



UNIVERSITÄT
DES
SAARLANDES

FAKULTÄT FÜR MATHEMATIK UND INFORMATIK

MODULHANDBUCH

Medieninformatik BSc

1. Oktober 2026

Liste der Modulbereiche und Module

1	Ringvorlesungen	3
1.1	Perspektiven der Informatik	4
2	Grundlagen der Mathematik	5
2.1	Mathematik für Informatiker 1	6
2.2	Mathematik für Informatiker 2	8
3	Grundlagen der Informatik	10
3.1	Big Data Engineering	11
3.2	Elements of Machine Learning	14
3.3	Grundzüge der Theoretischen Informatik	16
3.4	Grundzüge von Algorithmen und Datenstrukturen	18
3.5	Nebenläufige Programmierung	19
3.6	Programming 1	21
3.7	Programming 2	23
3.8	Statistics Lab	25
3.9	Systemarchitektur	27
4	Praktika der Informatik	29
4.1	Software Engineering Lab (Softwarepraktikum)	30
4.2	Software Engineering Lab (Softwarepraktikum) LA LAB und Sek I+II	32
5	Spezialisierte Bereich Medieninformatik	34
5.1	Grundlagen der Medieninformatik	35
5.2	Human Computer Interaction	36
5.3	Interaktive Systeme	37
5.4	Medienprojekt	38
5.5	Statistik mit R	39
6	HBKsaar	40
6.1	Grundlagen Media Art & Design	41

6.2	MAD-Projekt	42
7	Psychologie	43
7.1	Allgemeine Psychologie 1	44
7.2	Seminar Sozialpsychologische Aspekte der Medienpsychologie	46
8	Seminare	47
8.1	Proseminar Informatik	48
8.2	Seminar Informatik	50
9	Stammvorlesungen	52
9.1	Algorithms and Data Structures	53
9.2	Artificial Intelligence	54
9.3	Automated Reasoning	56
9.4	Compiler Construction	57
9.5	Complexity Theory	58
9.6	Computer Algebra	59
9.7	Computer Graphics	60
9.8	Cryptography	62
9.9	Cyber-Physical Systems	63
9.10	Data Networks	65
9.11	Database Systems	67
9.12	Digital Transmission & Signal Processing	69
9.13	Distributed Systems	71
9.14	Geometric Modelling	72
9.15	Image Processing and Computer Vision	74
9.16	Information Retrieval and Data Mining	76
9.17	Introduction to Computational Logic	77
9.18	Machine Learning	78
9.19	Operating Systems	79
9.20	Optimization	81
9.21	Security	82
9.22	Semantics	83
9.23	Software Engineering	84
9.24	Verification	86
10	Bachelor-Seminar und -Arbeit	87
10.1	Bachelor-Seminar	88
10.2	Bachelor-Arbeit	89

Modulbereich 1

Ringvorlesungen

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
1	6	jedes Wintersemester	1 Semester	2	2

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Dozent/inn/en der Fachrichtung

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen Nachweis über das inhaltliche Verständnis von mindestens drei Vorträgen, z.B. durch schriftliche Ausarbeitung oder Test.

Lehrveranstaltungen / SWS 2 SWS Vorlesung

Arbeitsaufwand 30 h Präsenzstudium
+ 30 h Eigenstudium
= 60 h (= 2 ECTS)

Modulnote Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde (unbenotet).

Sprache Deutsch / English

Lernziele / Kompetenzen

Frühzeitige Motivierung und Überblick über die zentralen wissenschaftlichen Fragestellungen der Informatik, sowie über die Kompetenzen der Saarbrücker Informatik.

Inhalt

Vorträge durch wöchentlich wechselnde Dozent/inn/en bieten einen Querschnitt durch die Forschungsthemen der Saarbrücker Informatik. Die Themen spannen einen attraktiven Bogen von aktuellster Forschung zu anspruchsvollen Problemen der industriellen Praxis.

Literaturhinweise

Material wird passend zu den zu den einzelnen Vorträgen bereitgestellt.

Weitere Informationen

Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *Perspectives in Computer Science*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
1	6	jedes Wintersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Joachim Weickert

Dozent/inn/en Prof. Dr. Joachim Weickert
 Prof. Dr. Mark Groves
 Prof. Dr. Henryk Zähle
 Prof. Dr. Christian Bender

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen

- Teilnahme an den Übungen und Bearbeitung der wöchentlichen Übungsaufgaben (50 Prozent der Übungspunkte werden zur Klausurteilnahme benötigt)
- Bestehen der Abschlussklausur oder der Nachklausur

Lehrveranstaltungen / SWS 4 SWS Vorlesung
 + 2 SWS Übung
 = 6 SWS

Arbeitsaufwand 90 h Präsenzstudium
 + 180 h Eigenstudium
 = 270 h (= 9 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch und Englisch

Lernziele / Kompetenzen

- Erarbeitung von mathematischem Grundlagenwissen, das im Rahmen eines Informatik- bzw. Bioinformatikstudiums benötigt wird
- Fähigkeit zur Formalisierung und Abstraktion
- Befähigung zur Aneignung weiteren mathematischen Wissens mit Hilfe von Lehrbüchern

Inhalt

Die Zahlen geben die Gesamtzahl der Doppelstunden an.

DISKRETE MATHEMATIK UND EINDIMENSIONALE ANALYSIS

- A. Grundlagen der diskreten Mathematik (8)
1. Mengen (1)
 2. Logik (1)
 3. Beweisprinzipien, incl. vollst. Induktion (1)
 4. Relationen (1)
 5. Abbildungen (2)
 - injektiv, surjektiv, bijektiv
 - Mächtigkeit, Abzählbarkeit
 - Schubfachprinzip
 6. Primzahlen und Teiler (1)
 7. Modulare Arithmetik (1)

B. Eindimensionale Analysis (22)

B.1 Zahlen, Folgen und Reihen (8)

- 8. Axiomatik der reellen Zahlen, $\sup$, $\inf$ (1)
- 9. Komplexe Zahlen (1)
- 10. Folgen (1 1/2)
- 11. Landau'sche Symbole (1/2)
- 12. Reihen: Konvergenzkriterien, absolute Kgz. (2)
- 13. Potenzreihen (1/2)
- 14. Zahlendarstellungen (1/2)
- 15. Binomialkoeffizienten und Binomialreihe (1)

B.2 Eindimensionale Differentialrechnung (8)

- 16. Stetigkeit (1)
- 17. Elementare Funktionen (1)
- 18. Differenzierbarkeit (1 1/2)
- 19. Mittelwertsätze und L'Hospital (1/2)
- 20. Satz von Taylor (1)
- 21. Lokale Extrema, Konvexität, Kurvendiskussion (2)
- 22. Numerische Differentiation (1)

B.3 Eindimensionale Integralrechnung (6)

- 23. Das bestimmte Integral (2)
- 24. Das unbestimmte Integral und die Stammfunktion (1)
- 25. Uneigentliche Integrale (1)
- 26. Numerische Verfahren zur Integration (1)
- 27. Kurven und Bogenlänge (1)

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Weitere Informationen

Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *Mathematics for Computer Scientists 1*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2	6	jedes Sommersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Joachim Weickert

Dozent/inn/en Prof. Dr. Joachim Weickert
 Prof. Dr. Mark Groves
 Prof. Dr. Henryk Zähle
 Prof. Dr. Christian Bender

Zulassungsvoraussetzungen Mathematik für Informatiker 1 (empfohlen)

Leistungskontrollen / Prüfungen

- Teilnahme an den Übungen und Bearbeitung der wöchentlichen Übungsaufgaben (50 Prozent der Übungspunkte werden zur Klausurteilnahme benötigt)
- Bestehen der Abschlussklausur oder der Nachklausur

Lehrveranstaltungen / SWS 4 SWS Vorlesung
 + 2 SWS Übung
 = 6 SWS

Arbeitsaufwand 90 h Präsenzstudium
 + 180 h Eigenstudium
 = 270 h (= 9 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch und Englisch

Lernziele / Kompetenzen

- Erarbeitung von mathematischem Grundlagenwissen, das im Rahmen eines Informatik- bzw. Bioinformatikstudiums benötigt wird
- Fähigkeit zur Formalisierung und Abstraktion
- Befähigung zur Aneignung weiteren mathematischen Wissens mit Hilfe von Lehrbüchern

Inhalt

Die Zahlen geben die Gesamtzahl der Doppelstunden an.

LINEARE ALGEBRA

C. Algebraische Strukturen (5)

- 29. Gruppen (2)
- 30. Ringe und Körper (1)
- 31. Polynomringe über allgemeinen Körpern (1/2)
- 32. Boole'sche Algebren (1/2)

D. Lineare Algebra (21)

- 33. Vektorräume (2)
 - Def., Bsp.,
 - lineare Abb.
 - Unterraum,
 - Erzeugnis, lineare Abhängigkeit, Basis, Austauschsatz

- 34. Lineare Abb. (Bild, Kern) (1)
- 35. Matrixschreibweise für lineare Abbildungen (1 1/2)
 - Interpretation als lineare Abbildungen
 - Multiplikation durch Hintereinanderausführung
 - Ringstruktur
 - Inverses
- 36. Rang einer Matrix (1/2)
- 37. Gauss-Algorithmus für lineare Gleichungssysteme: (2)
 - Gausselimination (1)
 - Lösungstheorie (1)
- 38. Iterative Verfahren für lineare Gleichungssysteme (1)
- 39. Determinanten (1)
- 40. Euklidische Vektorräume, Skalarprodukt (1)
- 41. Funktionalanalytische Verallgemeinerungen (1)
- 42. Orthogonalität (2)
- 43. Fourierreihen (1)
- 44. Orthogonale Matrizen (1)
- 45. Eigenwerte und Eigenvektoren (1)
- 46. Eigenwerte und Eigenvektoren symmetrischer Matrizen (1)
- 47. Quadratische Formen und positiv definite Matrizen (1)
- 48. Quadriken (1)
- 50. Matrixnormen und Eigenwertabschätzungen (1)
- 51. Numerische Berechnung von Eigenwerten und Eigenvektoren (1)

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Weitere Informationen

Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *Mathematics for Computer Scientists 2*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Sommersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Jens Dittrich

Dozent/inn/en Prof. Dr. Jens Dittrich

Zulassungsvoraussetzungen *Programming 1, Programming 2, Softwarepraktikum oder Projektpraktikum, Mathematik für Informatiker 1, sowie Grundzüge von Algorithmen und Datenstrukturen (jeweils empfohlen).*

Leistungskontrollen / Prüfungen Erfolgreiche Teilnahme an den Übungen/Projekt berechtigt zur Teilnahme an der Abschlussklausur.

Lehrveranstaltungen / SWS 2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS

Arbeitsaufwand 60 h Präsenzstudium
+ 120 h Eigenstudium
= 180 h (= 6 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen, und ggf. Projekt ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekanntgegeben.

Sprache Englisch

Lernziele / Kompetenzen

Die Vorlesung vermittelt grundlegende Kenntnisse über fundamentalen Konzepte von Datenmanagement und Datenanalyse im Kontext von Big Data und Data Science.

Im Rahmen der Übungen kann während des Semesters ein durchgehendes Projekt durchgeführt werden. Dies kann zum Beispiel ein soziales Netzwerk (im Stil von Facebook) sein bzw. jedes andere Projekt, in dem Techniken des Datenmanagements eingeübt werden können (z.B. naturwissenschaftliche Daten, Bilddaten, andere Webapplikationen, etc.). Zunächst wird dieses Projekt in E/R modelliert, dann umgesetzt und implementiert in einem Datenbankschema. Danach wird das Projekt erweitert, um auch unstrukturierte Daten verwalten und analysieren zu können. Insgesamt werden so an einem einzigen Projekt alle fundamentalen Techniken gezeigt, die für das Verwalten und Analysieren von Daten wichtig sind.

