

Grundlagen der Informatik für Ingenieure I

19. Repetitorium

- 19.1 *Rechnerarchitektur*
- 19.2 *Betriebssysteme*
- 19.2.1 *Dateisysteme*
- 19.2.3 *Prozessverwaltung*
- 19.2.4 *Speicherverwaltung*
- 19.3 *OO Programmentwicklung*
- 19.3.1 *Klassen*
- 19.3.2 *Objekte/Instanzen*
- 19.3.3 *Methoden und Konstruktoren*
- 19.3.4 *Variable und Konstanten*
- 19.3.5 *Datentypen, Operatoren*

19. Repetitorium

- 19.3.6 *Java-Sprachkonstrukte*
- 19.3.7 *Modifier*
- 19.3.8 *Applets und Applikationen*
- 19.3.9 *Exceptionhandling*
- 19.3.10 *Interfaces*
- 19.3.11 *Java-E/A-System*
- 19.3.12 *Threads*
- 19.3.13 *AWT - Abstract Window Toolkit*
- 19.4 *Datenstrukturen*
- 19.5 *Allgemeines*

19.1 Rechnerarchitektur

- Rechnerarchitektur
 - Funktionseinheiten der "von Neumann-Architektur"
 - Rechenwerk
 - Leitwerk
 - Speicher
 - E/A-Werk
 - Wichtige CPU-Register und ihre Funktion
 - Welche "Attribute" der Rechnerarchitektur spielen eine Rolle im Zusammenhang mit
 - Wertebereichen
 - Genauigkeit
 - Adressierbarkeit
 - Typzugehörigkeit primitiver Datentypen
 - Leistungsmaße für Prozessoren
 - Speicherhierarchien
 - "Geschwindigkeitsproblem"
 - "Adressraumproblem"

19.2 Betriebssysteme

- Betriebssystemklassen
 - Batchsysteme, Realtime Systeme, Time Sharing Systeme
- Grobe Gliederung
 - Systemschnittstelle (API), System-Call-Handler
 - Dateisystem
 - Prozessverwaltungssystem
 - Speicherverwaltungssystem
 - E/A-System
- Ereignisse
 - asynchrone
 - synchrone
 - Interrupt
 - Trap

1 Dateisysteme

- Darstellung
 - Kataloge
 - Dateien
 - Links
 - Zugriffsrechte
 - Dateihierarchie

- Organisation (Datenstrukturen) und dynamischer Ablauf
 - E/A-Systemcalls
 - Dateikontrollstrukturen (inodes, filetables)
 - Puffersysteme

- Plattenorganisation

2 Prozessverwaltungssystem

- Prozesse, Threads
 - Schutzumgebungen
 - Zustände
 - Wechselwirkungen
 - Speicherverwaltung
 - Dateiverwaltung/ E-A-Verwaltung
 - Scheduler
 - Prozesshierarchie

3 Speicherverwaltungssystem

- Strukturierung von Speichern
 - physikalischer Adressraum/ logischer Adressraum
 - Bit, Bytes, Worte, Segmente, Kacheln
 - Codesegment/Datensegment

- Verfahren der Speicherverwaltung/Hardwareunterstützung
 - Segmentierung/Paging
 - Virtueller Speicher

19.3 OO Programmentwicklung

1 Klassen

- Was ist eine Klasse?
- Klassenhierarchien
- Vererbung (*Inheritance*)
 - *Single ...*
 - *Multiple ...*
 - (Wie und) was erbt man implizit?
 - Wie erbt man explizit?
- Klassenvariable
- Klassenmethoden
 - *Modifier*
 - Zugriff auf Variable

1 Klassen

- Klassendefinition
 - *extends*
 - *implements*
 - *Modifier*
 - *public*
 - *abstract*
 - *final*
 - Konstruktoren
 - *default Constructor*
 - *default finalizer*
 - (Reflexion)

2 Objekte/Instanzen

- Was ist ein Objekt, eine Instanz?
- Kreieren eines Objekts, Instantiieren
 - Template --> Objekt
 - Referenz auf das Objekt
 - *Constructor*
 - Variable
 - Code vs. Daten
 - Speichermanagement
- Objektzustand
- Instanzvariable
 - Gültigkeitsbereich
 - *Modifier*
 - *Access Controll*

3 Methoden und Konstruktoren

■ Methoden

- Definition
 - Signatur
 - Returntyp
 - *Access Control*
- Parameterübergabe
 - *Call by Reference*
 - *Call by Value*
- Modifier
 - *abstract*
 - *static*
 - *final*
 - *synchronized*
 - *Overriding*
 - *Overloading*

3 Methoden und Konstruktoren

■ Konstruktoren

- *Overriding*
- *Overloading*
- *super()*
- *super(p1, p2,...)*

■ Referenzen

- *this*
- *this()*
- *this.name*
- *return[value]*
- *super.methodname()*

3 Methoden und Konstruktoren

- Ausgezeichnete Methoden
 - main()
 - init()
 - run()

