco Piccioni

Marco Trudel

Valentin Wuestholz

Christian Estler

Yu Pei (Max)

Marco Piccioni

Alexandre Lagadec

Sandra Weber

Daniel Schweizer

Daniela Meier



Scott West







Nadia Polikarpova





Mailingliste: [se-info1-nygaard@lists.inf.ethz.ch](mailto:se-info1-nygaard@lists.inf.ethz.ch)

E-mail: [christian.estler@inf.ethz.ch](mailto:christian.estler@inf.ethz.ch)

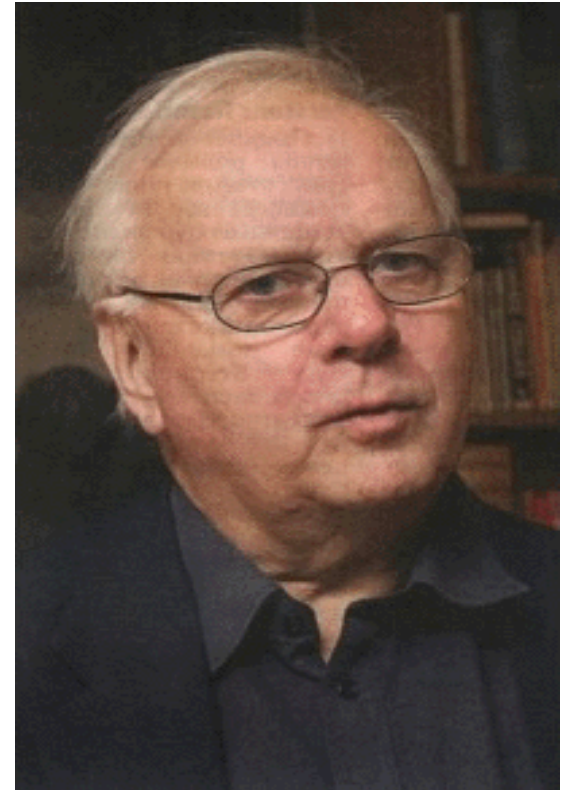
Büro: RZ J8

Telefon: 044 632 76 84

Sprache: Deutsch

Übungsraum:

- Dienstags: HG D1.2





Mailingliste: [se-info1-goldberg@lists.inf.ethz.ch](mailto:se-info1-goldberg@lists.inf.ethz.ch)

E-mail: [lagadeca@student.ethz.ch](mailto:lagadeca@student.ethz.ch)

Sprache: Deutsch

Übungsraum:

➤ Dienstags: CAB G52





Mailingliste: [se-info1-knuth@lists.inf.ethz.ch](mailto:se-info1-knuth@lists.inf.ethz.ch)

E-mail: [daniemei@student.ethz.ch](mailto:daniemei@student.ethz.ch)

Sprache: Deutsch

Übungsraum:

➤ Dienstags: ML F38



# Gruppe Edsger Dijkstra: Max (Yu) Pei

---



Mailingliste: [se-info1-dijkstra@lists.inf.ethz.ch](mailto:se-info1-dijkstra@lists.inf.ethz.ch)

E-mail: [yu.pei@inf.ethz.ch](mailto:yu.pei@inf.ethz.ch)

Sprache: Deutsch

Büro: RZ J3

Telefon: 044 632 89 02

Übungsraum:

➤ Montags: IFW B42





Mailingliste: [se-info1-wirth@lists.inf.ethz.ch](mailto:se-info1-wirth@lists.inf.ethz.ch)

E-mail: [marco.piccioni@inf.ethz.ch](mailto:marco.piccioni@inf.ethz.ch)

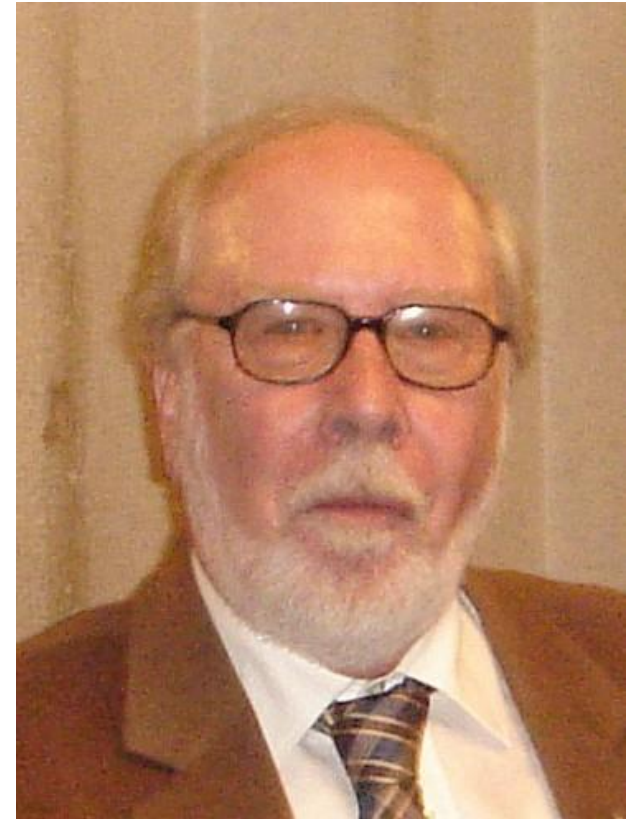
Büro: RZ J9

Telefon: 044 632 65 32

Sprache: Englisch

Übungsraum:

