universität innsbruck

Einführung in die Theoretische Informatik

David Drexel
Julian Fodor
Alexander Lochmann
Georg Moser
Alexander Maringele
David Obwaller
Jonas Schöpf

cbr.uibk.ac.at

Zeitplan

Woche 1	7. Oktober	Woche 7	18. November
Woche 2	14. Oktober	Woche 8	25. November
Woche 3	21. Oktober	Woche 9	2. Dezember
Woche 4	28. Oktober	Woche 10	9. Dezember
Woche 5	4. November	Woche 11	13. Jänner
Woche 6	11. November	Woche 12	20. Jänner
		1te Klausur	27. Jänner

Zeit und Ort

Vorlesung	Montag, 12:15–13:45, Großer Hörsaal	Georg Moser
Tutorium	Montag, 18:15-19:00, SR 12	Julian Fodor

in der Vorlesung besteht keine Anwesenheitspflicht

universität innsbruck

Organisation

Vorlesungsmaterial

1. Skriptum
   bei Studia (*8te überarbeite Auflage*)

Online-Lehrmittel

2. Skriptum ist diese Woche bei Studia und nächste Woche innerhalb des Universitätsnetzes verfügbar sein
3. Folien und Hausaufgaben sind auf der LVA-Homepage abrufbar
4. Folien sind vor der Vorlesung online
5. Ausgewählte Lösungen werden verfügbar gemacht, nachdem sie in den Proseminargruppen besprochen wurden

Zeit und Ort der Studienorientierungslehrveranstaltungen (SL)

Gruppe 1	Freitag, 12:15–13:00, SR 12	Alexander Maringele
Gruppe 2	Freitag, 11:15–12:00, SR 12	David Drexel
Gruppe 3	Freitag, 10:15–11:00, SR 12	David Drexel
Gruppe 4	Freitag, 12:15–13:00, SR 13	Alexander Lochmann
Gruppe 5	Freitag, 12:15–13:00, HS 11	David Obwaller
Gruppe 6	Freitag, 11:15–12:00, HS 11	David Obwaller
Gruppe 7	Freitag, 10:15–11:00, HS 11	Jonas Schöpf
Gruppe 8	Freitag, 12:15–13:00, HSB 4	David Drexl
Gruppe 9	Freitag, 11:15–12:00, HSB 1	Alexander Maringele
Gruppe 10	Freitag, 10:15–11:00, HSB 1	Georg Moser

Termine

1. Die SL beginnt am 11. 10. und endet am 31. Jänner
2. SL Klausur am 31. Jänner

Prüfungsmodus in Vorlesung & SL

SL

In der SL herrscht keine Anwesenheitspflicht;
Anwesenheit wird aber dringend empfohlen

Klausuren

1. Die erste Vorlesungsprüfung findet am 27. Jänner statt
2. Die Prüfung ist closed-book
3. Aufgrund des späten Beginns finden nur 11 SLs statt; teilweise werden deshalb Beispiele nur in der Vorlesung besprochen
4. Die Klausurvorbereitungen (für VO+SL) finden im Tutorium statt
5. Zur Abgrenzung von der VO KLausur wird sich die SL Klausur auf einen bestimmten Bereich der Theoretischen Informatik beschränken (aber vertiefen)

Notenschlüssel für die Klausuren

Punkte	$\geqslant 90$	$\geqslant 75$	$\geqslant 60$
Note	Sehr Gut	Gut	Befriedigend
Punkte	$\geqslant 50$	< 50	
Note	Genügend	Nicht Genügend	

universität innsbruck

Einleitung

Theoretische Informatik

Die Theoretische Informatik beschäftigt sich mit der Abstraktion, Modellbildung und grundlegenden Fragestellungen, die mit der Struktur, Verarbeitung, Übertragung und Wiedergabe von Informationen in Zusammenhang stehen.

Ihre Inhalte sind Automatentheorie, Theorie der formalen Sprachen, Berechenbarkeits- und Komplexitätstheorie, aber auch Logik und formale Semantik sowie die Informations-, Algorithmen- und Datenbanktheorie.

http://de.wikipedia.org/ 2019

1. Automatentheorie
2. Theorie der formalen Sprachen
3. Berechenbarkeits- und Komplexitätstheorie
4. Logik und formale Semantik
5. Informations-, Algorithmen- und Datenbanktheorie

Handbook of Theoretical Computer Science

 +  = 2293 Seiten, 4 Kilogramm

Inhaltsverzeichnis Band „Algorithms and Complexity“

Machine models and simulations, A catalog of complexity classes, Machine-independent complexity theory, Kolmogorov complexity and its applications, Algorithms for finding patterns in strings, Data structures, Computational geometry, Algorithmic motion planning in robotics, Average-case analysis of algorithms and data structures, Graph algorithms, Cryptography, Algebraic complexity theory, Algorithms in number theory, The complexity of finite functions, Communication networks, VLSI theory, Parallel algorithms for shared-memory machines, General purpose parallel architectures

Inhaltsverzeichnis Band „Formal Models and Semantics“

Finite automata, Context-free languages, Formal languages and power series, Automata on infinite objects, Graph rewriting: an algebraic and logic approach, Rewrite systems, Functional programming and lambda calculus, Type systems for programming languages, Recursive applicative program schemes, Logic programming, Denotational semantics, Semantic domains, Algebraic specification, Logics of programs, Methods and logics for proving programs, Temporal and modal logic, Elements of relational database theory, Distributed computing: models and methods, Operational and algebraic semantics of concurrent processes

universität innsbruck

Inhalte der Lehrveranstaltung

Einführung in die Logik

Syntax & Semantik der Aussagenlogik, Formales Beweisen, Konjunktive und Disjunktive Normalformen

