

Vorlesung

# Grundlagen der Informatik für Ingenieure I

G. Bolch, C.-U. Linster, F.-X. Wurm

Informatik 4

SS 2004

## 1 Einführung und Informationen zur Vorlesung

# Grundlagen der Informatik für Ingenieure I

## 1 Einführung und Informationen zur Vorlesung

- 1.1 Hörerkreis und Quiz
- 1.2 Struktur der Lehrveranstaltung
- 1.3 Überblick Programmiersprachen
- 1.4 Objektorientierte Programmierung
- 1.5 Java
- 1.6 Vorlesungsunterlagen/ Literatur
- 1.7 zu den Übungen
- 1.8 Prüfungen
- 1.9 Termine

# 1.1 Hörerkreis und Quiz

**Für welche Studienrichtungen ist diese Vorlesung geeignet?:**

WW, MB, W-Ing (jeweils neue Prüfungsordnung), CIW (auf freiwilliger Basis), sonstige?

Nicht geeignet (wg. Prüfungsord.) für: E-Technik, Mechatronik

**Wer hat schon einmal ein Programm mit mehr als 50 Zeilen Code geschrieben?**

Ja:        Nein:

**Wenn ja, welche Sprache?**

Pascal:        Fortran:        C:        C++:        Java:        sonstige:

**Wer hat sich schon einmal Informationen über das Internet beschafft?**

Ja:        Nein:

## 1.2 Struktur der Lehrveranstaltung

### ■ Motivation:

- ◆ In allen Ingenieurwissenschaften ist fundiertes Informatikwissen unabdingbare Voraussetzung.
- ◆ "Elektronik-/Informatik - Wertschöpfung" bei einem modernen KFZ bis zu 30%, über 50% spätestens 2010 (Drive by Wire).
- ◆ Lehrveranstaltung Grundlagen der Informatik (Gdi) 6h (für MB 7h):
  - vom Umfang her nur eine einführende Lehrveranstaltung.
  - 6 SWS entsprechen etwa 4% Gesamtstundenzahl des Studiums.
  - Schlußfolgerungen?

## 1.2 Struktur der Lehrveranstaltung

---

### ■ Zum Inhalt:

- ◆ Das Grundkonzept der Vorlesung:  
Den Hörern soll die Fähigkeit vermittelt werden, Problemstellungen der Informatik im Umfeld technischer Systeme zu lösen.
- ◆ Dabei wird als "roter Faden" die Programmierung technischer Systeme als zentrales Thema behandelt.
- ◆ Es werden immer dann, wenn es der Stoff erfordert, Kenntnisse aus den Fachgebieten Rechnerarchitektur, Betriebssysteme, Verteilte Systeme, Rechnernetze, Compiler-technik, usw. vermittelt.

## 1.2 Struktur der Lehrveranstaltung

---

- ◆ **Gliederung der Vorlesung** (Reihenfolgen können sich ändern):
  - Einführung und Informationen zur Vorlesung
  - Einführung in Java
  - Umgang mit UNIX
  - Java-Sprachkonstrukte
  - Klassen, Objekte, Methoden
  - Java-Applets
  - Einfache Graphik
  - AWT Abstract Windows Toolkit Teil 1
  - Oberflächen mit Swing
  - Programming in the Large/Small
  - Interaktivität und Ereignisbehandlung (Eventhandling)

## 1.2 Struktur der Lehrveranstaltung

---

### ◆ Gliederung der Vorlesung (cont):

- Programm- und Datenstrukturen
- Ausnahmenbehandlung, Fehlersituationen (Exceptionhandling)
- Streams, I/O (Java-Ein-/Ausgabesystem)
- Aktivitätsträger (Prozesse/Threads)
- Images, Animation, Sound
- Verteilte und Parallele Programmsysteme (Einführung)
- AWT Abstract Windows Toolkit Teil 2
- Programmiersprachen im Vergleich
- Ausblick und Repetitorium