- Dienstags: IFW A34





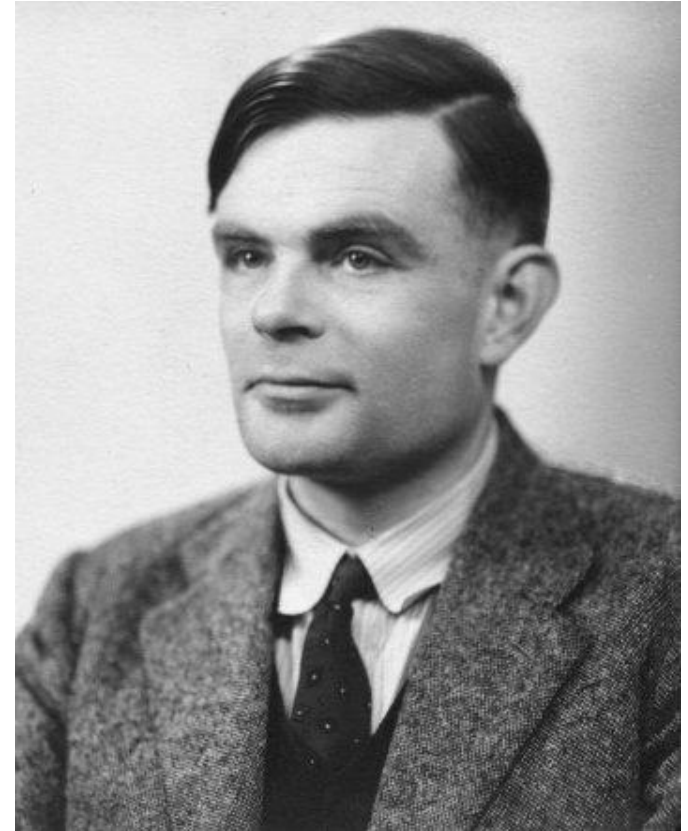
Mailingliste: [se-info1-turing@lists.inf.ethz.ch](mailto:se-info1-turing@lists.inf.ethz.ch)

E-mail: [daschwei@student.ethz.ch](mailto:daschwei@student.ethz.ch)

Sprache: Deutsch

Übungsraum:

- Dienstags: LFW C5



# Gruppe Ada Lovelace: Marco (Taco) Trudel



Mailingliste: [se-info1-lovelace@lists.inf.ethz.ch](mailto:se-info1-lovelace@lists.inf.ethz.ch)

E-mail: [marco.trudel@inf.ethz.ch](mailto:marco.trudel@inf.ethz.ch)

Sprache: Deutsch

Büro: RZ J9

Telefon: 044 632 85 92

Übungsraum:

- Montags: IFW A34







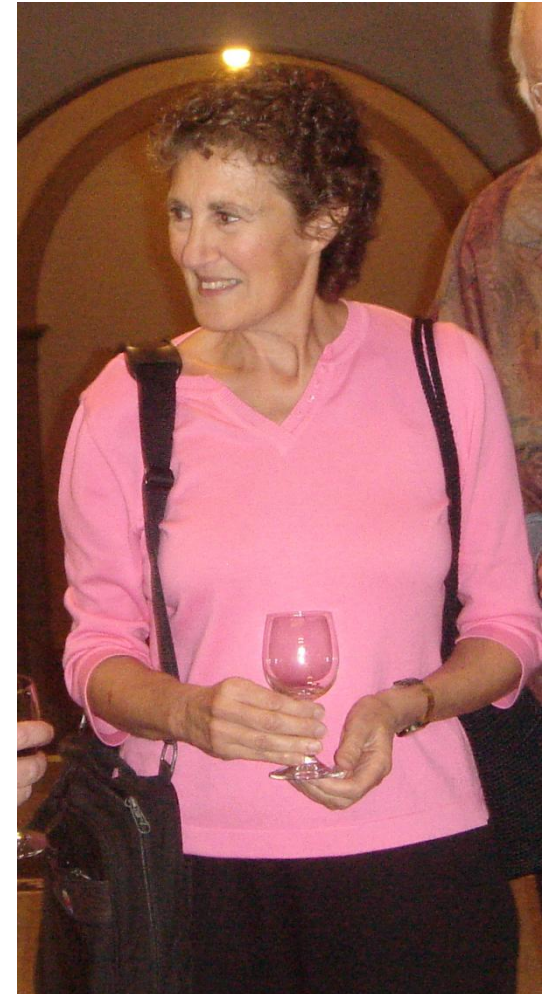
Mailingliste: [se-info1-liskov@lists.inf.ethz.ch](mailto:se-info1-liskov@lists.inf.ethz.ch)

E-mail: [s.weber@bluemail.ch](mailto:s.weber@bluemail.ch)

Sprache: Deutsch

Übungsraum:

- Montags: CHN D48





Mailingliste: [se-info1-torvalds@lists.inf.ethz.ch](mailto:se-info1-torvalds@lists.inf.ethz.ch)

E-mail: [scott.west@inf.ethz.ch](mailto:scott.west@inf.ethz.ch)

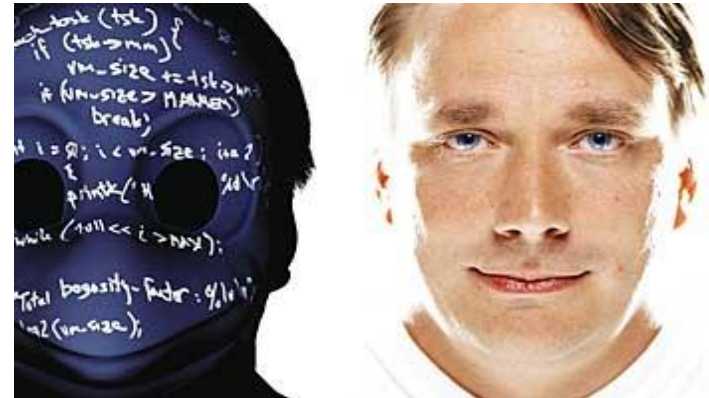
Sprache: Englisch

Büro: RZ J8

Telefon: 044 632 74 09

Übungsraum:

➤ Dienstags: IFW B42





Mailingliste: [se-info1-hoare@lists.inf.ethz.ch](mailto:se-info1-hoare@lists.inf.ethz.ch)

E-mail: [valentin.wuestholz@inf.ethz.ch](mailto:valentin.wuestholz@inf.ethz.ch)

Büro: RZ F9

Telefon: 044 632 79 42

Sprache: Deutsch

Übungsraum:

- Dienstags: IFW A32.1





An der ETH seit Ende 2001, Professor für Software Engineering  
Grossteil meiner Karriere in der Industrie, zuletzt bei *Eiffel Software* in  
Santa Barbara, Kalifornien, seit 1985. Heute "Chief Architect"

Assoc. Prof. an der University of California, Santa Barbara seit den 80ern  
Autor mehrerer Bücher, insbesondere *Object-Oriented Software  
Construction* (2. Ausgabe: 1997)

Forschungsschwerpunkte: Software Engineering, Methoden, Werkzeuge,  
Programmiersprachen, Objekt-orientierte Programmierung, Nebenläufige  
Programmierung, Programmbeweise, Testen, Entwicklungsumgebungen,  
Persistenz etc.