- Referenzen auf Objekte

- Zugehörigkeit eines Objekts zu einer Klasse

- Relationale Operationen auf Objekte

4 Variable und Konstanten

- Namenskonventionen

- Variable, Konstante
 - Definition
 - Datentypen
 - Namenskonstante, Literale (Konstante)
 - *Modifier*
 - *Initialisierung*
 - *Gültigkeitsbereiche*
 - *Wertebereiche (Wortbreite)*
 - *Genauigkeit (Wortbreite)*
 - *Lebensdauer*
 - *Casting*
 - *Converting*

5 Datentypen, Operatoren

■ Primitive Datentypen

- Typgruppen
- Typen
- Wertebereiche
- Operatoren
 - arithmetische
 - bool'sche
 - relationale
 - Vergleichsoperator als spez. Statement
 - *Casting*
 - *Converting*
 - *Objekte prim. Datentypen*
 - *arrays*
 - *strings*

6 Java-Sprachkonstrukte

- Quellcodelayout
- Strukturierung
- Statement
- Block of Statements
- Flußkontrolle
 - *if ..*
 - *if...then...*
 - *if...then...else*
 - *while...*
 - *do ... while*
 - *for....*
 - *break*
 - *continue*
 - *switch*
 - *label*

6 Java-Sprachkonstrukte

■ Regeln

- Precedenceregeln
- Klammerungen
- Ausprünge aus Schleifenkonstrukten
- Schreibweisen von
 - Klassennamen
 - Variablennamen
 - Methodennamen
 - Konstantennamen
 - Literalen

7 Modifier

- *default, public, private, final, static, protected, abstract, synchronized, (volatile), (native)* angewandt auf:
 - Variable
 - Instanzvariable
 - Klassenvariable
 - Methoden
 - Klassen
- Schutz-(Protection) Konzept
- Kapselung
- “Need to Know” - principle
- Access Methods

8 Applets und Applikationen

■ Applets

- Eigenschaften
- Voraussetzungen
- Anwendungsumfeld
- Restriktionen (Security)

■ Applikationen

- Eigenschaften
- Voraussetzungen
- Anwendungsumfeld

9 Exceptionhandling

- Worum geht es beim *Exceptionhandling*?
- Gruppierung von *Exceptions*
 - *synchron, asynchron*
 - *Error, Exceptions*
 - *Runtime-Exceptions*
 - *implizit, explizit*
 - *I/O-Exceptions*
- Sprachkonstrukte und ihre Bedeutung
 - *try {...}*
 - *catch {...}*
 - *throw*
 - *(...throws)*
 - *finally*

10 Interfaces

- Was ist ein Interface?(vs. abstracte Klassen)
 - Hierarchiebeziehungen
 - nur abstrakte Methoden
 - Namenskonstante sind möglich
 - *implements*
 - vollständige Implementierung
 - mehrfache Interfaces
 - Interfacedefinition
 - Beispiele
 - *Runnable (threads)*
 - *xxxxListener*
 - *LayoutManager*
 - *DataInput*
 - *DataOutput*

11 Java-E/A-System

- Streams-konzept
 - *InputStreams*
 - typische Methoden
 - Reader
 - InputStream
 - *OutputStreams*
 - typische Methode
 - Writer
 - OutputStream
 - Ein/Ausgabe auf Dateien
 - Ein-/Ausgabe auf Sockets
- Standard-E/A
- Formatierte E/A

12 Threads

- Was sind *Threads*?

- Threadkontrolle
 - *run()*
 - *start()*
 - *yield()*

- Threadanwendungen

13 Abstract Window Toolkit

- Was ist das AWT?
 - Basiskonstrukte
 - Layoutkonstrukte
 - *Windows*

- ...in Applets

- ...in Applikationen

- *Eventhandling*
 - *Listener*
 - *Events*

19.4 Datenstrukturen

- Pointer (Zeiger) in C und Java
- Felder(arrays)
- Verkettete Listen
- Warteschlangen, Kellerspeicher (wo einschlägig?)
- Suchen und Sortieren
- Gestreute Speicherung (Hashing)

19.5 Allgemeines

- Was versteht man unter dem Begriff *Java VM*?
- Was versteht man unter einer interpretativen Abarbeitung?
 - Welche Vorteile
 - Welche Nachteile
- Was ist eine Klassenbibliothek?
- Was ist ein Package?
- Was verspricht man sich vom Einsatz des OO-Programmiermodells?
- Wie erreicht man die Kompatibilität von Java-Programm-Systemen?
- *Byte Code*
- Was ist ein "*Just in Time Compiler*"?
- Was versteht man unter "Garbage Collection"?
- Java und Security