## 1.3 Überblick Programmiersprachen

---

### ■ Mögliche Gliederung

#### ◆ Maschinensprachen

- Prozessorspezifisch,
- binär, oktal oder hexadezimal

#### ◆ Maschinenorientierte Programmiersprachen

- Prozessorspezifische sog. Assembler Sprachen
- werden durch Assembler in Maschinensprache übersetzt

#### ◆ Problemorientierte (höhere) Programmiersprachen

- werden durch Compiler in Assemblersprache oder direkt in Maschinensprache übersetzt

## 1.3 Überblick Programmiersprachen

- ◆ Problemorientierte (höhere) Programmiersprachen (cont):
  - Technisch/Wissenschaftlich
    - C (C++) - insbesondere Systemprogrammierung
    - Fortran90/95 - insbesondere numerische Anwendungen
    - Java - in Netzwerken (Internet) und für grafische Oberflächen, *embedded systems*, architekturunabhängige Softwaresysteme
    - Algol, Pascal, Modula, Basic - kommerziell eher unbedeutend
  - Betriebswirtschaftlich: Cobol,..
  - Datenbanken: SQL, Natural,..
  - Echtzeitsysteme (Realtime): PEARL, ADA, ..
  - Wissensbasierte Systeme (KI): Lisp, Prolog
  - Dokumentbeschreibung: Postscript, HTML, ..

## 1.3 Überblick Programmiersprachen

### ■ Andere Unterscheidungskriterien:

- ◆ objektorientiert - nicht objektorientiert
  - objektorientiert: C++, Smalltalk, Java, C#, ...
    - Programmstruktur orientiert sich an einzelnen Objekten (Dreieck, Konto, ...)
  - nicht objektorientiert: Pascal, C, FORTRAN, ...
    - Programmstruktur orientiert sich an Unterprogrammen (Aktivitäten)
- ◆ direkte Abarbeitung - interpretative Abarbeitung
  - direkte Abarbeitung: C, C++, Pascal, ...
    - Programm wird erst in Maschinensprache übersetzt und dann abgearbeitet
    - Übersetzung und Abarbeitung getrennt
  - interpretative Abarbeitung: Java (teilweise), Basic, ....
    - Jede Zeile des Programms übersetzt und dann abgearbeitet.
    - Übersetzung und Abarbeitung gemeinsam

## 1.4 Objektorientierte Programmierung

### ■ Objektorientiert, warum?

- ◆ Die objektorientierte Programmierung (OOP) entspricht der natürlichen Denkweise.
- ◆ Ein Programm besteht aus einzelnen abgeschlossenen Einheiten, den sog. Objekten, mit definierten Schnittstellen nach außen.
- ◆ Beispiele für Objekte sind:
  - Dreieck
  - Konto
  - Fahrzeug
  - Roboter
- ◆ Objektorientierte Programme benötigen zum Teil erheblich mehr Rechenzeit
- ◆ Wegen der modernen leistungsfähigen Prozessoren setzt sich die OOP immer mehr durch

## 1.4 Objektorientierte Programmierung

### ■ Vorteile der OOP?

- ◆ wohldefinierte Schnittstellen
- ◆ bessere Programmstrukturen
- ◆ gekapselte Module, Baukastenkomponenten
- ◆ Wiederverwendbarkeit (ohne Kenntnis der Implementierung)
- ◆ einfachere Softwarewartung und -erweiterung
- ◆ Beherrschbarkeit komplexer Systeme
- ◆ Verteilte Programmsysteme

## 1.5 Java

### ■ Java, warum?

- ◆ Java ist eine "Weiterentwicklung" von C++ (ebenfalls eine objektorientierte Sprache)
- ◆ Ziel, eine möglichst einfache Sprache für den sogenannten "embedded" Bereich zu schaffen (von der Robotersteuerung bis hin zur Waschmaschine oder zum Mobiltelefon).
- ◆ Java ist "plattformunabhängig", d.h. unabhängig von der jeweiligen Hardware.
- ◆ Plattformunabhängigkeit durch einheitliche hardwareunabhängige Zwischensprache (Bytecode)
- ◆ Javaprogramm wird erst in Bytecode übersetzt.
- ◆ Programm in Bytecode wird durch einen prozessorspezifischen "Interpreter" abgearbeitet.