## Kontakt Daten:

- E-mail: [Bertrand.Meyer@inf.ethz.ch](mailto:Bertrand.Meyer@inf.ethz.ch), Büro: RZ J22
- Sekretariat: Claudia Günthart, 044 632 83 46  
[Claudia.Guenthart@inf.ethz.ch](mailto:Claudia.Guenthart@inf.ethz.ch), Büro: RZ J7

Sprechstunden: Donnerstags während des Semesters, kontaktieren Sie  
Frau Günthart



Webseite:

<http://se.ethz.ch/teaching/2010-H/eprog-0001/>

→ Zweimal wöchentlich anschauen

Deutsche Version ist vorhanden, aber die englische ist meistens aktueller

Vorlesungsunterlagen:

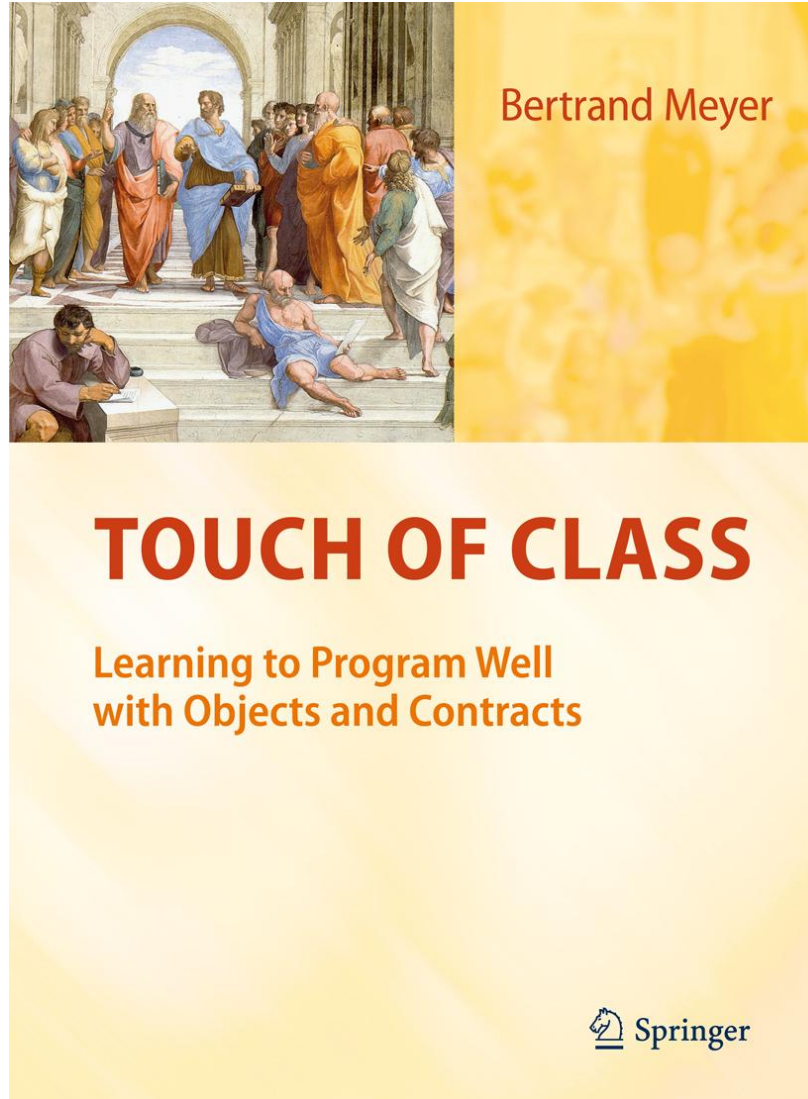
- Folien der Vorlesung
- Buch: *Touch of Class*  
Siehe nächste Folie

Übungsunterlagen:

- Übungen
- Musterlösungen

Auch vorhanden:

Videos der Vorlesung!





**ETH-Zentrum**  
MM B 96  
8092 Zürich

## Öffnungszeiten

**Montag – Donnerstag**  
**9:30 – 16:30 Uhr**  
**Freitag**  
**9:30 – 15:30 Uhr**

In der Buchhandlung akzeptieren wir  
Bargeld, Maestro, Postcard, Visa und Eurocard/Mastercard

**Meyer**  
**Touch of Class**  
**Fr. 58.00**

**Büchertisch vor dem Hörsaal**  
**Nur Barbezahlung möglich!**

Polybuchhandlung  
ETH Zentrum MM B96  
Öffnungszeiten: Mo-Do 9.30 – 16.30, Fr. 9.30-15.30  
Internet: [www.books.ethz.ch](http://www.books.ethz.ch), Email: [shop@books.ethz.ch](mailto:shop@books.ethz.ch)



Möglich aus dem Netz der ETH  
URL: siehe Vorlesungswebseite





## Diskussionsforen:

Hilfeforum für die gesamte Vorlesung:

<http://forum.vis.ethz.ch/>

Mailingliste für jede Übungsgruppe

## Ratschläge und Regeln:

- Benutzen Sie das VIS-Forum und die Mailinglisten! Programmieren zu lernen ist schwierig: Nutzen Sie jede Hilfe, die Ihnen angeboten wird.
- Es gibt keinen Grund, schüchtern zu sein. Es gibt keine dummen Fragen.
- Kritik ist willkommen, seien Sie aber immer freundlich und halten Sie sich an die **Etiquette**.

