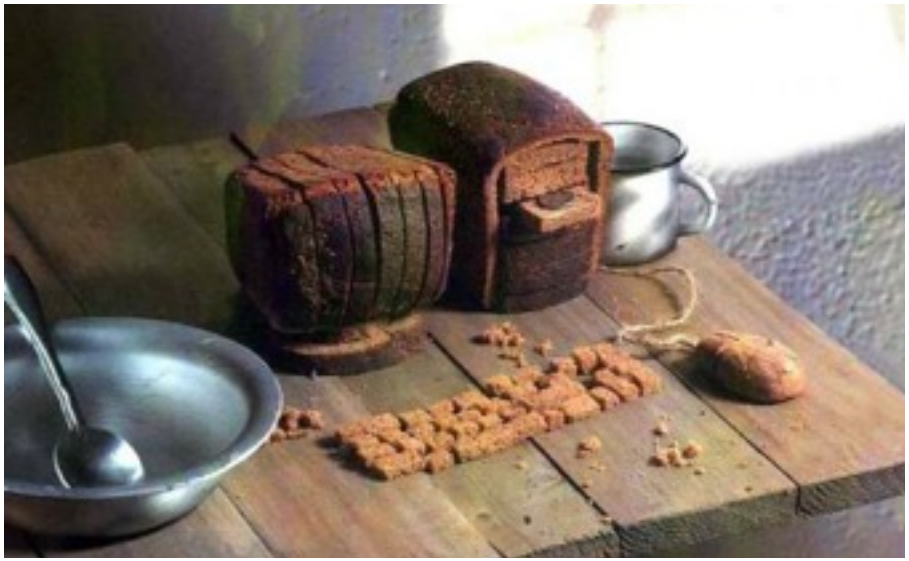
Programmiertechnik

Prof. Dr. Oliver Haase
Raum G124
haase@htwg-konstanz.de
Tel: 07531/206-150



Organisatorisches

Vorlesung:

- montags, 8:00 – 9:30h, Raum C-109
- freitags, 8:00 – 9:30h, Raum G-240
- Folien können von meiner Webseite heruntergeladen werden
  http://www-home.htwg-konstanz.de/~haase

Laborübungen:

- freitags 9:45 – 11:15h, F-114
- freitags 14:00 - 15:30h, F-112
- freitags 15:45 - 17:15h, F-112
- ***2-er Gruppen empfehlenswert (macht Spaß und bringt viel)***
- Übungsaufgaben befinden sich auf meiner Webseite
- Wenn Sie zuhause programmieren, benötigen Sie ***Java SDK1.5*** oder neuer.
  - Mehr Informationen dazu auf meiner Webseite.

Organisatorisches

RSS-Feed für kurzfristige Ankündigungen und Änderungen:

- unter http://www-home.htwg-konstanz.de/~haase/hp/Breaking_News/Breaking_News.html
- bitte unbedingt abonnieren!

Übungsscheinvergabe:

- Bestehen zweier von drei Testaten (30-minütige schriftliche Tests)
- Erstes Testat nach ca. 5 Wochen, zweites nach ca. 10 Wochen, drittes zum Semesterende
- Genaue Termine werden noch bekanntgegeben
- Pro Testat 50% der Punkte zum Bestehen erforderlich

Klausur:

- 90-minütige schriftliche Prüfung ohne Hilfsmittel.

Organisatorisches

Credit Points (ECTS) & Aufwand

- ❑ Vorlesung und Übungen ergeben zusammen 7 ECTS
- ⇨ wöchentlichen Arbeitsaufwand: ca. 11 Stunden
- ⇨ Vorlesung + Übungen: 4,5 Stunden pro Woche
- ⇨ *Selbstlernanteil: 6,5 Stunden pro Woche*, aufgeteilt auf
  - ➢ Vorlesungsvorbereitung (!!!)
  - ➢ Vorlesungsnachbereitung
  - ➢ eigenes Üben (zusätzlich zu Laborübungsstunden)

Empfohlene Literatur

- **Ratz, Scheffler, Seese. *Grundkurs Programmieren in Java*. Hanser Verlag 2010, ISBN 3446416552. €34,90**.
- Skript *Java 2 – Grundlagen und Einführung* des RRZN Hannover. Erhältlich für € 6.– (Mensakarte) bei Fr. Schulze, Raum G146.

- ansonsten: Jede Menge Material über Java im Web verfügbar…

Was ist Programmiertechnik?

Programmieren: *Die Tätigkeit, →Computerprogramme zu erstellen*.

Computerprogramm: *Folge von Befehlen, die auf einem Computer zur Ausführung gebracht werden können, um eine bestimmte Funktionalität zu erzeugen.*

Technik: *Die Anwendung von Methoden und Prinzipien, mit dem Ziel eine bestimmte Wirkung zu erzielen*.

Programmiertechnik: *Methoden und Prinzipien zur Erstellung von Computerprogrammen.*

Merke: Programmierung beinhaltet immer auch ein großes Maß an Kreativität!