Inhalt

1 Einführung und Einordnung

- Einordnung und Abgrenzung: Data Science
- Wert von Daten: Das Gold des 21. Jahrhunderts
- Bedeutung von Datenbanksystemen
- Architekturen: 2-Tier, 3-Tier, etc
- Was sind eigentlich Daten?
- Modellierung vs Realität
- Kosten mangelhafter Modellierung
- Datenbanksystem nutzen vs selbst entwickeln
- Positive Beispiele für Apps
- Anforderungen
- Literaturhinweise
- Vorlesungsmodus

- 2 Datenmodellierung
 - Motivation
 - E/R
 - Relationales Modell
 - Hierarchische Daten
 - Graphen und RDF
 - Redundanz, Normalisierung, Denormalisierung
 - Objektrelationale DBMS
- 3 Anfragesprachen
 - Relationale Algebra
 - Hierarchische Anfragesprachen
 - Graphorientierte Anfragesprachen
- 4 SQL
 - Grundlagen
 - Zusammenhang mit relationaler Algebra
 - PostgreSQL
 - Integritätsbedingungen
 - Transaktionskonzept
 - ACID
 - Sichten (und access control lists)
- 5 Implementierungstechniken
 - Übersicht
 - vom WAS zum WIE
 - Kosten verschiedener Operationen
 - EXPLAIN
 - Physisches Design
 - Indexe, Tuning
 - Datenbank-Tuning
 - Regelbasierte Anfrageoptimierung
 - Kostenbasierte Anfrageoptimierung
 - Machine Learning als Anfrageoptimierungstechnik
- 6 Zeitliche und räumliche Daten
 - als Teil des Schemas
 - as of/time travel
 - append-only und Streaming
 - Versioning
 - Snapshotting (Software und OS-basiert)
 - Differential Files/LSM et al
 - Publish/Subscribe
 - Indexstrukturen
- 7 Recovery, Durability, Archivierung
 - Grundproblematik
 - Vergessen vs Komprimieren vs Kondensieren
 - Heiße vs kalte Daten
 - Archivierung
 - Redundanz
 - Implementierungsaspekte
 - UNDO/REDO
 - Logging
- 8 Nebenläufigkeitskontrolle
 - Serialisierbarkeitstheorie
 - Isolationslevels
 - Verteilte Datenbanksysteme: Sharding, HP, VP, permissioned Blockchains
 - Implementierungsaspekte

9 ETL und Data Cleaning

- Datenbankschnittstellen: JDBC et al
- Textdatenbanken: CSV, SQLite
- Data Warehousing
- Schema Matching
- Reporting
- Data Cleaning
- Denormalisierung, Caching, Materialisierung
- Workflows
- ETL und Data Science in Data Science und Machine Learning

10 Big Data

- Was ist eigentlich Big Data?
- Big Data vs Privatheit
- Beispiele: Zusammenführen von Daten
- Physische Barrieren

11 NoSQL

- Key/Value Stores
- KeyDocument Stores: MongoDB
- MapReduce
- Flink
- Spark

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Weitere Informationen

Dieses Modul wurde früher auch unter dem Namen *Informationssysteme* geführt. Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *Big Data Engineering*.

Informationen für "Mathematik und Informatik BSc": Diese Veranstaltung ist Teil des Wahlpflicht-Bereichs der Grundlagen der Informatik (siehe Studienordnung §5 (2) 4)

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	every winter semester	1 semester	4	6

Modulverantwortliche/r Prof. Dr. Jilles Vreeken
Prof. Dr. Isabel Valera

Dozent/inn/en Prof. Dr. Jilles Vreeken
Prof. Dr. Isabel Valera

Zulassungsvoraussetzungen The lecture assumes basic knowledge in statistics, linear algebra, and programming. It is advisable to have successfully completed *Mathematics for Computer Scientists 2* and *Statistics Lab*. The exercises use the programming language R. We will give a basic introduction to R in the first tutorial. In addition, for preparation the following materials are useful: *R for Beginners* by Emmanuel Paradis (especially chapters 1, 2, 3 and 6) and *An introduction to R* (Venables/Smith).

Leistungskontrollen / Prüfungen Prerequisite for admission to the examination is a cumulative 50% of the points of the theoretical and a cumulative 50% of the points of the practical tasks on the exercise sheets. Depending on the number of participants, the examinations are either written or oral. The final modality will be announced in the first two weeks of the lecture.

Lehrveranstaltungen / SWS 2 h lectures
+ 2 h tutorial
= 4 h (weekly)

Arbeitsaufwand 60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)

Modulnote Will be determined from performance in exams.

Sprache English

Lernziele / Kompetenzen

In this course we will discuss the foundations – the elements – of machine learning. In particular, we will focus on the ability of, given a data set, to choose an appropriate method for analyzing it, to select the appropriate parameters for the model generated by that method and to assess the quality of the resulting model. Both theoretical and practical aspects will be covered. What we cover will be relevant for computer scientists in general as well as for other scientists involved in data analysis and modeling.

Inhalt

The lecture covers basic machine learning methods, in particular the following contents:

- Introduction to statistical learning
- Overview over Supervised Learning
- Linear Regression
- Linear Classification
- Splines
- Model selection and estimation of the test errors
- Maximum-Likelihood Methods
- Additive Models
- Decision trees

- Boosting
- Dimensionality reduction
- Unsupervised learning
- Clustering
- Visualization

Literaturhinweise

The course broadly follows the book *An Introduction to Statistical Learning with Applications in R*, Springer (2013). In some cases, the course receives additional material from the book *The Elements of Statistical Learning*, Springer (second edition, 2009). The first book is the introductory text, the second covers more advanced topics. Both books are available as free PDFs. Any change of, or additional material will be announced before the start of the course on the course webpage.

Weitere Informationen

Informationen für "Mathematik und Informatik BSc": Diese Veranstaltung ist Teil des Wahlpflicht-Bereichs der Grundlagen der Informatik (siehe Studienordnung §5 (2) 4)

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Wintersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Raimund Seidel

Dozent/inn/en Prof. Dr. Raimund Seidel
Prof. Dr. Bernd Finkbeiner
Prof. Dr. Kurt Mehlhorn
Prof. Dr. Markus Bläser

Zulassungsvoraussetzungen *Programming 1* und *2* und *Mathematik für Informatiker 1* und *2* oder vergleichbare Veranstaltungen der Mathematik sind empfohlen.

Leistungskontrollen / Prüfungen Erfolgreiche Bearbeitung der Übungsaufgaben berechtigt zur Klausurteilnahme.

Lehrveranstaltungen / SWS 4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS

Arbeitsaufwand 90 h Präsenzstudium
+ 180 h Eigenstudium
= 270 h (= 9 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Englisch

Lernziele / Kompetenzen

Die Studierenden kennen verschiedene Rechenmodelle und ihre relativen Stärken und Mächtigkeiten.

Sie können für ausgewählte Probleme zeigen, ob diese in bestimmten Rechenmodellen lösbar sind oder nicht.

Sie verstehen den formalen Begriff der Berechenbarkeit wie auch der Nicht-Berechenbarkeit.

Sie können Probleme aufeinander reduzieren.

Sie sind vertraut mit den Grundzügen der Ressourcenbeschränkung (Zeit, Platz) für Berechnungen und der sich daraus ergebenden Komplexitätstheorie.

Inhalt

Die Sprachen der Chomsky Hierarchie und ihre verschiedenen Definitionen über Grammatiken und Automaten; Abschlusseigenschaften; Klassifikation von bestimmten Sprachen („Pumping lemmas“);

Determinismus und Nicht-Determinismus;

Turing Maschinen und äquivalente Modelle von allgemeiner Berechenbarkeit (z.B. μ -rekursive Funktionen, Random Access Machines) Reduzierbarkeit, Entscheidbarkeit, Nicht-Entscheidbarkeit;

Die Komplexitätsmaße Zeit und Platz; die Komplexitätsklassen P und NP;

Grundzüge der Theorie der NP-Vollständigkeit

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Weitere Informationen

Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *Introduction to Theoretical Computer Science*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jedes Wintersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Raimund Seidel

Dozent/inn/en Prof. Dr. Raimund Seidel
Prof. Dr. Kurt Mehlhorn
Prof. Dr. Markus Bläser

Zulassungsvoraussetzungen *Programming 1* und 2, und *Mathematik für Informatiker 1* und 2 oder vergleichbare Veranstaltungen der Mathematik sind empfohlen.

Leistungskontrollen / Prüfungen Erfolgreiche Bearbeitung der Übungsblätter berechtigt zur Klausurteilnahme.

Lehrveranstaltungen / SWS 2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS

Arbeitsaufwand 60 h Präsenzstudium
+ 120 h Eigenstudium
= 180 h (= 6 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Englisch

Lernziele / Kompetenzen

Die Studierenden lernen die wichtigsten Methoden des Entwurfs von Algorithmen und Datenstrukturen kennen: Teile-und-Herrsche, Dynamische Programmierung, inkrementelle Konstruktion, „Greedy“, Dezimierung, Hierarchisierung, Randomisierung. Sie lernen Algorithmen und Datenstrukturen bzgl. Zeit- und Platzverbrauch für das übliche RAM Maschinenmodell zu analysieren und auf Basis dieser Analysen zu vergleichen. Sie lernen verschiedene Arten der Analyse (schlechtester Fall, amortisiert, erwartet) einzusetzen.

Die Studierenden lernen wichtige effiziente Datenstrukturen und Algorithmen kennen. Sie sollen die Fähigkeit erwerben, vorhandene Methoden durch theoretische Analysen und Abwägungen für ihre Verwendbarkeit in tatsächlich auftretenden Szenarien zu prüfen. Ferner sollen die Studierenden die Fähigkeit trainieren, Algorithmen und Datenstrukturen unter dem Aspekt von Performanzgarantien zu entwickeln oder anzupassen

Inhalt

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Weitere Informationen

Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *Fundamentals of Data Structures and Algorithms*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Sommersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr.-Ing. Holger Hermanns

Dozent/inn/en Prof. Dr.-Ing. Holger Hermanns
Prof. Bernd Finkbeiner, Ph.D
Prof. Dr. Verena Wolf

Zulassungsvoraussetzungen *Programming 1 und 2, Softwarepraktikum, und Grundzüge der Theoretischen Informatik* (empfohlen)

Leistungskontrollen / Prüfungen Zwei Klausuren (Mitte und Ende der Vorlesungszeit), praktisches Projekt.
Eine Nachklausur für die Zwischenklausur findet vor der Hauptklausur statt, eine Nachklausur für die Hauptklausur findet in den letzten Wochen vor Vorlesungsbeginn des Folgesemesters statt.