# Falls Sie einen Laptop brauchen...

---



Das NEPTUN-Programm der ETH verkauft Laptops zu guten Preisen

Thinkpad (Lenovo, ex-IBM), HP oder Apple

Sie wählen das Betriebssystem: Windows, Linux, MacOS

Zeitlich begrenzter Verkauf: siehe [www.neptun.ethz.ch](http://www.neptun.ethz.ch)

Die Übungen sind ein wichtiger Bestandteil der Vorlesung

- Ca. 10 wöchentliche Übungen
- Zwei Präsenzübungen

Für Ihre Übungsabgabe sollten Sie:

- nachweisen, dass Sie die Aufgaben zu lösen versucht haben.
- die **Umfrage** ausfüllen.

Absenzen wegen Militärdienst oder Krankheit:  
kontaktieren Sie Ihren Assistenten.



Die Grundregeln sind von der ETH diktiert, die Feinheiten für das Testat von uns bestimmt:

**Die Note beruht nur auf der Leistung in der Prüfung vom kommenden September**

Aber: damit Sie zur Prüfung zugelassen werden müssen Sie das Testat erhalten. Das bedeutet Sie müssen

- Alle wöchentlichen Übungen ausser einer abgeben
- An beiden Präsenzübungen teilnehmen

Damit diese akzeptiert werden, müssen Sie:

- nachweisen, dass Sie die Aufgaben zu lösen versucht haben.
- die **Umfrage** ausfüllen.

Absenzen wegen Militärdienst oder Krankheit: kontaktieren Sie Ihren Assistenten.

Repetenten: das Testat vom letzten Jahr ist gültig, aber wir empfehlen Ihnen die Übungen trotzdem zu machen.



Programme "stürzen ab"

Wenn Programme nicht abstürzen, bedeutet das nicht unbedingt, dass sie richtig funktionieren

Fehlerhafte Programme haben Menschen umgebracht, z.B. medizinische Geräte

Ariane 5 Rakete, 1996: \$10 Mia. verloren, wegen einem einfachen Programmfehler

Programmierer sind für die korrekte Funktionsweise ihrer Programme verantwortlich

Der Zweck dieser Vorlesung ist es nicht nur Ihnen programmieren beizubringen, sondern dass Sie **gut** programmieren lernen.



<http://www.youtube.com/watch?v=kYUrqdUyEpI>

Programm-Fehler: sehen

- Jean-Marc Jézéquel & Bertrand Meyer: *Design by Contract: The Lessons of Ariane*, in *Computer (IEEE)*, vol. 30, no. 1, January 1997, pages 129-130, [archive.eiffel.com/doc/manuals/technology/contract/ariane/](http://archive.eiffel.com/doc/manuals/technology/contract/ariane/).



Erfolgreich studieren (insbesondere an der ETH):

- Sie bestimmen selbst, was Sie wann tun
- Nutzen Sie die Möglichkeiten der ETH
  - Talks gehalten von Forschern von anderen Unis
  - Konferenzen
  - Bibliotheken
  - Experimente
  - Projekte
- Sprechen Sie mit Professoren und Assistenten
- Lesen Sie die Webseiten des Departements und des Chair of Software Engineering
- Halten Sie Ausschau nach Vorlesungen mit Projekten und anderen Möglichkeiten, individuell zu arbeiten



- Besuchen Sie die Vorlesungen
  - Besuchen Sie die Übungsstunden
  - Lesen und drucken Sie die Folien vor der Stunde
  - **Machen Sie sich Notizen**
  - Schliessen Sie sich zu Studiengruppen zusammen
  - Besuchen Sie auch Vorlesungen zu informatikfremden Themen, vor allem während den ersten zwei Jahren
  - Bereiten Sie sich früh auf die Prüfung vor
- 
- Bewahren Sie eine kritische, forschende Einstellung





Die Übungen bauen auf der Bibliothek **Traffic** auf

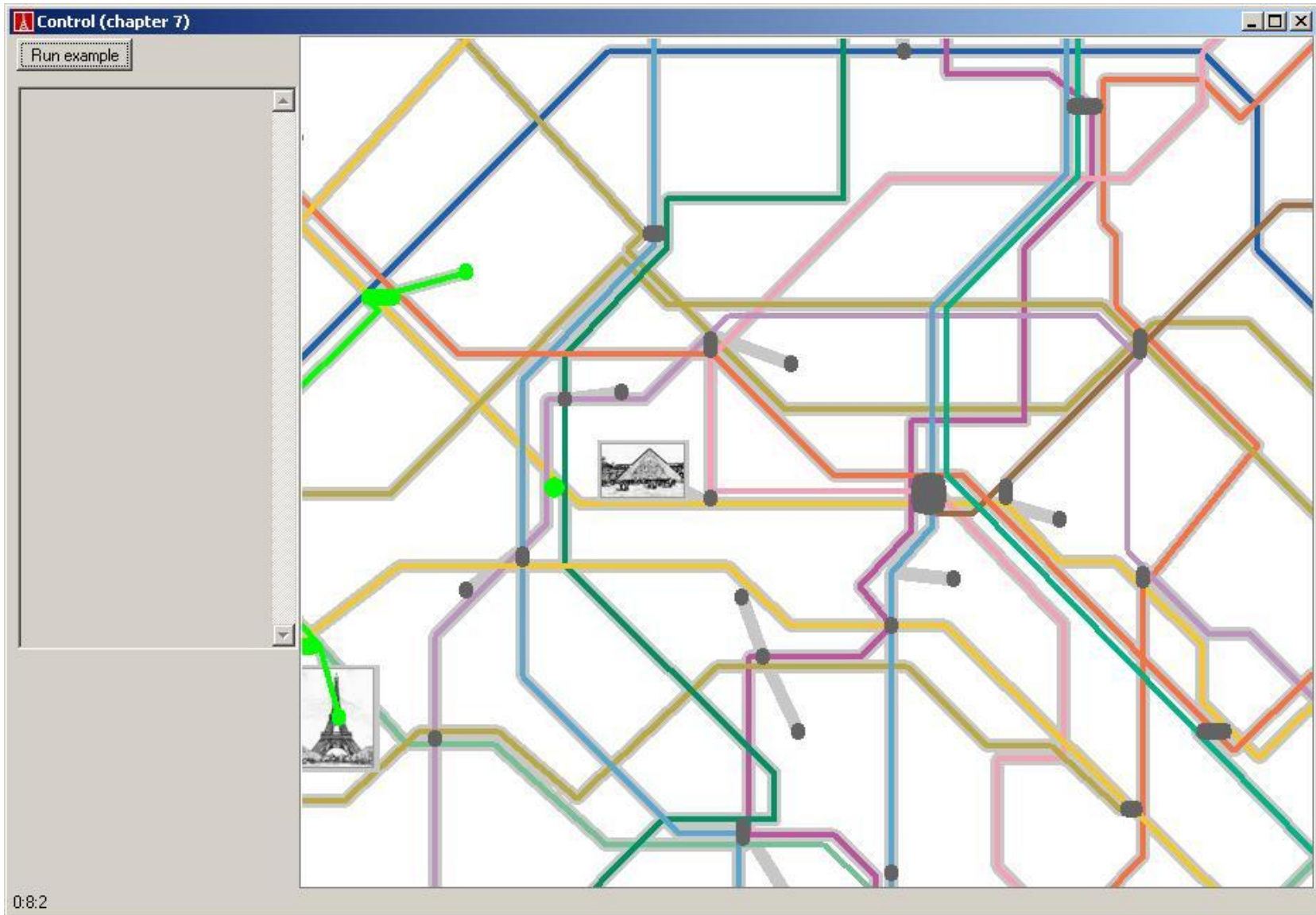
(150,000 Zeilen Eiffelcode)

