

Informatik-I

Einführung in das Programmieren
Technische Grundlagen der Informatik

Harald Gall, Burkhard Stiller
Institut für Informatik
seal.ifi.uzh.ch/info1

 University of Zurich
Department of Informatics



Ziele dieser LV

- Die wichtigsten **Konzepte des Programmierens** kennen
- Die erste **Programmiersprache** beherrschen: **Java**
- Grundlegende Konzepte von Design, Implementation und Evolution von **Software-Systemen** kennen
- Software und Hardware näher verstehen
- **Computer Architekturen**, Rechnerstrukturen, Rechnerorganisation kennen
- **Grundlagen** fürs Informatik-Studium

2

Praktische Informationen

 University of Zurich
Department of Informatics



Organisation

Dozierende	Prof. Harald Gall	Prof. Burkhard Stiller
Assistierende	Michael Würsch	Fabio Hecht
Tutoren	Teaching Assistants	
Zeit und Ort	Di 12-14 und Mi 12-15	
AP/ECTS	9	
Anmeldung	1. Modulanmeldung via Assessment-Buchung 2. Anmeldung in eine Übungsgruppe via OLAT	
Übung	Details siehe Web-Seite der Übungen	

4

Unterlagen zur LV

- Web-Seite für Vorlesung und Übungen
<http://seal.ifi.uzh.ch/info1/>
- Vorlesungsmaterial:
 - Vorlesungsfolien und 2 Bücher:
 - Walter Savitch, Frank M. Carrano, *Java: An Introduction to Problem Solving & Programming*, 5/e, Pearson, 2008
 - H. Herold, B. Lurz, J. Wohrab, *Grundlagen der Informatik*, Pearson, 2007
- Übungsmaterial:
 - Übungsblätter, Beispiel-Lösungen




5

Übungen

- Wöchentliche Übungen
- Lösung und Besprechung von Übungsbeispielen mit Teaching Assistants
- Ablegen von Zwischentests
- Übungen und Tests am Computer

6

Das Team

- Professoren
 - Harald Gall (EPROG)
 - Burkhard Stiller (TGI)
- Assistenten
 - Michael Würsch (EPROG)
 - Fabio Hecht, Martin Waldburger (TGI)
- Teaching Assistants & Tutoren
 - Markus Cadonau, Yves Bilgerig
 - Stefan Hiltbrand, Johanna Gaudenz, Manuel Gugger, Christoph Moser, Marc Tobler, Pascal Muther

7

Anmeldung

- Web-Seite der LV genau studieren und entsprechend anmelden
- 2 Anmeldungen erforderlich
 - Für das Modul „Informatik-I“ (ECTS Punkte)
 - und für die Übung via OLAT (Übungsgruppe und -zeit)
- Mathematiker buchen „Programmierung für Mathematik“

8

Übungen

- Wöchentliche Übungen
- Einzelarbeit in Übungsgruppen
- Zwischentests
- Endklausur = Assessmentprüfung
- Übungsstunden mit Teaching Assistants/Tutoren
 - für Fragen,
 - Vor- und Nachbesprechungen der Beispiele,
 - sowie für zusätzliche Beispiele
- Tipp: Nutzen Sie die Übungsstunden aktiv!

9

Computer und Software



Software Ingenieure bauen Maschinen

- Unsere Maschinen kann man nicht berühren – sie sind immateriell
- Sie sind aber trotzdem **Maschinen**
- Wir nennen sie **Programme** oder Systeme

- Um ein Programm auszuführen braucht man eine physikalische Maschine: einen **Computer**
- Computer und Peripherie: **Hardware**
- Programme und der damit verbundene Intellektuelle Wert: **Software**

(c) Bertrand Meyer, ETH 11

Software überall

- Banken: Verwalten Millionen von Konten
- Handel: Entscheidet über Kauf oder Verkauf
- Transport: Kontrolliert Züge, Flugzeuge...
 - Moderne Autos enthalten in ihren Systemen Millionen Zeilen von Programm-Code
- Reisen: Luftfahrt, Eisenbahn, Hotel-Reservationen
- Staat: Steuerverwaltung, Verwaltung von Gesetzen...
- Medizin: Patientenverwaltung
- Schule
- Unterhaltung
- Information
- etc.

12

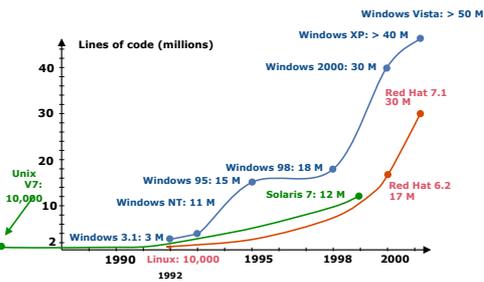
Computer

- Computer sind universelle Maschinen. Sie führen das Programm, das man ihnen eingibt, aus.
- Die Vorstellungskraft ist die einzige Grenze
- Die guten Neuigkeiten:
 - Der Computer wird **genau** das tun, was das Programm sagt

(c) Bertrand Meyer, ETH

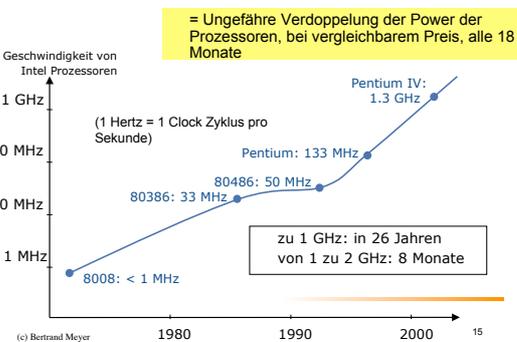
13

Size of Operating Systems (LOC)