Lehrveranstaltungen / SWS 4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS

Arbeitsaufwand 50 h Präsenzstudium
+ 130 h Eigenstudium
= 180 h (= 6 ECTS)

Modulnote Wird aus Leistungen in den Klausuren, sowie den Prüfungsvorleistungen ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Englisch

Lernziele / Kompetenzen

Die Teilnehmer lernen die Nebenläufigkeit von Berechnungen als ein weitreichendes, grundlegendes Prinzip in der Theorie und Anwendung der modernen Informatik kennen. Durch die Untersuchung und Verwendung unterschiedlicher formaler Modelle gewinnen die Teilnehmer ein vertieftes Verständnis von Nebenläufigkeit. Dabei lernen die Teilnehmer wichtige formale Konzepte der Informatik korrekt anzuwenden. Das im ersten Teil der Veranstaltung erworbene theoretische Wissen wird in der zweiten Hälfte in der (Programmier-)Praxis angewendet. Dabei lernen die Teilnehmer Verwendung der Programmierparadigmen „Shared Memory“ und „Message Passing“ zuerst gemeinsam in der Programmiersprache *pseuCo*, bevor sie dann diese Fähigkeiten auf Java und teilweise Rust übertragen. Außerdem lernen die Teilnehmer verschiedene Phänomene des nebenläufigen Programmierens in den formalen Modellen zu beschreiben und mit deren Hilfe konkrete Lösungen für die Praxis abzuleiten. Des Weiteren untersuchen die Teilnehmer in der Praxis existierende Konzepte auf ihre Verlässlichkeit hin. Ein spezifischer Aspekt dieser beruflichen Praxis ist das taktisch adäquate Reagieren auf Problemstellungen der Nebenläufigkeit unter engen Zeitvorgaben.

Inhalt

Nebenläufigkeit als Konzept

- Potentieller Parallelismus
- Tatsächlicher Parallelismus
- Konzeptioneller Parallelismus

Nebenläufigkeit in der Praxis

- Objektorientierung

- Betriebssysteme
- Multi-core Prozessoren, Coprozessoren
- Programmierte Parallelität
- Verteilte Systeme (Client-Server, Peer-to-Peer, Datenbanken, Internet)

Die Schwierigkeit von Nebenläufigkeit

- Ressourcenkonflikte
- Fairness
- Gegenseitiger Ausschluss
- Verklemmung (Deadlock)
- gegenseitige Blockaden (Livelock)
- Verhungern (Starvation)

Grundlagen der Nebenläufigkeit

- Sequentielle vs. Nebenläufige Prozesse
- Zustände, Ereignisse und Transitionen
- Transitionssysteme
- Beobachtbares Verhalten
- Determinismus vs. Nicht-Determinismus
- Algebren und Operatoren

CCS: Der Kalkül kommunizierender Prozesse

- Konstruktion von Prozessen: Sequenz, Auswahl, Rekursion
- Nebenläufigkeit und Interaktion
- Strukturelle operationelle Semantik
- Gleichheit von Beobachtungen
- Implementierungsrelationen
- CCS mit Datentransfer

Programmieren von Nebenläufigkeit

- pseuCo
- Message Passing in pseuCo und Go
- Shared Memory in pseuCo und Java
- Monitore und Semaphoren
- Shared Objects und Threads in Java
- Shared Objects und Threads als Transitionssysteme

Programmier- und Analyseunterstützung

- Erkennung von Verklemmungen
- Zusicherung von Sicherheit und Lebendigkeit
- Model-Basiertes Design von Nebenläufigkeit
- Software Architekturen für Nebenläufigkeit

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Weitere Informationen

Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *Concurrent Programming*.

Informationen für "Mathematik und Informatik BSc": Diese Veranstaltung ist Teil des Wahlpflicht-Bereichs der Grundlagen der Informatik (siehe Studienordnung §5 (2) 4)

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
1	6	every winter semester	1 semester	6	9

Modulverantwortliche/r Prof. Dr.-Ing. Holger Hermanns

Dozent/inn/en Prof. Dr.-Ing. Holger Hermanns
Prof. Dr. Sebastian Hack
Prof. Dr. Martina Maggio

Zulassungsvoraussetzungen none

Leistungskontrollen / Prüfungen

- Weekly exercises and tests
- Mid-term and final exam
- Re-examination at end of the semester

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English and German

Lernziele / Kompetenzen

Programming 1 is an introduction to the foundations of computer science. Using the functional programming language OCaml, students learn to write programs, to understand precisely what programs mean, to prove them correct, and to analyse their running time. From the first day on, students work hands-on with an interpreter, which serves as a laboratory for experimenting with programs and with the programming language itself.

Upon completion of the module, students

- can write well-structured functional programs in OCaml, using recursion, tail recursion and accumulators, higher-order and polymorphic functions, pattern matching, algebraic data types, options, and exceptions;
- have acquired practical programming skills: experimenting with an interpreter, reading and reacting to type errors, systematic testing and debugging, and using the standard library;
- can design and implement recursive algorithms on numbers, lists, strings, and trees (e.g. sorting algorithms, search trees, tries, prime factorisation);
- understand static typing, polymorphism, and type inference;
- can structure programs with modules, signatures, abstract data types, representation invariants, and functors;
- can program with mutable state (references, arrays, loops, imperative data structures) and understand how data is represented in linear memory;
- can specify functions mathematically, reason about termination, and prove recursive programs correct by induction (natural, structural, and well-founded induction);
- can determine and classify the time complexity of algorithms (running-time functions, recurrences, asymptotic complexity classes);
- understand the structure of programming languages and their formal description by grammars and inference rules (abstract and concrete syntax, static and dynamic semantics);
- can implement the components of a language processor for a small functional language: lexer, a simple parser, type checker, and evaluator;
- are able to use the basic mathematical structures underlying computer science: sets, propositions, tuples, relations, directed graphs, functions, and well-founded orders.

Inhalt

see above

Literaturhinweise

Lecture notes (in English) are provided to the participants.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2	6	every summer semester	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Sebastian Hack

Dozent/inn/en Prof. Dr. Sebastian Hack
Prof. Dr. Holger Hermanns

Zulassungsvoraussetzungen *Programming 1* and *Mathematics for Computer Scientists 1* and mathematics courses in the study semester or comparable knowledge from other mathematics courses are recommended.

Leistungskontrollen / Prüfungen

- Practical part: students independently implement a series of programming projects in RISC-V assembly, C, and Java. The projects practise the language concepts of the lecture and introduce more complex algorithms and data structures. Automatic tests check the quality of the implementations.
- Lecture part: short written tests (minitests) accompanying the lecture, and a written end-of-term exam (with re-exam).
- Admission to the written exam requires successful participation in the programming projects and the minitests. The exact thresholds are announced at the beginning of the module.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in the written exams. Successful participation in the programming projects and minitests is a prerequisite for admission to the exams. The exact modalities will be announced at the beginning of the module.

Sprache English and German

Lernziele / Kompetenzen

Programming 2 introduces the foundations of imperative and object-oriented programming. The course follows the software stack from the machine level (computer arithmetic, RISC-V machine code) via the systems language C to the object-oriented language Java, and closes the loop with the formal semantics of a small imperative language and a simple compiler that translates it back to machine code. A central goal is that students can identify the different levels of abstraction at work in programming and move between them deliberately, so that they can reason about the behaviour, performance, and security of programs rather than relying on unexamined habits.

After successfully completing the module, students

- understand how computers represent numbers and execute programs, and can read, write, and debug small programs in RISC-V assembly, including function calls, the stack, and a calling convention;
- can implement, debug, and test small to mid-size C programs, and understand the C memory model, pointers, dynamic memory management, translation units, and the consequences of undefined behaviour;
- can design, implement, debug, and test mid-size Java programs using classes, interfaces, inheritance, subtyping, dynamic dispatch, encapsulation, and immutability, use the Java Collections Framework, and can judge when an object-oriented decomposition is appropriate;

- can implement fundamental data structures and algorithms (lists, trees, hash tables, dynamic programming) in an imperative language and choose suitable ones for a given programming task;
- know the formal syntax, operational semantics, and a type system of a small imperative language with pointers and scopes, and can explain type safety and undefined behaviour in these terms;
- can specify the functional correctness of programs, know the fundamental techniques and limits of testing (test oracles, black-box and white-box testing, coverage criteria, fuzzing), can prove simple imperative programs correct with Hoare logic and loop invariants, and understand how such proofs are mechanized via verification conditions;
- understand how a high-level language is translated to machine code by working through the phases of a simple compiler (syntactic analysis, type checking, syntax-directed code generation);
- have practical experience in independently completing programming projects of non-trivial size, including working with an existing code base, automatic tests, and debugging tools.

Inhalt

1. **Computer arithmetic:** positional notation, binary numbers, addition and subtraction, two's-complement integers, shifts
2. **Machine code:** the von Neumann architecture, the RV32I instruction set, the assembler, execution traces, the data segment, memory segmentation, linking, calling functions (stack, calling convention)
3. **Simple algorithms at the machine level:** number conversion, the sieve of Eratosthenes, insertion sort, evaluating polynomials with Horner's method, cycle detection (tortoise and hare)
4. **Introduction to C:** basics, execution traces, imperative variables, the C memory model, translation units, number types, control flow, expressions, calling functions, pointers, dynamic memory management, structs, input and output, undefined behaviour
5. **Algorithms and data structures:** lists, trees, hashing, dynamic programming
6. **Formal semantics:** the syntax of programming languages, abstract syntax and operational semantics of the language C0, extension by pointers (C0p) and scopes (C0b), a simple type system (C0t) and type safety
7. **Testing and verification:** functional correctness and specifications, failures, testing (the oracle problem, black-box testing, white-box testing and coverage, fuzzing), verification with Hoare logic, loop invariants, verification condition generation and mechanized verification
8. **Object-oriented programming with Java:** compilation, main method, and packages; types and reference types; classes; encapsulation; references, aliasing, and immutable objects; inheritance, subtyping, and dynamic dispatch; overriding and overloading methods; the Java Collections Framework; using object orientation properly; the expression problem and the visitor pattern; object-oriented programming in C
9. **A simple compiler:** overview of syntactic analysis, type checking, syntax-directed code generation from C0 to RISC-V

as well as lectures specifically designed for the individual programming projects (e.g. algorithmic background such as Dijkstra's algorithm for route planning or bounding volume hierarchies for ray tracing).

Literaturhinweise

Lecture notes: Sebastian Hack, *An Introduction to Imperative Programming*, <https://prog2.de/book>. Further literature will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Sommersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Verena Wolf
Prof. Dr. Vera Demberg

Dozent/inn/en Prof. Dr. Verena Wolf
Prof. Dr. Vera Demberg

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen mündliche oder schriftliche Prüfung

Lehrveranstaltungen / SWS 2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS

Arbeitsaufwand 60 h Präsenzstudium
+ 120 h Eigenstudium
= 180 h (= 6 ECTS)

Modulnote Wird aus Leistungen in der Klausur, sowie den Prüfungsvorleistungen ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben. Alle Modulelemente sind innerhalb eines Prüfungszeitraumes erfolgreich zu absolvieren.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

- Verständnis der mathematischen Konzepte von Zufallsvariablen und Verteilungen
- Verständnis und Anwendung von Methoden der Punkt- und Intervallschätzung, statistischer Tests
- Verständnis der mathematischen Konzepte von Zustandsdiskreten Markovprozessen und Verwendung solcher Prozesse zur Beschreibung von realen Phänomenen

Inhalt

Probabilities and Discrete Random Variables

- Probability
- discrete RVs
- expectation, variance and quantiles (also visualization of them)
- higher moments
- important discrete probability distributions
- Generating discrete random variates Continuous Random Variables and Laws of Large Numbers
- σ -algebras (very lightweight)
- Continuous Random Variables
- Important Continuous Distributions
- generating continuous random variates
- Chebyshev's inequality
- Weak/Strong Law of Large Numbers
- Central Limit Theorem

Multidimensional Probability Distributions

- joint probability distribution

- conditional probability distribution
- Bayes' Theorem
- covariance and correlation
- independence
- important multidimensional probability distributions

Point Estimation

- (generalized) method of moments
- maximum likelihood estimation
- Bayesian inference (posterior mean/median, MAP)
- Kernel density estimation
- OLS estimator (this is simple regression but should be mentioned here!)
- (shortly: model selection)

Interval Estimation

- confidence intervals for sample mean/variance
- confidence intervals for MLE
- bootstrap confidence interval
- Bayesian credible interval

Statistical Testing

- Level α tests (Z-Test, T-Test)
- p-value
- chi-squared tests, Fisher test
- multiple testing (Bonferroni correction, Holm-Bonferroni method, Benjamini-Hochberg, etc)

Discrete-time Markov chains (only if time)

- transient distributions
- equilibrium distributions
- Monte-Carlo simulation

HMMs

- Baum-Welch-Algorithmus
- Viterbi-Algorithmus

Literaturhinweise

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Sommersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Jan Reineke

Dozent/inn/en Prof. Dr. Jan Reineke

Zulassungsvoraussetzungen *Programming 1, Programming 2* (im selben Semester) und *Mathematik für Informatiker 1* oder vergleichbare Veranstaltungen der Mathematik sind empfohlen.