Anwendungsgebiet: öffentlicher Verkehr in einer Stadt  
(benutzt Paris oder Zürich als Beispiel)

Sie müssen folgende Software herunterladen:

- EiffelStudio: <http://sourceforge.net/projects/eiffelstudio>
- Traffic: <http://traffic.origo.ethz.ch/download>

# Entdecken Sie Traffic





Michela Pedroni (ehemals Traffic-Projektleiterin)

Viele ETH studenten, unter anderem:

Marcel Kessler, Rolf Bruderer, Ursina Caluori,  
Roger Küng, Alan Fehr, Sarah Hauser, Michele Croci,  
Matthias Bühlmann, Florian Geldmacher, Susanne  
Kasper, Lars Krapf, Valentin Wüstholtz, Stefan Daniel,  
Etienne Reichenbach, Maria Husmann

...

Patrick Schönbach (erste Version)



Natürlich ist nicht alles perfekt.

Traffic beinhaltet wahrscheinlich Fehler ("bugs"), und das Buch wahrscheinlich auch.

(Fehlerliste: <http://touch.ethz.ch> -> *Errata*)

**Aber:**

- Wir versuchen, die Fehler so schnell wie möglich zu korrigieren.
- Schieben Sie beim Ausprobieren die Schuld jeweils nicht zuerst der Software in die Schuhe. Vielleicht folgt sie bloss Ihren Anweisungen.

# Weshalb diese Lehrmethode?

---



Viele Probleme verschlimmern sich bei **grossen** Systemen  
Mit anderen Lehrmethoden würden Sie nur mit kleineren  
Programmen arbeiten

Wir verwenden grosse Softwaresysteme; benutzen Sie  
diese als **Beispiele** und Inspiration

Sie benutzen die Software durch ihre **abstrakten**  
Schnittstellen (auch bekannt als **Verträge** (contracts))

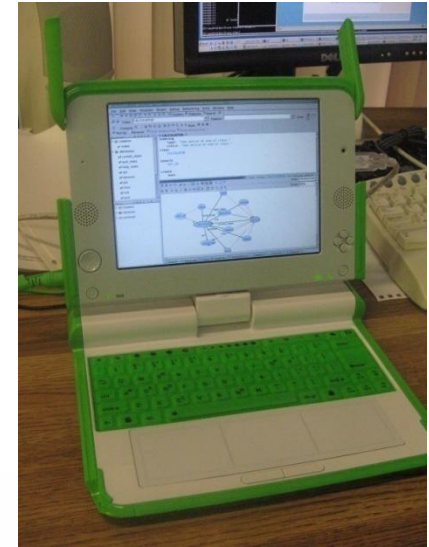
Sie verwandeln sich vom Konsumenten zum Produzenten:  
**outside-in**

Traffic ist grafisch und macht Spass!

Im besten Fall verstehen Sie am Ende die **gesamte**  
**Software**.