Einführung in die Algebra

algebraische Strukturen, Beispiele von Algebren, Zusammenhang Boolesche Algebra und Aussagenlogik, Universelle Algebra, Satz von Birkhoff

Einführung in die Theorie der Formalen Sprachen

Grammatiken und Formale Sprachen, Reguläre Sprachen, Kontextfreie Sprachen, Chomsky-Hierarchie, Anwendungen von formalen Sprachen

Einführung in die Berechenbarkeitstheorie

Algorithmisch unlösbare Probleme, Turing Maschinen, Registermaschinen

Einführung in die Programmverifikation

Prinzipien der Analyse von Programmen, Verifikation nach Hoare

universität innsbruck

Wegweiser

Einführung in die Logik

Grundlage für: Logik (4. Semester), Induktive Logische Programmierung und Automatisches Beweisen (Master); nützlich für Funktionale Programmierung (1. Semester)

Einführung in die Algebra

Nützlich für Softwarearchitektur (3. Semester); Anwendungen in der Rechnerarchitektur (1. Semester)

Einführung in die Theorie der Formalen Sprachen

Grundlage für: Diskrete Strukturen (3. Semester); nützlich für Formale Sprachen und Automatentheorie, Compilerbau (Master)

Einführung in die Programmverifikation

wiederverwendet in Software Engineering (4. Semester) und Software Qualität (Wahlpflichtmodul 5. Semester)

	Rechenmodelle	Digitalrechner	
	Turing Maschinen, Berechenbarkeitstheorie, Alan Turing	—	1930er
	Formale Sprachen, Automatentheorie	`Zuse Z3`, `ENIAC`	1940er
	Grammatiken, Grundlagen des Compilerbaus, Noam Chomsky	`UNIVAC`, Transistoren statt Röhren	1950er
	P vs. NP Komplexitätstheorie, Stephen Cook	Minicomputer, integrierte Schaltkreise	1960er

Entschlüsselung der ENIGMA

- „Enigma“ ist griechisch für Rätsel
- deutsche Kodiermaschine eingesetzt im 2. Weltkrieg
- galt als unentzifferbar, Entschlüsselung benötigte etwa 8 Jahre
- Hauptakteure der Entschlüssung: Rejewski & Turing
- manuelle Entschlüsselung erwies sich als nicht praktiabel (und eigentlich unmöglich)
- Code wurde maschinell entschlüsselt
- wesentliche Werkzeuge: mathematische Analyse und Automatisierung
- “*It was thanks to Ultra that we won the war*” (W. Churchill)

universität innsbruck

Einführung in die Logik

Frage

Wie argumentieren wir im täglichen Leben?

Beispiel

Wenn Ostern auf Pfingsten fällt und Weihnachten nach Ostern ist, stimmt es dann zu sagen Pfingsten ist vor Weihnachten?[1]

Ja? ☑ Nein? ☐

Beispiel (fortgeführt)

„Keine logische Beziehung zwischen 'vor' und 'nach', sowie 'ist' und 'fällt'"
© Hesse/Schrader, Career4Young

[1] www.der-eignungstest.de/testtraining-einstellungstest/absurde-schlussfolgerungen-test-logik.php

Beispiel

Großbritannien ist kein Königreich und die Queen ist ein Mann. Stimmt es dann, dass ein Mann die Queen ist?

Ja? ☐ Nein? ☑

Beispiel

Die Nächte sind im Sommer immer hell.
Im Sommer sind die Tage immer Winter.
Im Winter sind die Tage immer dunkel.

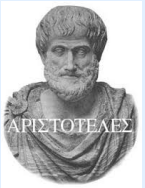
Stimmt es dann, dass die Tage im Winter dunkel sind?

Ja? ☑ Nein? ☐

Aristoteles sagte

Topik I 1, 100a25-27

Eine Deduktion (syllogismos) ist also ein Argument, in welchem sich, wenn etwas gesetzt wurde, etwas anderes als das Gesetzte mit Notwendigkeit durch das Gesetzte ergibt



Beispiel

Sokrates ist ein Mensch } Prämisse ①
Alle Menschen sind sterblich } Prämisse ②
Somit ist Sokrates sterblich } Konklusion

Definition

- Schlussfiguren dieser Art heißen Syllogismen
- Syllogismen wurden bereits im antiken Griechenland untersucht, Grundlage der modernen Logik

Fakt

Nicht die Wahrheit der Prämissen, oder der Konklusion, sondern die Wahrheit der Schlussfigur ist entscheidend

Beispiele für Arten von Syllogismen

Alle Griechen sind Menschen — AAA - modus barbara
Alle Menschen sind sterblich
Somit sind alle Griechen sterblich

Alle Professoren sind ernst — AOO - modus baroco
Einige Dozenten sind nicht ernst
Somit sind einige Dozenten keine Professoren

Typisierung der Relationen

A Alle *S* sind *P* E Keine *S* sind *P*
I Einige *S* sind *P* O Einige *S* sind nicht *P*

Modus Ponens

Beispiel

Wenn das Kind schreit, hat es Hunger
Das Kind schreit
Also, hat das Kind Hunger

Fakt

Korrektheit dieser Schlussfigur ist unabhängig von den konkreten Aussagen

Definition (*Modus Ponens*)

Wenn *A*, dann *B*
A gilt
Also, gilt *B*

Die Bedeutung der Logik in der Informatik ist um einiges größer als die Bedeutung der Logik in der Mathematik (oder Philosophie, Lingustik, . . .)