Leistungskontrollen / Prüfungen Prüfungsleistungen werden in zwei Teilen erbracht, die beide in die Endnote eingehen. Um die Gesamtveranstaltung zu bestehen, muss jeder Teil einzeln bestanden werden.

Im *Projektteil* müssen die Studierenden eine Reihe von Projekten selbstständig bearbeiten. Diese Projekte vertiefen das praktische Verständnis des Vorlesungsstoffes in den Bereichen Rechnerarchitektur und Betriebssysteme.

Im *Vorlesungsteil* müssen die Studierenden Klausuren absolvieren und Übungsaufgaben und/oder Minitests bearbeiten. Die erfolgreiche Bearbeitung der Übungsblätter beziehungsweise der Minitests ist Voraussetzung zur Teilnahme an der Klausur.

Lehrveranstaltungen / SWS 4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS

Arbeitsaufwand 90 h Präsenzstudium
+ 180 h Eigenstudium
= 270 h (= 9 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen, Minitests und praktischen Projekten ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Englisch

Lernziele / Kompetenzen

Die Studierenden sollen die Funktionsweise und die wichtigsten Eigenschaften moderner Rechnerarchitekturen und Betriebssystemen kennenlernen.

Außerdem sollen die Studierenden die der Implementierung solcher Systeme zugrundeliegenden Entwurfsprinzipien verstehen.

Inhalt

1. Rechnerarchitektur
 - a. Boolesche Algebra und Schaltkreise
 - b. Zahlendarstellungen und arithmetische Schaltkreise
 - c. Befehlssatzarchitekturen
 - d. Mikroarchitekturen, insbesondere der Entwurf eines einfachen Reduced Instruction Set Computers, sowie Techniken zur Leistungsoptimierung.
2. Betriebssysteme
 - a. Virtualisierungsmechanismen
 - b. Planungsalgorithmen
 - c. Dateisysteme

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Weitere Informationen

Dieses Modul ist inhaltsgleich mit dem englischsprachigen Modul *System Architecture*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2-3	6	lecture free time after SS	7 weeks	BLOCK	9

Modulverantwortliche/r Prof. Dr. Sven Apel

Dozent/inn/en Prof. Dr. Sven Apel
Dr. Norman Peitek

Zulassungsvoraussetzungen Participation in the Software Engineering Lab requires extensive programming skills as taught in the courses *Programming 1* and *Programming 2*. A passing grade in *Programming 2* is required to enroll in this course.

Students are required to bring their own laptops.

Leistungskontrollen / Prüfungen The goal of the Software Engineering Lab is to develop a non-trivial software system in a team effort. In this course, a number of documents (design models, documentation, implementation plan, etc.) and artifacts (source code, tests, etc.) need to be developed and submitted. Correctness, completeness, quality, and timely submission of all documents and artifacts are major grading criteria.

The Software Engineering Lab consists of two phases: exercise phase and group phase.

In the *exercise phase*, participants will complete an entry exam, covering current topics from the lecture. Only participants that have passed the exercise phase will be admitted to the group phase.

In the *group phase*, participants will first design and then implement and test a substantial software system in a team effort, and submit both their design and their implementation (including tests) for evaluation. All documents (design models, documentation, implementation plan, etc.) and artifacts (source code, tests, etc.) of the group phase will be evaluated based on the principles and quality criteria conveyed in the lectures. To pass the group phase, students must pass both the design submission and the implementation submission, and prove individually their substantial contribution to the group project.

More details on the exams will be announced at the beginning of the course.

Lehrveranstaltungen / SWS Daily exercises and lectures (first few weeks)
Daily project work with tutoring

Arbeitsaufwand 35 h of lectures and exercises
+ 235 h project work
= 270 h (= 9 ECTS)

Modulnote ungraded

Sprache English

Lernziele / Kompetenzen

Participants acquire the ability to solve complex software development problems individually and in teams.

Participants are aware of common problems and pitfalls of software development and know how to address them.

Participants are able to accomplish and coordinate software development tasks based on a set of given requirements. For this purpose, they are able to select proper methods and techniques to minimize risks and maximize software quality.

Participants know about foundations and principles of software design, including cohesion, coupling, modularity, encapsulation, abstraction, and information hiding. They are acquainted with a whole array of design patterns, knowing their aim

and individual strengths and weaknesses. They are able to apply design patterns beneficially and to judge and improve the quality of software designs.

Participants master fundamental techniques and tools for software testing, debugging, and version control.

Inhalt

- Software design
- Software testing
- Team work
- Debugging

Literaturhinweise

- Software Engineering. I. Sommerville, Addison-Wesley, 2004.
- Software Engineering: A Practitioner's Approach. R. Pressman, McGraw Hill Text, 2001.
- Using UML: Software Engineering with Objects and Components. P. Stevens, et al., Addison-Wesley, 1999.
- UML Distilled. M. Fowler, et al., Addison-Wesley, 2000.
- Objects, Components and Frameworks with UML, D. D'Souza, et al., Addison-Wesley, 1999.
- Designing Object-Oriented Software. R. Wirfs-Brock, et al., Prentice Hall, 1990.
- Design Patterns. Elements of Reusable Object-Oriented Software. E. Gamma, et al., Addison-Wesley, 1995.
- Head First Design Patterns. E. Freeman, et al. O'Reilly, 2004.
- Software Architecture: Perspectives on an Emerging Discipline. M. Shaw, et al., Prentice-Hall, 1996.
- Refactoring: Improving the Design of Existing Code. M. Fowler, et al., Addison-Wesley, 1999.
- Software Testing and Analysis: Process, Principles and Techniques. M. Pezze, Wiley. 2007.

Weitere Informationen

This module is identical in content to the German-language module *Softwarepraktikum*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
2-3	6	lecture free time after SS	7 weeks	BLOCK	9

Modulverantwortliche/r**Dozent/inn/en**

Zulassungsvoraussetzungen Participation in the Software Engineering Lab requires extensive programming skills as taught in the courses *Programming 1* and *Programming 2*. Passing *Programming 2* is recommended before enrolling in this course.

Students are required to bring their own laptops.

Leistungskontrollen / Prüfungen The goal of the Software Engineering Lab is to develop a non-trivial software system in a team effort. In this course, a number of documents (design models, documentation, implementation plan, etc.) and artifacts (source code, tests, etc.) need to be developed and submitted. Correctness, completeness, quality, and timely submission of all documents and artifacts are major grading criteria.

The Software Engineering Lab consists of two phases: exercise phase and group phase.

In the *exercise phase*, participants will complete an entry exam, covering current topics from the lecture. Only participants that have passed the exercise phase will be admitted to the group phase.

In the *group phase*, participants will first design and then implement and test a substantial software system in a team effort, and submit both their design and their implementation (including tests) for evaluation. All documents (design models, documentation, implementation plan, etc.) and artifacts (source code, tests, etc.) of the group phase will be evaluated based on the principles and quality criteria conveyed in the lectures. To pass the group phase, students must pass both the design submission and the implementation submission, and prove individually their substantial contribution to the group project.

More details on the exams will be announced at the beginning of the course.

Lehrveranstaltungen / SWS Daily exercises and lectures (first few weeks)
Daily project work with tutoring

Arbeitsaufwand 35 h of lectures and exercises
+ 235 h project work
= 270 h (= 9 ECTS)

Modulnote ungraded

Sprache English

Lernziele / Kompetenzen

Participants acquire the ability to solve complex software development problems individually and in teams.

Participants are aware of common problems and pitfalls of software development and know how to address them.

Participants are able to accomplish and coordinate software development tasks based on a set of given requirements. For this purpose, they are able to select proper methods and techniques to minimize risks and maximize software quality.

Participants know about foundations and principles of software design, including cohesion, coupling, modularity, encapsulation, abstraction, and information hiding. They are acquainted with a whole array of design patterns, knowing their aim and individual strengths and weaknesses. They are able to apply design patterns beneficially and to judge and improve the quality of software designs.

Participants master fundamental techniques and tools for software testing, debugging, and version control.

Inhalt

- Software design
- Software testing
- Team work
- Debugging

Literaturhinweise

- Software Engineering. I. Sommerville, Addison-Wesley, 2004.
- Software Engineering: A Practitioner's Approach. R. Pressman, McGraw Hill Text, 2001.
- Using UML: Software Engineering with Objects and Components. P. Stevens, et al., Addison-Wesley, 1999.
- UML Distilled. M. Fowler, et al., Addison-Wesley, 2000.
- Objects, Components and Frameworks with UML, D. D'Souza, et al., Addison-Wesley, 1999.
- Designing Object-Oriented Software. R. Wirfs-Brock, et al., Prentice Hall, 1990.
- Design Patterns. Elements of Reusable Object-Oriented Software. E. Gamma, et al., Addison-Wesley, 1995.
- Head First Design Patterns. E. Freeman, et al. O'Reilly, 2004.
- Software Architecture: Perspectives on an Emerging Discipline. M. Shaw, et al., Prentice-Hall, 1996.
- Refactoring: Improving the Design of Existing Code. M. Fowler, et al., Addison-Wesley, 1999.
- Software Testing and Analysis: Process, Principles and Techniques. M. Pezze, Wiley. 2007.

Weitere Informationen

This module is identical in content to the German-language module *Softwarepraktikum*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
1	6	jedes Wintersemester	1 Semester	6	9

Modulverantwortliche/r Prof. Dr. Antonio Krüger

Dozent/inn/en Prof. Dr. Antonio Krüger

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen Klausur und erfolgreiche Bearbeitung von Übungsblättern

Lehrveranstaltungen / SWS 4 SWS Vorlesung
+ 2 SWS Übung
= 6 SWS

Arbeitsaufwand 90 h Präsenzstudium
+ 180 h Eigenstudium
= 270 h (= 9 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden erwerben Kenntnisse im Hinblick auf Themen, Fragestellungen und Methoden der Medieninformatik. Die Studierenden erwerben die Kompetenz, wissenschaftliche Inhalte reflektiert und systematisch in mündlicher und schriftlicher Form zu präsentieren. Die Studierenden werden befähigt, bei Referaten geeignete Präsentationstechniken einzusetzen. In sozialen Arbeitszusammenhängen entwickeln die Studierenden Konflikt- und Kritikfähigkeit, Teamfähigkeit und Empathie.

Inhalt

Die Vorlesung gibt eine einführende Übersicht über die wichtigsten technischen und konzeptionellen Grundlagen zur Realisierung digitaler Medien. Es werden keinerlei Programmierkenntnisse und nur elementare Informatikkenntnisse vorausgesetzt. Dagegen wird ein relativ breites Themenspektrum verschiedenster Disziplinen (Mathematik, Psychologie, Physik, Nachrichtentechnik, Design) angesprochen.

