INFORMATIK

Vorlesungskarte zu den Vorlesungen
Informatik A für Molecular Life Sciences 2006
Informatik B für Molecular Life Sciences 2007

Veranstaltungsziele
- Informationsverarbeitende Systeme theoretisch verstehen
- Informationsverarbeitende Systeme in Forschungs- und Entwicklungsprojekten benutzen können
- Algorithmen und Datenstrukturen verschiedenen Projektbedürfnissen anpassen können
- Vorbereitung auf vertiefende Informatikveranstaltungen

Theoretische Informatik

Linugistik
Satz
Subjekt ist
Das Sein

- kontextfrei
- regulär
- Grammatiken
- Spachen
- reguläre Ausdrücke

B7 Formale Grammatiken
- Das Konzept der formalen Sprache verstehen
- Probleme als formale Sprachen formulieren können
- Syntax von kontextfreien und regulären Grammatiken kennen
- Ableitungen bilden können und die erzeugte Sprache angeben können
- Probleme mittels formalen Grammatiken beschreiben können

B9 Reguläre Ausdrücke
- Konzept des regulären Ausdrucks verstehen
- Probleme als reguläre Ausdrücke beschreiben können
- Verhältnis von regulären Ausdrücken und endlichen Automaten kennen

- Theorie formaler Sprachen
- DFAs
- NFAs
- Automatentheorie

B8 Endliche Automaten
- Funktionsweise von endlichen Automaten verstehen
- Formale Sprachen mittels endlicher Automaten entscheiden können
- Endliche Automaten implementieren können
- Einschätzen können, ob eine Sprache regulär ist

- Komplexitätstheorie
- Ressourcen
- Klassen

A16 Laufzeit von Algorithmen
- Wichtige Ressourcemaße verstehen und erklären können
- Das Laufzeitverhalten der wichtigsten Algorithmen kennen
- Das Laufzeitverhalten einfacher Programme abschätzen können
- Die Gruppierung von Problemen nach groben Laufzeitklassen verstehen

Mathematik
$\sum_{i=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$

- Typtheorie
- Künstliche Intelligenz
- Maschinenmodelle

Elektrotechnik

Maschinenbau

Technische Informatik

- Robotik
- Prozessoren
- Schaltkreise

A2 Computer-Hardware
- Grundlegende Rechnerarchitekturen kennen
- Klassen von Computern kennen und ihre Leistungsfähigkeit abschätzen können
- Den prinzipiellen Aufbau von Computern beschreiben können
- Die wichtigsten Hardwarekomponenten kennen und ihre Leistungsfähigkeit abschätzen können

B6 IT-Sicherheit
- Bedrohungsszenarien der Sicherheit von IT-Systemen kennen und einschätzen können
- Schutzmaßnahmen für die Sicherheit von IT-Systemen kennen und ergreifen können

- Netzwerke
- Peripherie
- Maus
- Bildschirm
- Drucker
- Tastatur
- Netzwerkkarte
- Internet
- Funk
- Bluetooth
- Wirless LAN
- Kabel
- Telefon
- ISDN
- DSL

B5 Rechnernetze und das Internet
- Konzept des Kommunikationsprotokolls kennen.
- OSI-Schichtenmodell kennen.
- Unterschied zwischen LAN und Internet kennen.
- Vor- und Nachteile netzbasierter Projekte und Ansätze beurteilen können.

B4 Das World-Wide-Web
- Aufbau des WWW kennen
- Aufbau von HTML kennen
- Unterschied zwischen Web-Clients und Web-Servern kennen
- Eigene Webseiten erstellen können

Praktische Informatik

Algorithmen

- Algorithmische Methoden
- Back-tracking
- Approximation
- Greedy-Heuristiken
- Rekursion
- Dynamische Programme
- geometrische
- Graph-algorithmen
- Sortieren
- Suchen
- Optimierung

A17 Einführung zur Rekursion
- Das Konzept der Rekursion verstehen
- Rekursive Methoden implementieren können für einfache Probleme

A18 Rekursion und schnelles Sortieren
- Merge-Sort verstehen und implementieren können
- Quick-Sort verstehen und implementieren können