Programmiersprachen

- Programme werden in einer ***Programmiersprache*** geschrieben
- Programme werden in ***Maschinencode*** übersetzt, d.h. in Zahlenfolgen, die der Prozessor ausführen kann.
- Die primitivsten Programmiersprachen sind die ***Assembler***-Sprachen
  - lediglich lesbare Namen für Maschinenbefehle, z.B.
    - ✓ `mov B, A` (schiebt Wert von Adresse `A` nach Adresse `B`)
    - ✓ `xchg Op1, Op2` (vertauscht die Werte von Operand `OP1` und `Op2`)
    - ✓ `jmp Marke` (springt zu dem mit `Marke` markierten Befehl)
  - rechnerabhängig
  - sehr *effizient*, d.h. kann sehr schnell ausgeführt werden
  - mühsam und fehleranfällig

Programmiersprachen

Heute fast nur noch ***höhere Programmiersprachen***

- rechnerunabhängig
- komfortabler zu programmieren durch abstraktere Konzepte wie
  - Prozeduren, Subroutinen, Methoden
  - Module
  - Klassen und Vererbung
  - etc.
- weniger effizient als Assembler
- wesentlich einfacher (billiger) zu warten (verändern, anpassen, erweitern)
- es existiert eine Vielzahl höherer Programmiersprachen, z.B. → *Ada, Algol, C, C++, C#, Cobol, Eiffel, Fortran,* ***Java****, Lisp, Modula-2, Oberon, ObjectiveC, Pascal, Prolog, Scheme, Smalltalk.*

Einschub: Warum Java?

- Eine Programmiersprache ist nicht unbedingt *besser* oder *schlechter* als andere, sondern hat *bestimmte Eigenschaften*, die für bestimmte Anwendungsgebiete wichtig sind.
- Für Java sind die für uns wichtigen Eigenschaften:
  - Java besitzt einfache, klare, objektorientierte Konzepte
  - Java ist *die* Programmiersprache des Internets (Web-Services, E-Commerce, E-Banking, Applets, Servlets, …)
  - Java verfügt über eine reiche Auswahl an Programmbibliotheken für *graphische Benutzeroberflächen*, *Computerkommunikation*, *verteilte Anwendungen*, *Datenbankapplikationen*, *Medien* (*Audio* und *Video*), …
  - Java hat eine hohe praktische Relevanz

Programmausführung

Ein Programm in einer höheren Programmiersprache kann ***übersetzt*** (***kompiliert***) oder ***interpretiert*** werden:

- ***Kompilierung***:
  - Compiler übersetzt Quellprogramm in Maschinencode
  - Maschinencode kann vom Rechner direkt ausgeführt werden
  - effizient
  - Compiler und Maschinencode maschinenabhängig → portierter Code muss neu übersetzt werden
  - typische Vorgehensweise für C/C++

Programmausführung

- ***Interpretierung***:
  - Quellcode wird von einem Interpretierer Zeile für Zeile übersetzt und direkt ausgeführt
  - Maschinencode wird nicht gespeichert
  - Programmteile, die mehrfach durchlaufen werden, werden auch mehrfach übersetzt.
  - ineffizient
  - Programm kann direkt auf jede Zielmaschine portiert werden, die einen Interpretierer für die Sprache besitzt

Programmausführung

- ***Mischform – Zwischenformat***:
  - Compiler übersetzt Quellcode in ein *maschinennahes* Zwischenformat (Bytecode)
  - Bytecode wird auf der Zielmaschine interpretiert
  - Bytecode maschinenunabhängig → portabel
  - typische Repräsentanten → Java, C#
  - Bytecode-Interpreter = ***Java Virtual Machine*** (***JVM***)

Das obige Ausführungsmodell führt zu portablem Code, der sehr schnell ausgeführt werden kann.

Java-Dateinamen

- Das einfachste Java-Programm besteht aus einer einzelnen →*Klasse*, z.B. **`MyClass`**.
- Diese Klasse muss in einer Datei des Namens *MyClass.java* definiert sein.
- Der Java-Compiler (javac) übersetzt diese Klasse in Bytecode und speichert das Ergebnis in der Datei *MyClass.class*.
- Die virtuelle Maschine (Java virtual machine – JVM) interpretiert diesen Bytecode.

Java-Programmgerüst

Die Datei *MyClass.java*, die die Klasse **`MyClass`** definiert, hat die folgende Struktur:

```
public class MyClass {

  public static void main(String[] args) {

    // hier kommt spaeter mal Ihr eigener
    // Programmtext hin.
  }
}
```

- Im Moment genügt es, diese Struktur zu *kennen*. Sie müssen sie noch nicht *verstehen*!
- Die Zeichen '//' leiten eine Kommentarzeile ein, d.h. der Java-Compiler ignoriert die entsprechende Zeile.

Have a nice cup of

Java coffee!