Dann können auch Sie **neues** hinzufügen

# Computer in allen Grössen, Farben und Formen



# Was Computer tun



## Speichern und wieder abrufen

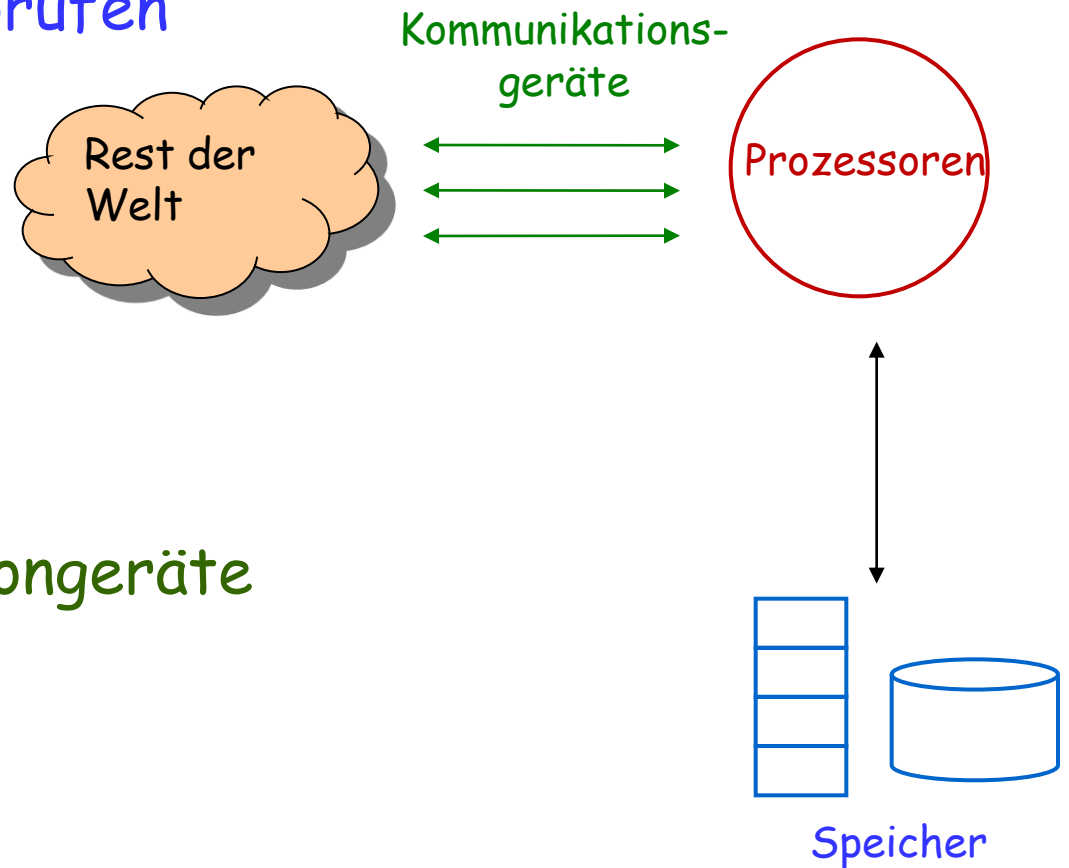
- Speicher

## Operationen ausführen

- Prozessoren

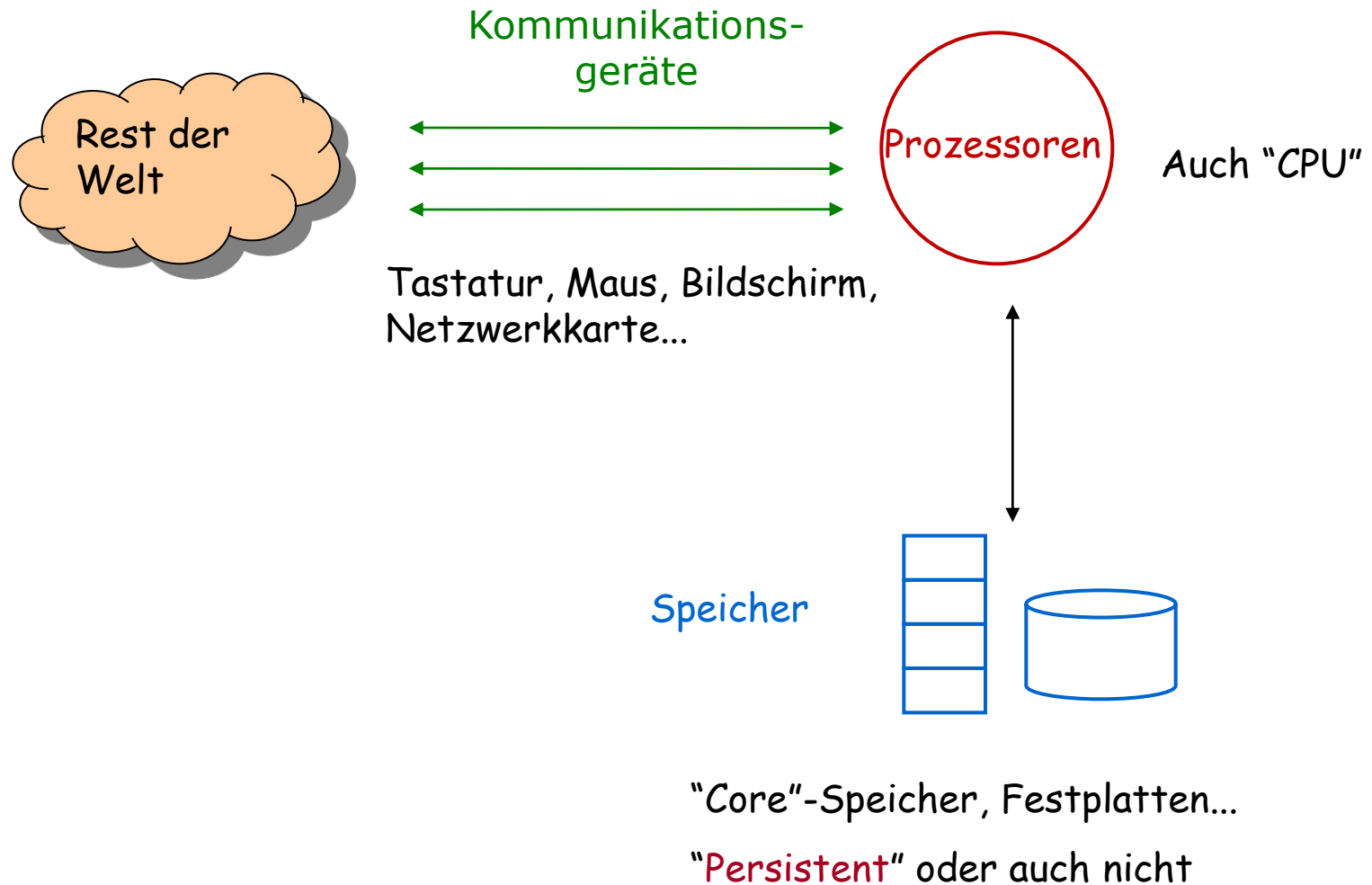
## Kommunikation

- Kommunikationsgeräte



Speicher, Prozessoren und Kommunikationsgeräte gehören zur Hardware

# Genereller Aufbau

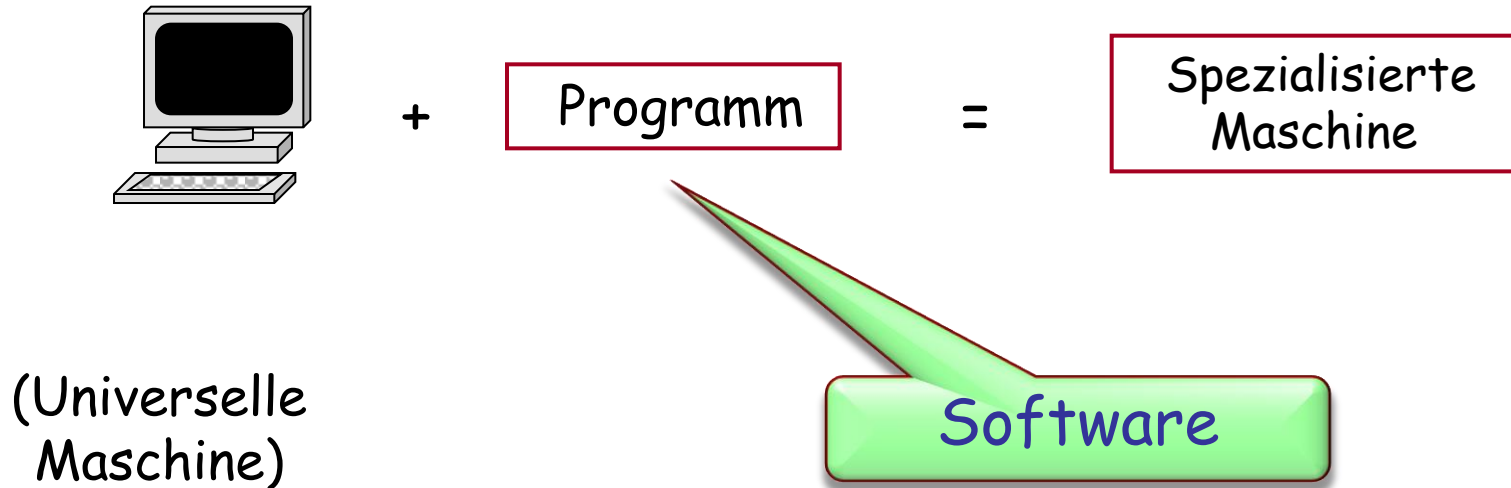