A15 Sortieralgorithmen
- Arten von Sortierproblemen kennen
- Bubble-Sort, Insertion-Sort, Selection-Sort verstehen und implementieren können

A14 Suchalgorithmen
- Lineare Suche verstehen und implementieren können
- Binäre Suche verstehen und implementieren können
- Laufzeit der Verfahren vergleichen können
- Abschätzen können, wann sich Vorsortierung lohnt

B13 Approximationsalgorithmen
- Das Konzept der approximativen Lösung verstehen
- Lösungsverfahren für konkrete Probleme kennen

B12 Backtracking-Algorithmen
- Die Idee des Backtracking verstehen
- Backtracking-Algorithmen implementieren können

B11 Greedy-Heuristiken
- Das Konzept der heuristischen Lösung verstehen
- Die Idee von Greedy-Heuristiken kennen
- Beispiele von Greedy-Heuristiken kennen
- Greedy-Heuristiken entwerfen können

A1 Information und Daten
- Die binäre Speicherform verschiedener Daten kennen und verstehen
- Wichtige Einheiten von Information kennen und die Speichergrößte von Daten abschätzen können
- Beliebige Daten kodieren können
- Das Konzept der Datenkompression kennen und verstehen

B10 Optimierungsprobleme
- Probleme als Entscheidungsprobleme, Funktionsprobleme und Optimierungsprobleme formulieren können
- Die Konzepte Optimierungsproblem, Lösung und Maße verstehen
- Schwierige und leichte Optimierungsprobleme kennen

Datenstrukturen

- zusammen-gesetzte
- Graphen
- Bäume
- Hashing
- Arrays
- Listen
- einfache
- Strings
- Zahlen

A25 Suchbäume
- Suchbäume in Java implementieren können
- 2-3-Suchbäumen kennen und verstehen
- Vor- und Nachteile von Arrays, Listen, einfachen Bäumen und fortgeschrittenen Suchbäumen beurteilen können

A24 Bäume
- Das Konzept des (Such)Baumes verstehen
- (Such)Bäume in Java implementieren können
- Baumtraversierung kennen und implementieren können

A26 Graphen
- Graphen als mathematische Objekte verstehen
- Daten und ihre Struktur mittels Graphen modellieren können
- Die Datenstruktur des Graphen verstehen und implementieren können

A27 Graphalgorithmen
- Einige einfache und schwierige Graphprobleme kennen
- Arten der Graphtraversierung kennen
- Einen Kürzeste-Wege-Algorithmus kennen

A11 Arrays (Felder)
- Konzept des Arrays verstehen
- Java-Syntax von Arrays beherrschen
- Java-Programme mit Array-Nutzung implementieren können

A22 Listen als Datenstruktur
- Das Konzept der komplexen Datenstruktur verstehen
- Die Liste als Datenstruktur verstehen und in Java implementieren können
- Den Unterschied zwischen Listen und Arrays erklären können

A23 Angewandte Listen
- Fortgeschrittene Arten von Listen kennen
- Von und Nachteile von Listen und Arrays benennen können
- Das Konzept der abstrakten Datenstruktur kennen

A10 Zeichenketten
- Zeichenketten (Strings) als Datentypen kennen
- Einfache Stringalgorithmen kennen und implementieren können
- Fortgeschrittene Stringalgorithmen kennen und implementieren können

A6 Der Algorithmusbegriff
- Den Begriff des Algorithmus verstehen
- Einfache Lösungsideen als Algorithmen formulieren können
- Probleme spezifizieren können
- Programmiersprachen als Kommunikationsmittel für Algorithmen begreifen
- Das Konzept des Übersetzers verstehen

A9 Elementare Datentypen
- Variablendeklarationen verstehen und benutzen können
- Elementare Datentypen von Java kennen
- Typkonzeptkraft und Typfehler verstehen

Programmier-sprachen

- imperativ
- C
- Assembler
- Pascal
- objektorientiert
- Java
- C++
- Smalltalk
- funktional
- logisch
- ML
- Scheme
- Prolog

A7 Imperative Programmierung
- Zuweisungen verstehen und benutzen können
- Grundlegende Kontrollstrukturen verstehen und benutzen können
- Algorithmen als imperative Java-Programme formulieren können