## 1.5 Java

### ■ Java, warum? (cont)

- ◆ Für Java gilt daher der Werbeslogan:  
**Write once - Run everywhere**
- ◆ Java eignet sich besonders zur Lösung von Aufgabenstellungen im Internetbereich.
- ◆ Darüberhinaus eignet sich Java besonders auch zur Lösung von Sicherheitsproblemen im Internet.
- ◆ Verwendete Programmiersprache eher zweitrangig.
- ◆ Wichtig ist das Erlernen der Programmieretechnik als Handwerkszeug.
- ◆ Die grundlegenden Sprachelemente sind in vielen Sprachen ähnlich.
- ◆ Beherrscht man grundsätzlich die "Kunst des Programmierens", so sollte die Sprache aufgrund der jeweiligen Anwendung ausgewählt werden.

## 1.6 Vorlesungsunterlagen

### ■ Vorlesungsunterlagen:

- Vortragsfolien werden im Internet spätestens 1 Woche vor ihrer Behandlung zur Verfügung gestellt.
- Java - Bücher gibt es wie Sand am Meer!  
Die Vorlesung orientiert sich zum Teil an:
  - Teach Yourself Java 2 in 21 Days, Laura Lemay u. a., Sams-Publishing, 2001, ISBN: 0-672-3206-4
  - The Java-Tutorial  
<http://www4.informatik.uni-erlangen.de/Services/Doc/Java/tutorial/>
  - auch als Buch erhältlich: The Java-Tutorial, JavaSoft Addison-Wesley-Verlag
- Preiswerte Einführung in Java:
  - Java 2 - Grundlagen und Einführung, RRZN Uni Hannover, 2004, 410 Seiten, 5.90 Euro, erhältlich im RRZN Martensstr.1
- Sprachdefinition und Informationen zur aktuellen Entwicklung:
  - <http://www4.informatik.uni-erlangen.de/Services/Doc/Java>

## 1.7 Zu den Übungen

### ■ Zu den Übungen:

- ◆ 3 SWS Übungen, davon 2 betreute Übungen am Rechner
- ◆ Der Rest: freies Üben
- ◆ Learning by Doing!
- ◆ Programmieren kann man nicht "lesend" lernen!



## 1.7 Zu den Übungen

---

### ■ Übungsablauf

- ◆ Kleinere bis mittlere, z. T. in vorgegebene Rahmenprogramme einzubettende Aufgaben.
- ◆ 3 bis 4 größere Hausaufgaben zum Abgeben.
- ◆ Erlaubt sind Zweiergruppen.
- ◆ Die Hausaufgaben sind Voraussetzung für die Zulassung zur Abschlussprüfung für den Schein.

## 1.8 Prüfungen

---

- ◆ Abschlussprüfung für den Schein:
  - Um den Übungsschein zu erhalten, müssen Sie eine praktische Aufgabe am Rechner lösen (3h).
  - Der Schein ist Voraussetzung für die Zulassung zur Vordiplom-Prüfung.
- ◆ Vordiplom-Prüfung:
  - 90 Minuten Klausur

## 1.9 Termine

---

■ **Termine** (bis auf weiteres):

- ◆ Vorlesung            Dienstag 14:15 bis 15:45 H7  
                                 Mittwoch 14:15 bis 15:00 H7

- ◆ Übungen: 02.151 (2. Stock Inf.-Hochhaus)

- betreute:            Mittwoch:    8:00 - 10:00, 10:00 - 12:00, 12:00 - 14:00  
                                 Donnerstag: 8:00 - 10:00, 10:00 - 12:00, 12:00 - 14:00  
                                 Freitag:      8:00 - 10:00, 10:00 - 12:00, 12:00 - 14:00

- freie:                jederzeit

- ◆ Wer's noch nicht getan hat:

- Bitte melden Sie sich im Terminalraum 02.151 (2. Stock Inf.-Hochhaus) unter dem Loginnamen: "gdijava" bis spätestens Freitag, 23.04. an.
- Das ist nötig, damit Sie Ihren Login bekommen!

## Notizen

---