# Computer



Computer sind universelle Maschinen. Sie führen das Programm aus, das Ihnen gegeben wird.





Information ist, was Sie wollen, z.B. Text oder Musik

Daten sind Informatinen, die für den Computer kodiert sind, z.B. im MP3-Audioformat

- Daten: Menge von Symbolen auf dem Computer gespeichert
- Informationen: Interpretation der Daten für menschliche Zwecke

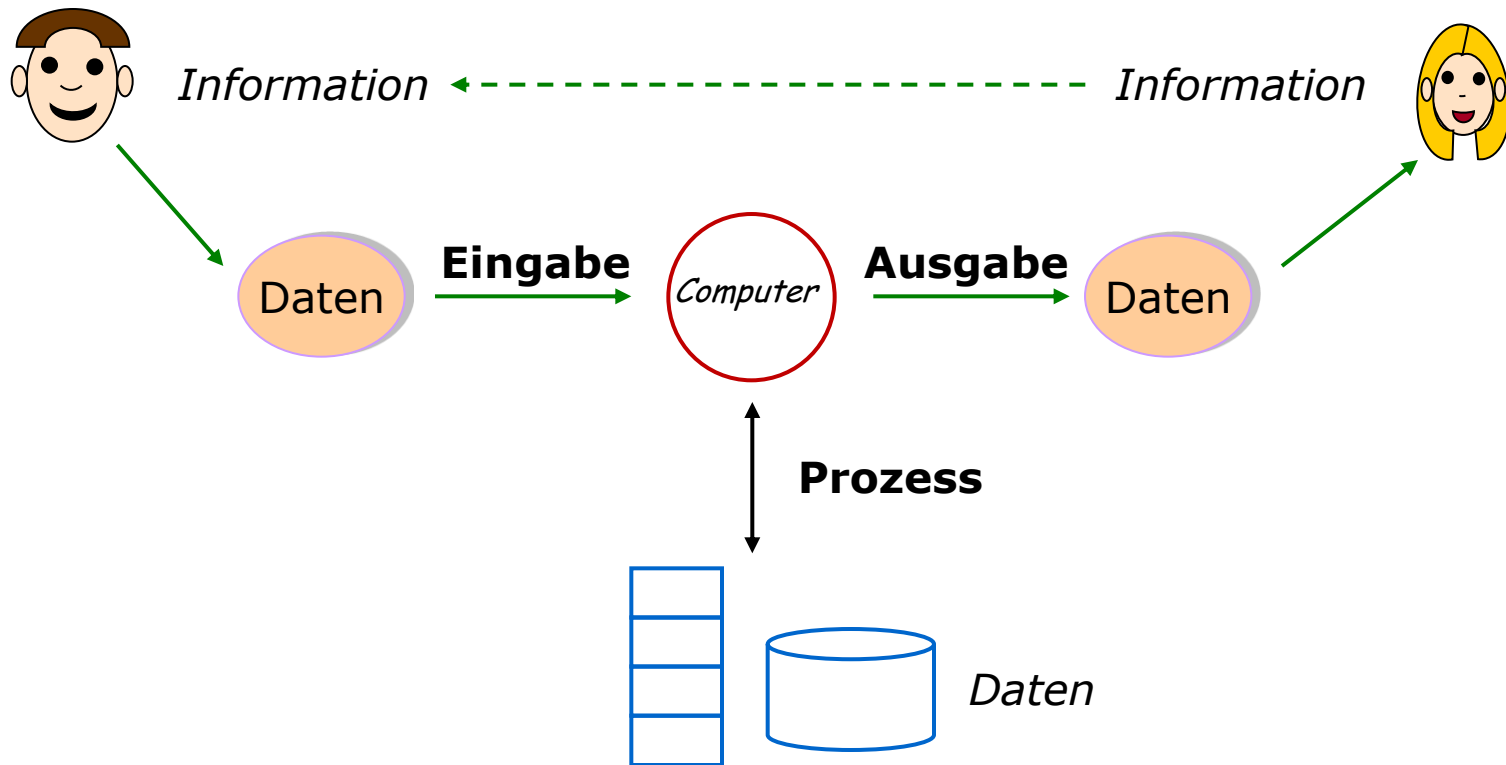
# Verarbeitung von Daten und Information