Die Vorlesung behandelt folgende Themengebiete:

1. Grundlagen von Bildrepräsentation und -verarbeitung
2. Grundlagen der Audiorepräsentation und -verarbeitung
3. Grundlagen der Wahrnehmung
4. Text und Typographie
5. Grundlagen der Computergraphik, 2D/3D-Graphik
6. Medien und das Internet
7. Grundlagen der Mensch-Maschine Kommunikation

Die Vorlesung bildet die Basis für die stärker ins technische Detail gehende Lehrveranstaltung "Ubiquitous Media" und stellt die Grundlagen für vertiefende Lehrangebote zu Multimedia-Themen, sowie „User Interface Design“.

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekannt gegeben.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Jürgen Steimle

Dozent/inn/en Prof. Dr. Jürgen Steimle

Zulassungsvoraussetzungen undergraduate students: *Programming 1* and 2
graduate students: none

Leistungskontrollen / Prüfungen Regular attendance of classes and tutorials
Successful completion of exercises and course project
Final exam
A re-exam takes place (as written or oral examination).

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

This course teaches the theoretical and practical foundations for human computer interaction. It covers a wide overview of topics, techniques and approaches used for the design and evaluation of modern user interfaces.

The course covers the principles that underlie successful user interfaces, provides an overview of input and output devices and user interface types, and familiarizes students with the methods for designing and evaluating user interfaces. Students learn to critically assess user interfaces, to design user interfaces themselves, and to evaluate them in empirical studies.

Inhalt

- Fundamentals of human-computer interaction
- User interface paradigms, input and output devices
- Desktop & graphical user interfaces
- Mobile user interfaces
- Natural user interfaces
- User-centered interaction design
- Design principles and guidelines
- Prototyping

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Sommersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Antonio Krüger

Dozent/inn/en Prof. Dr. Antonio Krüger

Zulassungsvoraussetzungen Grundlagen der Medieninformatik (empfohlen)

Leistungskontrollen / Prüfungen Klausur und erfolgreiche Bearbeitung von Übungsblättern

Lehrveranstaltungen / SWS 2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS

Arbeitsaufwand 60 h Präsenzstudium
+ 60 h Eigenstudium (Vor- und Nachbereitung)
+ 60 h Eigenstudium (Prüfungsvorbereitung)
= 60 h (= 9 ECTS)

Modulnote Wird aus Leistungen in Klausuren, Übungen und praktischen Aufgaben ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die Studierenden erwerben Kenntnisse im Hinblick auf erweiterte Themen, Fragestellungen und Methoden der Medieninformatik. Sie lernen die Bewertung und Anwendung zukunftsweisender Medientechnologien unter Berücksichtigung des Ubiquitous Computing Paradigmas. Insbesondere werden Medientechniken vertieft, die im Zusammenhang für Tangible, und Natural User Interfaces verwendet werden.

Inhalt

Die Vorlesung gibt behandelt ein reiches Themenspektrum erweiterten Medieninformatik. Insbesondere folgende Themen werden behandelt:

1. Ubiquitous Computing
2. Neuartige Display Technologien
3. Mobile Medien und Projektorinteraktion
4. Technologien zu Tangible User Interfaces
5. Advanced Spatial and directed Audio
6. Media in Virtual Environments

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekannt gegeben.

Weitere Informationen

Das Modul war vormals unter dem Titel *Ubiquitous Media* bekannt.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5	6	jedes Wintersemester	1 Semester	3	9

Modulverantwortliche/r Prof. Dr. Antonio Krüger

Dozent/inn/en N.N.

Zulassungsvoraussetzungen Grundlagen der Medieninformatik, Programming 1 und 2 (empfohlen)

Leistungskontrollen / Prüfungen

- Wöchentliche Kurzpräsentationen
- Finale Abschlusspräsentation
- Schriftliche Ausarbeitung

Lehrveranstaltungen / SWS

1 SWS Vorlesung / Projektpräsentation
 + 2 SWS Treffen mit Projektansprechpartner
 = 3 SWS

Arbeitsaufwand

45 h Präsenzstudium
 + 210 h Projektarbeit
 + 15 h Präsentationsvorbereitung
 = 270 h (= 9 ECTS)

Modulnote Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.
 (unbenotet)

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die Studierenden erwerben im Medienprojekt theoretische und praktische bezüglich der Durchführung eines Projektes aus dem Bereich der Medieninformatik. Neben Kenntnissen der Konzeption und Modellierung von medienrelevanten Prozessen mit Methoden der Informatik, wird in dieser Projektarbeit großen Wert auf Arbeiten in interdisziplinären Teams gelegt mit 3 bis 5 Studierenden gelegt. Die Studierenden erwerben dabei die Kompetenz in solchen Teams zu arbeiten, wissenschaftliche Inhalte zu reflektieren und diese systematisch in mündlicher und schriftlicher Form zu präsentieren.

Inhalt

Das Medienprojekt widmet sich der Konzeption, Implementation und Durchführung eines komplexeren Projektes mit wechselndem Bezug zur Medieninformatik. Neben den Studierenden der Medieninformatik sollten auch Studierende der am Medieninformatikstudiengang beteiligten Spezialisierungen teilnehmen, um einen stärkeren Fachbezug herzustellen. Typische Themen eines Medienprojektes können die Konzeption, Implementierung und Evaluierung einer innovativen Mensch-Maschine-Schnittstelle sein oder der Entwurf und die Durchführung eines künstlerischen Medienprojektes oder die Entwicklung einer multimedialen virtuellen Umgebung in der medien- und wahrnehmungsspsychologische Experimente durchgeführt werden können. Die Studierenden sollen entsprechend ihrer Spezialisierung Aufgaben im Projektteam übernehmen und diesen Bereich selbstverantwortlich betreuen.

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekannt gegeben.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jedes Wintersemester	1 Semester	4	6

Modulverantwortliche/r Prof. Dr. Vera Demberg

Dozent/inn/en Prof. Dr. Vera Demberg

Zulassungsvoraussetzungen Wahrscheinlichkeitstheorie

Leistungskontrollen / Prüfungen Prüfungsvorleistung: Erfolgreiche Bearbeitung der Übungsblätter für Zulassung zur Klausur
Abschlussklausur

Lehrveranstaltungen / SWS 2 SWS Vorlesung
+ 2 SWS Übung
= 4 SWS

Arbeitsaufwand 60 h Präsenzstudium
+ 120 h Eigenstudium
= 180 h (= 6 ECTS)

Modulnote Die Note ergibt sich aus der in der Klausur erreichten Punktzahl. Die genauen Modalitäten werden vom Dozenten zu Beginn der Veranstaltung bekanntgegeben.

Sprache Englisch

Lernziele / Kompetenzen

Die Studierenden lernen zu entscheiden, wann welcher statistische Hypothesentest angewendet werden kann. Sie können für einen neuen Datensatz selbständig mit dem Statistikprogramm R Datenvisualisierung und Hypothesentests durchführen und die Testergebnisse wissenschaftlich korrekt berichten. Linear mixed effects models sollen korrekt auf experimentelle Datensätze angewendet und interpretiert werden können. Studenten sollen dabei die Angemessenheit der Anwendung statistischer Modelle bewerten können (Annahmen und Voraussetzungen zur Anwendbarkeit).

Inhalt

Wahrscheinlichkeitsverteilungen: normal, binomial, Poissonverteilung

Das Hauptaugenmerk der Veranstaltung wird auf Hypothesentests, Korrelationen, time series analyses und linearen Regression liegen. Hierbei behandelt der Kurs auch linear mixed effects models und generalized additive models. Neuere Literatur zur Modellierung von random effects und model selection wird als Teil des Kurses besprochen werden.

Ein weiterer Themenkomplex wird sich mit der Visualisierung von Daten beschäftigen, in den Übungen werden wir mit ggplot2 arbeiten.

Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jedes Wintersemester	1 Semester	4	4

Modulverantwortliche/r Dozent/inn/en der Hochschule der Bildenden Künste (HBKsaar)

Dozent/inn/en Dozent/inn/en der Hochschule der Bildenden Künste (HBKsaar)

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen Projektarbeit, Klausur und/oder mündliche Prüfung

Lehrveranstaltungen / SWS 4 SWS Grundlagen Media Art & Design

Arbeitsaufwand 60 h Präsenzstudium
 + 30 h Eigenstudium (Vor- und Nachbereitung)
 + 30 h Eigenstudium (Prüfungsvorbereitung)
 = 120 h (= 4 ECTS)

Modulnote Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.
 (unbenotet)

Sprache Deutsch

Lernziele / Kompetenzen

Kurzüberblick der vermittelten Kompetenzen:

- Fähigkeit, sich eigenständig ästhetisch differenziert zu artikulieren, und dies aufgrund anschlussfähiger künstlerisch-praktischer und gestalterischer Erfahrungen
- Erkenntnis von Potentialen der Entfaltung gestalterischer Kreativität in unterschiedlichen Kontexten, Situationen und Problemstellungen

Inhalt

Vorstellung der Gestaltungsgrundlagen des Studiengangs Media Art & Design. Die Dozenten vermitteln die spezifischen Inhalte ihrer Arbeitsbereiche anhand praktischer Beispiele und Übungen.

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekannt gegeben.

Weitere Informationen

Übungen am Computer.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Semester	1 Semester	8	8

Modulverantwortliche/r Dozent/inn/en der Hochschule der Bildenden Künste (HBKsaar)

Dozent/inn/en Dozent/inn/en der Hochschule der Bildenden Künste (HBKsaar)

Zulassungsvoraussetzungen *Grundlagen Media Art & Design* (empfohlen)

Leistungskontrollen / Prüfungen Projektarbeit, Projektdokumentation, Projektpräsentation

Lehrveranstaltungen / SWS 8 SWS Projektarbeit

Arbeitsaufwand 120 h Projektarbeit
 + 120 h Vor- und Nachbereitung
 (Projektrecherche und -dokumentation)
 = 240 h (= 8 ECTS)

Modulnote Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.
 (benotet)

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die konkreten Lernziele der einzelnen Veranstaltungen werden jährlich von der Hochschule der Bildenden Künste (HBKsaar) aktualisiert und bekannt gegeben. Kurzüberblick der vermittelten Kompetenzen:

- Reflexion diverser Medien, Materialien und Methoden
- Kritische Reflexion medialer Bildwelten und von Phänomenen der Alltagsästhetik sowie deren Darstellung in rhetorischen Grundformen
- Bewirken ästhetischer Entscheidungen in Gestaltungsprozessen sowie deren begründete Reflexion
- Entwurf, Vorbereitung und Durchführung eines eigenständig bearbeiteten Projekts
- Projektdokumentation, die das Projekt und die eigene Leistung dokumentiert
- Präsentation der eigenen Arbeit in einem Ateliergespräch oder einer Ausstellung

Inhalt

Die konkreten Inhalte orientieren sich an den angebotenen Projektarbeiten und werden jährlich von der Hochschule der Bildenden Künste (HBKsaar) aktualisiert und bekannt gegeben. Die Themen spannen einen attraktiven Bogen von aktueller Forschung zu anspruchsvollen Problemen der industriellen Praxis.

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekannt gegeben.