A8 Die Programmiersprache Java
- Syntax des imperativen Kerns von Java beherrschen
- Java-Programme selbstständig erstellen können
- Java-Programme übersetzen und testen können

Softwaretechnik

- Planung
- Modularisierung
- Strukturierung
- Modellierung

A13 Modularisierung im Kleinen
- Das Konzept des Unterprogramms/Methode verstehen und anwenden können
- Java-Syntax der Modularisierung auf Methodenebene beherrschen

A19 Modularisierung im Großen
- Das Konzept des Top-Down-Entwurf verstehen und anwenden können
- Das Konzept der Klassen- und Modulbildung verstehen

A12 Scoping
- Scoping verstehen und anwenden können
- Die Speicherung von Variablen verstehen

A20 Klassen und Modellierung
- Konzept der Klasse verstehen
- Modellierung als Methodik kennen
- Statische Klassenmodelle von Anwendungsszenarien erstellen können

A21 Klassen und Objekte
- Lebenszyklus von Objekten kennen
- Neue Datentypen deklarieren und benutzen können
- Nichtstatische Methoden kennen und benutzen können
- Java-Syntax für Objekte kennen

Wirtschaft €

Angewandte Informatik

WWW

Betriebssysteme

- Shells
- Unix
- Windows

A4 Betriebssysteme
- Aufgaben von Betriebssystemen kennen
- Konzepte der Benutzer-, Prozess-, und Dateiverwaltung kennen
- Mit Dateien und Dateirechten umgehen können
- Begriff der Shell kennen

A5 Shells
- Unix- und Programm-Shells benutzen können
- Wichtige Unix-Kommandos kennen
- Einfache Shell-Skripte erstellen können

Anwendung

- Word
- Matlab
- SPSS
- Mail

A3 Computer-Software
- Ein-Prozess-Systeme und das EVA-Prinzip verstehen
- Multi-Prozess-Systeme verstehen
- Programme als Instruktionsfolgen begreifen
- Verschiedene Arten von Software benennen und unterscheiden können

B1 Datenbanken
- Das Konzept der Datenbank kennen und verstehen
- Das relationale Datenbankmodell kennen
- Daten mittels des relationalen Modells modellieren können

Datenbanken

- Theorie
- ER-Diagramme
- Relationen-Algebren
- Datenbank-Sprachen
- SQL
- Datenbank-Systeme
- MySQL
- DB2

B2 Relationenalgebren
- Konzeptder Relation-Algebra kennen
- Anfragen mittels Operationen formulieren können
- Joins verstehen und anwenden können

B3 SQL
- Syntax von SQL beherrschen
- SQL-Queries erstellen können
- Mit Datenbanken mittels einer SQL-Shell kommunizieren können

Informatik-Einführung

Einführung, Erwartungsabfrage
Information und Daten
Computer-Hardware
Computer-Software
Betriebssyteme
Shells

Java-Einführung

Der Algorithmus-Begriff
Imperative Programmierung
Programmiersprache Java
Elementare Datentypen
Zeichenketten
Arrays
Scoping
Modularisierung im Kleinen

Algorithmik

Suchalgorithmen
Sortieralgorithmen
Laufzeit von Algorithmen
Einführung Rekursion
Rekursives Sortieren

Modellierung

Modularisierung im Großen
Klassen und Modellierung

Datenstrukturen

Klassen und Objekte
Listen I
Listen II
Bäume I
Bäume II
Graphen I
Graphen II
Zusammenfassung, Evaluation

Oktober 2006 · November 2006 · Dezember 2006 · Januar 2007 · Februar 2007

April 2007 · Mai 2007 · Juni 2007 · Juli 2007

Einführung
Datenbanken
Datenbanken
Relationen-Algebren
SQL
Internet und WWW
WWW
Internet
IT-Sicherheit
Theorie der Zeichenketten
Grammatiken
Endliche Automaten
Reguläre Ausdrücke
Theorie schwieriger Probleme
Optimierung
Greedy-Heuristiken
Backtracking
Approximation
Zusammenfassung, Evaluation

Universität zu Lübeck · Institut für Theoretische Informatik · Theoretical Computer Science

Stand vom 8. Oktober 2006
www.tcs.uni-luebeck.de/pages/tantau/teaching.html – Lehre von Till Tantau