14

Moore's "Law"

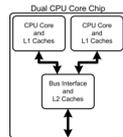


(c) Bertrand Meyer

15

Von NetBurst zu Multi-Core

- Taktfrequenz-Wettläufen Ende der 1990er
 - AMD Ur-Athlon vs. Intel (hohe Taktfrequenzen - und geringere Effizienz)
 - Pentium 4 (1,5 GHz in 2000, 4 GHz in 2002)
 - 2004 Ende vom NetBurst - stattdessen hohe Rechenkraft *pro* Taktzyklus
 - Hyper-Threading
 - Dual-Kern-Prozessoren ("Conroe"-Kern, bis 2.93 GHz)
 - Intel Xeon vs. AMD Opteron
 - Quad-Kern-Prozessoren



Quelle: c't 16/2006

16

Allgegenwärtige Vorurteile & Ausreden

- "Computer sind intelligent"
 - Fakt: Computer sind weder intelligent noch dumm. Sie führen nur Programme aus, die von Menschen geschrieben werden. Diese Programme reflektieren die Intelligenz ihrer Autoren.*
 - Die Grundoperationen eines Computers sind sehr elementar (speichere diesen Wert, addiere diese beiden Zahlen).*
- "Der Computer ist abgestürzt"
- "Der Computer erlaubt das nicht"
- "Der Computer hat die Datei verloren"
- "Der Computer hat die Datei kaputt gemacht"

17

Computer machen keine Fehler*

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

- * In der Tat: Hardware kann Fehler haben, aber das ist viel seltener als Programm-Fehler

18

Computer

- Computer sind universelle Maschinen. Sie führen das Programm, das man ihnen eingibt, aus.
- Die Vorstellungskraft ist die einzige Grenze
- Die guten Neuigkeiten:
 - Der Computer wird *genau* das tun, was das Programm sagt
 - Er wird es sehr schnell tun
- Die schlechten Neuigkeiten:
 - Der Computer wird *genau das* tun, was das Programm sagt
 - Er wird es *sehr schnell* tun

19

Software schreiben ist schwer

- Programme *stürzen ab*
- Programme, die *nicht abstürzen*, funktionieren aber auch nicht unbedingt richtig
- Schlecht funktionierende Programme haben bereits Opfer gefordert (Medizinbereich)
 - Ariane 5, am 4. Juni 1996: Millionen Dollar Schaden wegen einem einfachen Programmfehler (Konvertierung 64-bit float auf 16-bit integer)
- **Programmierer** sind verantwortlich fürs korrekte Funktionieren ihrer Programme
- Das Ziel diese Kurses ist nicht Ihnen Programmieren beizubringen, das Ziel ist dass jede/r *gut programmieren* lernt!

20

Wo ist das Programm?

- **Gespeichertes Programm**: das Programm ist im Speicher. "Ausführbare Daten"
- Der Computer kann, mit Hilfe von einigen grundlegenden Programmen (**dem Betriebssystem**) das Programm im Speicher finden und es ausführen
- Ein Programm kann in verschiedenen Formen im Speicher erscheinen:
 - **Source Code**: in menschen-lesbarer Form (in der Programmiersprache geschrieben)
 - **Ziel Form, Maschinen Code, Objekt Form**: vom Computer ausführbare Form
- **Compiler** verwandeln Quelltext in Maschinen-Code

(c) Bertrand Meyer

21

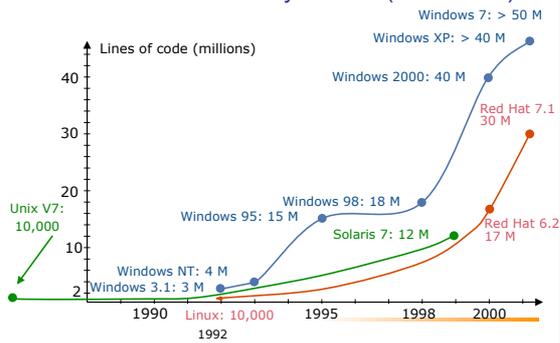
Software Engineering

Software, die

- korrekt ist, tut was sie soll!
- erweiterbar ist, ist einfach zu ändern!
- lesbar ist, kann von anderen Programmierern einfacher verstanden werden!
- wiederverwendbar ist, muss nicht das Rad neu erfinden!
- robust ist, reagiert angemessen auf Fehler!

22

Grösse von Betriebssystemen (Quellcode)



(c) Bertrand Meyer, ETH

23

Software schreiben ist schwierig

- Umsetzen eines Problems in Anweisungen für die Maschine
- Verarbeiten der Informationen als Daten
- Berechnungen durchführen
- Spezifizieren, Codieren, Testen, Debuggen

24

Software schreiben macht Spass

Man designed und baut **eigene** Maschinen

Man kann **kreativ** sein und seine Vorstellungskraft gebrauchen

Die Erfahrung des **Gefühls** wenn eines der eigenen Programme läuft und funktioniert

25

Ausblick

- Konzepte, Algorithmen
- Modelle, Programme
- Java

26

FAQ - Frequently Asked Questions

- Fragen zum Informatik Studium
 - <http://www.ifi.uzh.ch/teaching/faq/>
- Fragen zu Studium, Organisation, UZH
 - <http://www.uzh.ch/studium/faq/>

27