Weitere Informationen

Die zugehörigen Veranstaltungen werden im Vorlesungszeugnis der HBKsaar als *Atelierprojekt kurz* bezeichnet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
3	6	jährlich	2 Semester	4	8

Modulverantwortliche/r Prof. Dr. Dirk Wentura

Dozent/inn/en Prof. Dr. Dirk Wentura

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen Art der Prüfung wird zu Beginn der Vorlesung bekannt gegeben: Klausur (120 Minuten, benotet) oder mündliche Prüfung (25 – 30 Minuten, benotet); zusammenfassende Modulprüfung über den Stoff der Vorlesungen (benotet)

Lehrveranstaltungen / SWS Das Modul teilt sich auf in 2 Modulelemente:
 Teil 1 / Wintersemester:
 Allgemeine Psychologie I: Wahrnehmung und Aufmerksamkeit
2 SWS Vorlesung
 Teil 2 / Sommersemester:
 Allgemeine Psychologie I: Gedächtnis und Denken
2 SWS Vorlesung
= 4 SWS

Arbeitsaufwand Teil 1:
 30 h Präsenzstudium
 + 30 h Eigenstudium (Vor- und Nachbereitung)
 + 60 h Eigenstudium (Prüfungsvorbereitung)

 Teil 2:
 + 30 h Präsenzstudium
 + 30 h Eigenstudium (Vor- und Nachbereitung)
 + 60 h Eigenstudium (Prüfungsvorbereitung)

 = 240 h (= 8 ECTS)

Modulnote Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde. (benotet)

Sprache Deutsch

Lernziele / Kompetenzen

Kenntnisse der wesentlichen Theorien und Befunde zu den zentralen Themen der Kognitiven Psychologie, d.h. zu den Funktionsbereichen Wahrnehmung, Aufmerksamkeit, Gedächtnis und Denken und ein vertieftes Verständnis der Rolle funktionaler Modellvorstellungen und experimentalpsychologischer Methodik in der Psychologie. Studierende kennen zentrale Theorien und Ergebnisse der Kognitiven Psychologie; sie wissen wie Kognitive Psychologie funktioniert, d.h. ihnen ist die Formulierung kognitionspsychologischer Struktur und Prozessannahmen vertraut; sie kennen die Rolle der Experimentalparadigmen zur Überprüfung kognitionspsychologischer Theorien; sie können die Beziehung funktionaler Modelle zu Erkenntnissen über die neuronale Basis richtig einschätzen; sie kennen die Rolle funktionaler Modelle zum Verständnis komplexer Denkvorgänge und Handlungsprozesse; sie können Theorien und Befunde der Kognitiven Psychologie auf die Fragen anderer Teildisziplinen der Psychologie transferieren. Die Studierenden erwerben die Kompetenz, sich vertieftes Wissen aus der Fachliteratur zu erschließen. Die Studierenden werden zudem in die Lage versetzt, wissenschaftliche Theorien sowie empirische Befunde zu verstehen. Die Studierenden erwerben die Fähigkeit, analytisch zu denken, methodisch zu reflektieren sowie begründet mit Bezug auf wissenschaftliche Theorien und empirische Befunde zu argumentieren.

Inhalt

Einführung in die Kognitive Psychologie und ihre Vorgehensweise; Überblick über Theorien und zentrale Forschungsbefunde zu den Funktionsbereichen Wahrnehmung, Aufmerksamkeit, Gedächtnis und Denken.

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekannt gegeben.

Weitere Informationen

Übungen am Computer.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5	6	jedes Wintersemester	1 Semester	2	4

Modulverantwortliche/r Prof. Dr. Malte Frieze

Dozent/inn/en Prof. Dr. Malte Frieze

Zulassungsvoraussetzungen *Allgemeine Psychologie I* (empfohlen)

Leistungskontrollen / Prüfungen Referat und/oder Arbeitsaufträge und/oder Testate

Lehrveranstaltungen / SWS 2 SWS Seminar

Arbeitsaufwand 30 h Präsenzstudium
 + 30 h Eigenstudium (Vor- und Nachbereitung)
 + 60 h Eigenstudium (Prüfungsvorbereitung)
 = 120 h (= 4 ECTS)

Modulnote Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.
 (unbenotet)

Sprache Deutsch

Lernziele / Kompetenzen

Die Studierenden erwerben Kenntnisse im Hinblick auf Themen, Fragestellungen und Methoden der Medienpsychologie. Die Studierenden erwerben die Kompetenz, wissenschaftliche Inhalte reflektiert und systematisch in mündlicher und schriftlicher Form zu präsentieren. Die Studierenden werden befähigt, bei Referaten geeignete Präsentationstechniken einzusetzen. In sozialen Arbeitszusammenhängen entwickeln die Studierenden Konflikt- und Kritikfähigkeit, Teamfähigkeit und Empathie.

Inhalt

- Methodische Grundlagen der Untersuchung des Einflusses von Medien auf menschliches Denken, Fühlen und Handeln
- Auswirkungen des Konsums gewalthaltiger Medien auf aggressives Verhalten
- Auswirkungen des Konsums pornographischer Medien
- Soziale Netzwerke
- Politische Massenkommunikation

Literaturhinweise

Die Literatur zum Modul kann englisch- und/oder deutschsprachig sein und wird zu Beginn der Veranstaltung bekannt gegeben.

Seminare

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
4	6	jedes Semester	1 Semester	2	5

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Dozent/inn/en der Fachrichtung

Zulassungsvoraussetzungen Grundlegende Kenntnisse im jeweiligen Teilbereich des Studienganges.

Leistungskontrollen / Prüfungen

- Thematischer Vortrag mit anschließender Diskussion
- Aktive Teilnahme an der Diskussion
- Gegebenenfalls kurze schriftliche Ausarbeitung und/oder Projekt

Lehrveranstaltungen / SWS 2 SWS Proseminar

Arbeitsaufwand 30 h Präsenzstudium
+ 120 h Eigenstudium
= 150 h (= 5 ECTS)

Modulnote Wird aus den Leistungen im Vortrag und der schriftlichen Ausarbeitung und/oder dem Seminarprojekt ermittelt. Die genauen Modalitäten werden von dem/der jeweiligen Dozenten/in bekannt gegeben.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die Studierenden haben am Ende der Veranstaltung ein grundlegendes Verständnis aktueller oder fundamentaler Aspekte eines spezifischen Teilbereiches der Informatik erlangt.

Sie haben insbesondere grundlegende Kompetenz im eigenständigen wissenschaftlichen Recherchieren, Einordnen, Zusammenfassen, Diskutieren, Kritisieren und Präsentieren von wissenschaftlichen Erkenntnissen gewonnen.

Im Vergleich zum Seminar liegt der Fokus beim Proseminar auf der Aneignung der grundlegenden wissenschaftlichen Arbeitsweisen.

Inhalt

Unter Anleitung werden folgende Punkte praktisch geübt:

- Lesen und Verstehen wissenschaftlicher Arbeiten
- Diskutieren der Arbeiten in der Gruppe
- Analysieren, Zusammenfassen und Wiedergeben des spezifischen Themas
- Präsentationstechnik

Spezifische Vertiefung in Bezug auf das individuelle Thema des Seminars.

Der typische Ablauf eines Proseminars ist üblicherweise wie folgt:

- Vorbereitende Gespräche zur Themenauswahl
- Regelmäßige Treffen mit Diskussion ausgewählter Beiträge
- ggf. Bearbeitung eines themenbegleitenden Projekts
- Vortrag und ggf. Ausarbeitung zu einem der Beiträge

Literaturhinweise

Material wird dem Thema entsprechend ausgewählt.

Weitere Informationen

Die jeweils zur Verfügung stehenden Proseminare werden vor Beginn des Semesters angekündigt und unterscheiden sich je nach Studiengang.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5	6	jedes Semester	1 Semester	2	7

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Dozent/inn/en der Fachrichtung

Zulassungsvoraussetzungen Grundlegende Kenntnisse im jeweiligen Teilbereich des Studienganges.

Leistungskontrollen / Prüfungen

- Thematischer Vortrag mit anschließender Diskussion
- Aktive Teilnahme an der Diskussion
- Gegebenenfalls schriftliche Ausarbeitung oder Projekt

Lehrveranstaltungen / SWS 2 SWS Seminar

Arbeitsaufwand 30 h Präsenzstudium
+ 180 h Eigenstudium
= 210 h (= 7 ECTS)

Modulnote Wird aus den Leistungen im Vortrag und der schriftlichen Ausarbeitung und/oder dem Seminarprojekt ermittelt. Die genauen Modalitäten werden von dem/der jeweiligen Dozenten/in bekannt gegeben.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die Studierenden haben am Ende der Veranstaltung vor allem ein tiefes Verständnis aktueller oder fundamentaler Aspekte eines spezifischen Teilbereiches der Informatik erlangt.

Sie haben weitere Kompetenz im eigenständigen wissenschaftlichen Recherchieren, Einordnen, Zusammenfassen, Diskutieren, Kritisieren und Präsentieren von wissenschaftlichen Erkenntnissen gewonnen.

Inhalt

Weitgehend selbstständiges Erarbeiten des Seminarthemas:

- Lesen und Verstehen wissenschaftlicher Arbeiten
- Analyse und Bewertung wissenschaftlicher Aufsätze
- Diskutieren der Arbeiten in der Gruppe
- Analysieren, Zusammenfassen und Wiedergeben des spezifischen Themas
- Erarbeiten gemeinsamer Standards für wissenschaftliches Arbeit
- Präsentationstechnik

Spezifische Vertiefung in Bezug auf das individuelle Thema des Seminars.

Der typische Ablauf eines Seminars ist üblicherweise wie folgt:

- Vorbereitende Gespräche zur Themenauswahl
- Regelmäßige Treffen mit Diskussion ausgewählter Beiträge
- ggf. Bearbeitung eines themenbegleitenden Projekts
- Vortrag und ggf. Ausarbeitung zu einem der Beiträge

Literaturhinweise

Material wird dem Thema entsprechend ausgewählt.

Weitere Informationen

Die jeweils zur Verfügung stehenden Seminare werden vor Beginn des Semesters angekündigt und unterscheiden sich je nach Studiengang.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Kurt Mehlhorn

Dozent/inn/en Prof. Dr. Raimund Seidel
Prof. Dr. Kurt Mehlhorn

Zulassungsvoraussetzungen For graduate students: C, C++, Java

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Passing the midterm and the final exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

The students know standard algorithms for typical problems in the area's graphs, computational geometry, strings and optimization. Furthermore, they master a number of methods and data-structures to develop efficient algorithms and analyze their running times.

Inhalt

- graph algorithms (shortest path, minimum spanning trees, maximal flows, matchings, etc.)
- computational geometry (convex hull, Delaunay triangulation, Voronoi diagram, intersection of line segments, etc.)
- strings (pattern matching, suffix trees, etc.)
- generic methods of optimization (tabu search, simulated annealing, genetic algorithms, linear programming, branch-and-bound, dynamic programming, approximation algorithms, etc.)
- data-structures (Fibonacci heaps, radix heaps, hashing, randomized search trees, segment trees, etc.)
- methods for analyzing algorithms (amortized analysis, average-case analysis, potential methods, etc.)

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Jörg Hoffmann

Dozent/inn/en Prof. Dr. Jörg Hoffmann

Zulassungsvoraussetzungen *Programming 1, Programming 2, Fundamentals of Data Structures and Algorithms, and Elements of Machine Learning* or other courses in machine learning are recommended.