Daten werden im Speicher gehalten

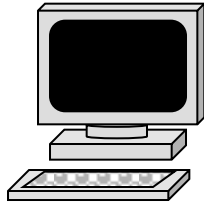
Eingabegeräte produzieren Daten aus Informationen

Ausgabegeräte produzieren Informationen aus Daten



# Wo ist das Programm?

---



+

Programm

=

Spezialisierte  
Maschine

(Universelle  
Maschine)

# Wo befindet sich ein Programm?

---



**Speicherprogrammierte Computer:** Das Programm befindet sich im Speicher

- "Executable data"

Ein Programm kann im Speicher in verschiedenen Formen vorhanden sein:

- **Quellcode:** von Menschen lesbar (Programmiersprache)
- **Maschinencode:** vom Computer ausführbar

Ein **Compiler** übersetzt Quellcode in Maschinencode

Der Computer

(genauer: die Plattform aus Computer + **Betriebssystem**)  
findet Ihr Programm im Speicher und führt es aus



Schreiben von Software, die folgende Eigenschaften besitzt:

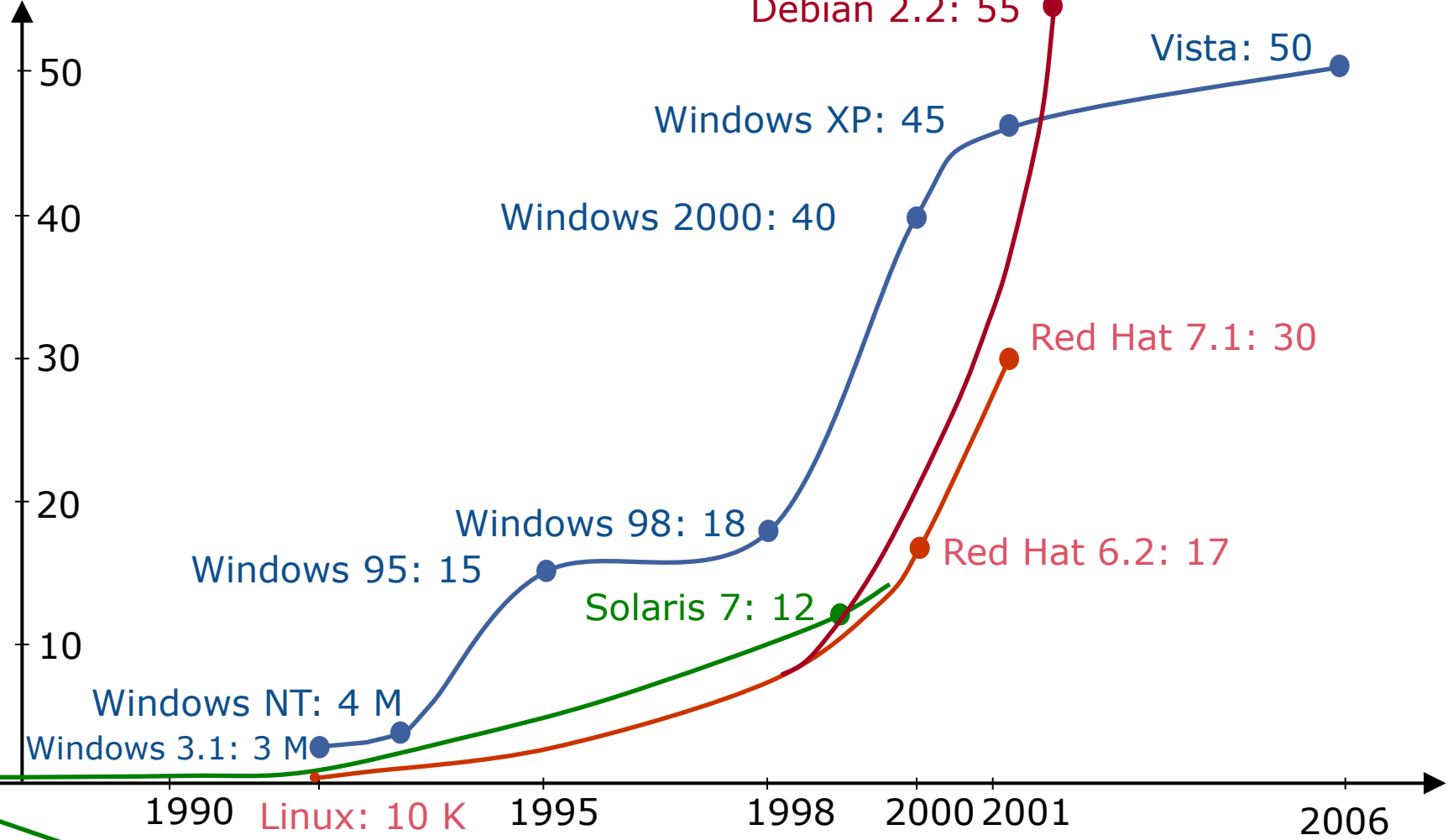
- Korrekt  
Macht was sie soll!
- Erweiterbar  
Einfach zu ändern!
- Lesbar  
Von Menschen!
- Wiederverwendbar  
Das Rad nicht wiedererfinden!
- Robust  
Reagiert angemessen auf Fehler
- Sicher  
Gegen Angreifer

# Betriebssysteme: Quellcodeumfang



Debian 3.1: 213!

Anzahl Codezeilen (Mio.)



Unix V7: 10K

# Software zu schreiben ist schwierig

---



Es ist schwierig, ein korrektes Programm zu schreiben

“Trial-and-error“ ist sehr ineffizient



# Software zu schreiben macht Spass

---



Entwerfen und entwickeln Sie Ihre eigenen Maschinen

Leben Sie Ihre Kreativität und Ihren Ideenreichtum aus

Programme retten Leben und helfen, die Welt zu einem besseren Ort zu machen

Erleben Sie das Gefühl, wenn ein Programm, das Sie geschrieben haben, funktioniert



Lesen Sie Kapitel 1 und 2 von *Touch of Class*

Schauen Sie sich die Folien der nächsten zwei Vorlesungen (2 und 3) an