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Solving of weekly assignments
- Passing the final written exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from the performance in exams. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

Knowledge about basic methods in Artificial Intelligence

Inhalt

Search:

- Uninformed- and informed search procedures
- Monte-Carlo tree search

Planning:

- Formalism and complexity
- Critical-path heuristics
- Delete relaxation heuristics
- Abstraction heuristics

Markov decision processes:

- Discounted reward and expected cost
- Value iteration
- Informed search
- Reinforcement learning

Games:

- Adversarial search
- Learning from self-play

Literaturhinweise

Russel & Norvig Artificial Intelligence: A Modern Approach;
further reading will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Christoph Weidenbach

Dozent/inn/en Prof. Dr. Christoph Weidenbach

Zulassungsvoraussetzungen *Introduction to Computational Logic*

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Weekly assignments
- Practical work with systems
- Passing the final and mid-term exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

The goal of this course is to provide familiarity with logics, calculi, implementation techniques, and systems providing automated reasoning.

Inhalt

Propositional Logic – CDCL, Superposition - Watched Literals
First-Order Logic without Equality – (Ordered) Resolution,
Equations with Variables – Completion, Termination
First-Order Logic with Equality – Superposition (SUP) - Indexing

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Sebastian Hack

Dozent/inn/en Prof. Dr. Sebastian Hack

Zulassungsvoraussetzungen For graduate students: none

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Written exam at the end of the course, theoretical exercises, and compiler-laboratory project.
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS

4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand

90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

The students learn, how a source program is lexically, syntactically, and semantically analyzed, and how they are translated into semantically equivalent machine programs. They learn how to increase the efficiency by semantics-preserving transformations. They understand the automata-theoretic foundations of these tasks and learn, how to use the corresponding tools.

Inhalt

Lexical, syntactic, semantic analysis of source programs, code generation for abstract and real machines, efficiency-improving program transformations, foundations of program analysis.

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Markus Bläser

Dozent/inn/en Prof. Dr. Raimund Seidel
Prof. Dr. Markus Bläser

Zulassungsvoraussetzungen undergraduate course on theory of computation (e.g. *Grundzüge der Theoretischen Informatik*) is highly recommend.

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- assignments
- exams (written or oral)

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be calculated from the results in the assignments and/or exams, as announced by the Lecturer at the beginning of the course

Sprache English

Lernziele / Kompetenzen

The aim of this lecture is to learn important concepts and methods of computational complexity theory. The student shall be enabled to understand recent topics and results in computational complexity theory.

Inhalt

Relation among resources like time, space, determinism, nondeterminism, complexity classes, reduction and completeness, circuits and nonuniform complexity classes, logarithmic space and parallel complexity classes, Immerman-Szelepcsényi theorem, polynomial time hierarchy, relativization, parity and the polynomial methods, Valiant-Vazirani theorem, counting problems and classes, Toda's theorem, probabilistic computations, isolation lemma and parallel algorithms for matching, circuit identity testing, graph isomorphism and interactive proofs.

Literaturhinweise

Arora, Barak: Computational Complexity – A Modern Approach, Cambridge University Press
Oded Goldreich: Computational Complexity – A Conceptual Approach, Cambridge University Press
Dexter Kozen: Theory of Computation, Springer
Schöning, Pruim: Gems of Theoretical Computer Science, Springer

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Frank-Olaf Schreyer

Dozent/inn/en Prof. Dr. Frank-Olaf Schreyer

Zulassungsvoraussetzungen For graduate students: none

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Solving the exercises, passing the midterm and the final exam.

Lehrveranstaltungen / SWS

4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand

90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

Solving problems occurring in computer algebra praxis
The theory behind algorithms

Inhalt

Arithmetic and algebraic systems of equations in geometry, engineering and natural sciences

- integer and modular arithmetics, prime number tests
- polynomial arithmetics and factorization
- fast Fourier-transformation, modular algorithms
- resultants, Gröbnerbasen
- homotopy methods for numerical solving
- real solutions, Sturm chains and other rules for algebraic signs
- integer and modular arithmetics, prime number tests
- polynomial arithmetics and factorization
- fast Fourier-transformation, modular algorithms
- resultants, Gröbnerbasen
- homotopy methods for numerical solving
- real solutions, Sturm chains and other rules for algebraic signs

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Philipp Slusallek

Dozent/inn/en Prof. Dr. Philipp Slusallek

Zulassungsvoraussetzungen Solid knowledge of linear algebra is recommended.

Leistungskontrollen / Prüfungen

- Successful completion of weekly exercises (30% of final grade)
- Successful participation in rendering competition (10%)
- Mid-term written exam (20%, final exam prerequisite)
- Final written exam (40%)
- In each of the above a minimum of 50% is required to pass

A re-exam typically takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote The grade is derived from the above assessments. Possible changes will be announced at the beginning of each semester.

Sprache English

Lernziele / Kompetenzen

This course provides the theoretical and practical foundation for computer graphics. It gives a wide overview of topics, techniques, and approaches used in various aspects of computer graphics but has some focus on image synthesis or rendering. The first part of the course uses ray tracing as a driving applications to discuss core topics of computer graphics, from vector algebra all the way to sampling theory, the human visual system, sampling theory, and spline curves and surfaces. A second part then uses rasterization approach as a driving example, introducing the camera transformation, clipping, the OpenGL API and shading language, plus advanced techniques.

As part of the practical exercises the students incrementally build their own ray tracing system. Once the basics have been covered, the students participate in a rendering competition. Here they can implement their favorite advanced algorithm and are asked to generate a high-quality rendered image that shows their techniques in action.

Inhalt

- Introduction
- Overview of Ray Tracing and Intersection Methods
- Spatial Index Structures
- Vector Algebra, Homogeneous Coordinates, and Transformations
- Light Transport Theory, Rendering Equation
- BRDF, Materials Models, and Shading
- Texturing Methods
- Spectral Analysis, Sampling Theory
- Filtering and Anti-Aliasing Methods

- Recursive Ray Tracing & Distribution Ray-Tracing
- Human Visual System & Color Models
- Spline Curves and Surfaces
- Camera Transformations & Clipping
- Rasterization Pipeline
- OpenGL API & GLSL Shading
- Volume Rendering (opt.)

Literaturhinweise

Will be announced in the lecture.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Dr. Nico Döttling

Dozent/inn/en Prof. Dr. Cas Cremers
 Dr. Nico Döttling
 Dr. Antoine Joux
 Dr. Lucjan Hanzlik
 Dr. Julian Loss

Zulassungsvoraussetzungen For graduate students: Basic knowledge in theoretical computer science required, background knowledge in number theory and complexity theory helpful

Leistungskontrollen / Prüfungen

- Oral / written exam (depending on the number of students)
- A re-exam is normally provided (as written or oral examination).

Lehrveranstaltungen / SWS 4 h lectures
 + 2 h tutorial
 = 6 h (weekly)

Arbeitsaufwand 90 h of classes
 + 180 h private study
 = 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

The students will acquire a comprehensive knowledge of the basic concepts of cryptography and formal definitions. They will be able to prove the security of basic techniques.

Inhalt

- Symmetric and asymmetric encryption
- Digital signatures and message authentication codes
- Information theoretic and complexity theoretic definitions of security, cryptographic reduction proofs
- Cryptographic models, e.g. random oracle model
- Cryptographic primitives, e.g. trapdoor-one-way functions, pseudo random generators, etc.
- Cryptography in practice (standards, products)
- Selected topics from current research

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Martina Maggio

Dozent/inn/en Prof. Dr. Martina Maggio

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen

- Written exam at the end of the course.
- A re-exam takes place before the start of the following semester.

Lehrveranstaltungen / SWS

4 h lectures
+ 2 h tutorials
= 6 h (weekly)

Arbeitsaufwand

75 h lectures
+ 15 h mandatory assignments
+ 180 h individual study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams and assignments. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

By completing the Cyber-Physical Systems course, students will acquire the ability to model, analyze, control, and implement embedded systems that interact with the physical world, equipping them to design reliable and efficient systems for a variety of applications in modern technology.

Inhalt

Cyber-Physical Systems are embedded systems that integrate computation with physical processes. These systems are ubiquitous in our daily lives, powering technologies such as smart watches, household appliances, mobile phones, and automotive control systems. In fact, the majority of modern computing devices are embedded systems, with an estimated 98% of new CPUs being embedded in larger systems.

This course provides a comprehensive foundation for understanding, designing, and programming cyber-physical systems, emphasizing their theoretical and practical aspects. It is structured into three interconnected parts:

1. *Models*: Students will learn how to represent the physical systems that embedded systems interact with, exploring dynamical systems in both continuous and discrete time. Additionally, the course will briefly introduce more advanced models, which combine discrete state systems with dynamical systems.
2. *Control*: This module focuses on principles for modifying the behavior of physical systems through computation. Students will study and apply control techniques such as state feedback and PID control, learning how these methods influence the interaction between embedded systems and their environments.
3. *Implementation*: The final course part addresses practical challenges in embedded systems programming. Topics include scheduling, communication, and fault tolerance. This ensures that students are equipped to implement robust and efficient embedded systems in real-world scenarios.

By the end of this course, students will possess the skills needed to design and implement cyber-physical systems that meet specific functional and performance requirements, preparing them for roles in cutting-edge industries where embedded systems play a critical role, such as the automotive industry and for research in the cyber-physical systems domain.

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Weitere Informationen

This module was formerly also known as *Embedded Systems*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr.-Ing. Holger Hermanns

Dozent/inn/en Prof. Dr.-Ing. Holger Hermanns
Prof. Dr. Anja Feldmann

Zulassungsvoraussetzungen For graduate students: none

Leistungskontrollen / Prüfungen

- Qualification for midterm through a multiple choice test
- Qualification for final exam through a multiple choice test and midterm exam
- Possibility to get bonus points through excellent results in the midterm exam
- Final exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

After taking the course students have

- a thorough knowledge regarding the basic principles of communication networks,
- the fundamentals of protocols and concepts of protocol,
- Insights into fundamental motivations of different pragmatics of current network solutions,
- Introduction to practical aspects of data networks focusing on internet protocol hierarchies

Inhalt

Introduction and overview

Cross section:

- Stochastic Processes, Markov models,
- Fundamentals of data network performance assessment
- Principles of reliable data transfer
- Protocols and their elementary parts
- Graphs and Graphalgorithms (maximal flow, spanning tree)
- Application layer:
- Services and protocols
- FTP, Telnet
- Electronic Mail (Basics and Principles, SMTP, POP3, ..)
- World Wide Web (History, HTTP, HTML)

- Transport Layer:
 - Services and protocols
 - Addressing
 - Connections and ports
 - Flow control
 - QoS
 - Transport Protocols (UDP, TCP, SCTP, Ports)
- Network layer:
 - Services and protocols
 - Routing algorithms
 - Congestion Control
 - Addressing
 - Internet protocol (IP)
- Data link layer:
 - Services and protocols
 - Medium access protocols: Aloha, CSMA (-CD/CA), Token passing
 - Error correcting codes
 - Flow control
 - Applications: LAN, Ethernet, Token Architectures, WLAN, ATM
- Physical layer
- Peer-to-Peer and Ad-hoc Networking Principles

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Jens Dittrich

Dozent/inn/en Prof. Dr. Jens Dittrich

Zulassungsvoraussetzungen especially Saarland University CS department's undergraduate lecture *Big Data Engineering* (former *Informationssysteme*), *Programming 1* and *2*, *Algorithmen und Datenstrukturen* as well as *Nebenläufige Programmierung*

For graduate students:

- motivation for databases and database management systems;
- the relational data model;
- relational query languages, particularly relational algebra and SQL;
- **solid** programming skills in Java and/or C++
- undergrad courses in algorithms and data structures, concurrent programming

Leistungskontrollen / Prüfungen

- Passing a two-hour written exam at the end of the semester
- Successful demonstration of programming project (teams of up to three students are allowed); the project may be integrated to be part of the weekly assignments

Grades are based on written exam; 50% in weekly assignments (in paper and additionally paper or electronic quizzes) must be passed to participate in the final and repetition exams.

A repetition exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

This class may be run as a flipped classroom, i.e. 2 hours of lectures may be replaced by self-study of videos/papers; the other 2 hours may be used to run a group exercise supervised by the professor called "the LAB")

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined based on project, midterm and best of endterm and reexam.

Sprache English

Lernziele / Kompetenzen

Database systems are the backbone of most modern information systems and a core technology without which today's economy – as well as many other aspects of our lives – would be impossible in their present forms. The course teaches the architectural and algorithmic foundations of modern database management systems (DBMS), focussing on database systems internals rather than applications. Emphasis is made on robust and time-tested techniques that have led databases to be considered a mature technology and one of the greatest success stories in computer science. At the same time, opportunities for exciting research in this field will be pointed out.

In the exercise part of the course, important components of a DBMS will be treated and where possible implemented and their performance evaluated. The goal this is to work with the techniques introduced in the lecture and to understand them and their practical implications to a depth that would not be attainable by purely theoretical study.

Inhalt

The course "Database Systems" will introduce students to the internal workings of a DBMS, in particular:

- storage media (disk, flash, main memory, caches, and any other future storage medium)
- data managing architectures (DBMS, streams, file systems, clouds, appliances)
- storage management (DB-file systems, raw devices, write-strategies, differential files, buffer management)
- data layouts (horizontal and vertical partitioning, columns, hybrid mappings, compression, defragmentation)
- indexing (one- and multidimensional, tree-structured, hash-, partition-based, bulk-loading and external sorting, differential indexing, read- and write-optimized indexing, data warehouse indexing, main-memory indexes, sparse and dense, direct and indirect, clustered and unclustered, main memory versus disk and/or flash-based)
- processing models (operator model, pipeline models, push and pull, block-based iteration, vectorization, query compilation)
- processing implementations (join algorithms for relational data, grouping and early aggregation, filtering)
- query processing (scanning, plan computation, SIMD)
- query optimization (query rewrite, cost models, cost-based optimization, join order, join graph, plan enumeration)
- data recovery (single versus multiple instance, logging, ARIES)
- parallelization of data and queries (horizontal and vertical partitioning, shared-nothing, replication, distributed query processing, NoSQL, MapReduce, Hadoop and/or similar and/or future systems)
- read-optimized system concepts (search engines, data warehouses, OLAP)
- write-optimized system concepts (OLTP, streaming data)
- management of geographical data (GIS, google maps and similar tools)
- main-memory techniques

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr.-Ing. Thorsten Herfet

Dozent/inn/en Prof. Dr.-Ing. Thorsten Herfet

Zulassungsvoraussetzungen The lecture requires a solid foundation of mathematics (differential and integral calculus) and probability theory. The course will, however, refresh those areas indispensably necessary for telecommunications and potential intensification courses and by this open this potential field of intensification to everyone of you.

Leistungskontrollen / Prüfungen Regular attendance of classes and tutorials
 Passing the final exam in the 2nd week after the end of courses.
 Eligibility: Weekly exercises / task sheets, grouped into two blocks corresponding to first and second half of the lecture. Students must provide min. 50% grade in each of the two blocks to be eligible for the exam.

Lehrveranstaltungen / SWS 4 h lectures
 + 2 h tutorial
 = 6 h (weekly)

Arbeitsaufwand 90 h of classes
 + 180 h private study
 = 270 h (= 9 ECTS)

Modulnote Final exam mark

Sprache English

Lernziele / Kompetenzen

Digital Signal Transmission and Signal Processing refreshes the foundation laid in "Signals and Systems" [Modulkennung]. Including, however, the respective basics so that the various facets of the introductory study period (Bachelor in Computer Science, Vordiplom Computer- und Kommunikationstechnik, Elektrotechnik or Mechatronik) and the potential main study period (Master in Computer Science, Diplom-Ingenieur Computer- und Kommunikationstechnik or Mechatronik) will be paid respect to.

Inhalt

As the basic principle, the course will give an introduction into the various building blocks that modern telecommunication systems do incorporate. Sources, sinks, source and channel coding, modulation and multiplexing are the major keywords, but we will also deal with dedicated pieces like A/D- and D/A-converters and quantizers in a little bit more depth.

The course will refresh the basic transformations (Fourier, Laplace) that give access to system analysis in the frequency domain, it will introduce derived transformations (z, Hilbert) for the analysis of discrete systems and modulation schemes and it will briefly introduce algebra on finite fields to systematically deal with error correction schemes that play an important role in modern communication systems.

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Weitere Informationen

This module was formerly also known as *Telecommunications I*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Peter Druschel, Ph.D.

Dozent/inn/en Prof. Peter Druschel, Ph.D.
Allen Clement, Ph.D

Zulassungsvoraussetzungen *Operating Systems or Concurrent Programming*

Leistungskontrollen / Prüfungen

- Regular attendance at classes and tutorials.
- Successful completion of a course project in teams of 2 students. (Project assignments due approximately every 2 weeks.)
- Passing grade on 2 out of 3 written exams: midterm, final exam, and a re-exam that takes place during the last two weeks before the start of lectures in the following semester.
- Final course grade: 50% project, 50% best 2 out of 3 exams.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

Introduction to the principles, design, and implementation of distributed systems.

Inhalt

- Communication: Remote procedure call, distributed objects, event notification, Inhalt dissemination, group communication, epidemic protocols.
- Distributed storage systems: Caching, logging, recovery, leases.
- Naming. Scalable name resolution.
- Synchronization: Clock synchronization, logical clocks, vector clocks, distributed snapshots.
- Fault tolerance: Replication protocols, consistency models, consistency versus availability trade-offs, state machine replication, consensus, Paxos, PBFT.
- Peer-to-peer systems: consistent hashing, self-organization, incentives, distributed hash tables, Inhalt distribution networks.
- Data centers. Architecture and infrastructure, distributed programming, energy efficiency.

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Hans-Peter Seidel

Dozent/inn/en Prof. Dr. Hans-Peter Seidel
Dr. Rhaleb Zayer

Zulassungsvoraussetzungen calculus and basic programming skills

Leistungskontrollen / Prüfungen

- Regular attendance and participation.
- Weekly Assignments (10% bonus towards the course grade; bonus points can only improve the grade; they do not affect passing)
- Passing the written exams (mid-term and final exam).
- The mid-term and the final exam count for 50% each, but 10% bonus from assignments will be added.
- A re-exam takes place at the end of the semester break or early in the next semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Practical assignments in groups of 3 students (practice)
Tutorials consists of a mix of theoretical + practical assignments.

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be based on the performance in exams, exercises and practical tasks. The detailed terms will be announced by the module coordinator.

Sprache English

Lernziele / Kompetenzen

Gaining knowledge of the theoretical aspect of geometric modelling problems, and the practical solutions used for modelling and manipulating curves and surfaces on a computer. From a broader perspective: Learning how to represent and interact with geometric models in a discretized, digital form (geometric representations by functions and samples; design of linear function spaces; finding “good” functions with respect to a geometric modelling task in such spaces).

Inhalt

- Differential geometry Fundamentals
- Interpolation and Approximation
- Polynomial Curves
- Bezier and Rational Bezier Curves
- B-splines, NURBS
- Spline Surfaces
- Subdivision and Multiresolution Modelling
- Mesh processing
- Approximation of differential operators
- Shape Analysis and Geometry Processing

Literaturhinweise

Will be announced before the term begins on the lecture website.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Joachim Weickert

Dozent/inn/en Prof. Dr. Joachim Weickert

Zulassungsvoraussetzungen Undergraduate mathematics (e.g. Mathematik für Informatiker I-III) and elementary programming knowledge in C

Leistungskontrollen / Prüfungen

- For the homework assignments one can obtain up to 24 points per week. Actively participating in the classroom assignments gives 12 more points per week, regardless of the correctness of the solutions. To qualify for both exams one needs 2/3 of all possible points.
- Passing the final exam or the re-exam.
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from the performance in the exam or the re-exam. The better grade counts.

Sprache English

Lernziele / Kompetenzen

Broad introduction to mathematical methods in image processing and computer vision. The lecture qualifies students for a bachelor thesis in this field. Together with the completion of advanced or specialised lectures (9 credits at least) it is the basis for a master thesis in this field.

Inhalt

Inhalt

1. Basics
 - 1.1 Image Types and Discretisation
 - 1.2 Degradations in Digital Images
2. Colour Perception and Colour Spaces
3. Image Transformations
 - 3.1 Continuous Fourier Transform
 - 3.2 Discrete Fourier Transform
 - 3.3 Image Pyramids
 - 3.4 Wavelet Transform
4. Image Compression
5. Image Interpolation
6. Image Enhancement
 - 6.1 Point Operations

- 6.2 Linear Filtering and Feature Detection
- 6.3 Morphology and Median Filters
- 6.3 Wavelet Shrinkage, Bilateral Filters, NL Means
- 6.5 Diffusion Filtering
- 6.6 Variational Methods
- 6.7 Deconvolution Methods
- 7. Texture Analysis
- 8. Segmentation
 - 8.1 Classical Methods
 - 8.2 Variational Methods
- 9. Image Sequence Analysis
 - 9.1 Local Methods
 - 9.2 Variational Methods
- 10. 3-D Reconstruction
 - 10.1 Camera Geometry
 - 10.2 Stereo
 - 10.3 Shape-from-Shading
- 11. Object Recognition
 - 11.1 Hough Transform
 - 11.2 Invariants
 - 11.3 Eigenspace Methods

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Gerhard Weikum

Dozent/inn/en Prof. Dr. Gerhard Weikum

Zulassungsvoraussetzungen Good knowledge of undergraduate mathematics (linear algebra, probability theory) and basic algorithms.

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutor groups
- Presentation of solutions in tutor groups
- Passing 2 of 3 written tests (after each third of the semester)
- Passing the final exam (at the end of the semester)

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined by the performance in written tests, tutor groups, and the final exam. Details will be announced on the course web site.

Sprache English

Lernziele / Kompetenzen

The lecture teaches models and algorithms that form the basis for search engines and for data mining and data analysis tools.

Inhalt

Information Retrieval (IR) and Data Mining (DM) are methodologies for organizing, searching and analyzing digital contents from the web, social media and enterprises as well as multivariate datasets in these contexts. IR models and algorithms include text indexing, query processing, search result ranking, and information extraction for semantic search. DM models and algorithms include pattern mining, rule mining, classification and recommendation. Both fields build on mathematical foundations from the areas of linear algebra, graph theory, and probability and statistics.

Literaturhinweise

Will be announced on the course web site.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Gert Smolka

Dozent/inn/en Prof. Dr. Gert Smolka

Zulassungsvoraussetzungen keine

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials.
- Passing the midterm and the final exam.

Lehrveranstaltungen / SWS

4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand

90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

- structure of logic languages based on type theory
- distinction notation / syntax / semantics
- structure and formal representation of mathematical statements
- structure and formal representation of proofs (equational and natural deduction)
- solving Boolean equations
- proving formulas with quantifiers
- implementing syntax and deduction

Inhalt

Type Theory:

- functional representation of mathematical statements
- simply typed lambda calculus, De Bruijn representation and substitution, normalization, elimination of lambdas
- Interpretations and semantic consequence
- Equational deduction, soundness and completeness
- Propositional Logic
- Boolean Axioms, completeness for 2-valued interpretation
- resolution of Boolean equations, canonical forms based on decision trees and resolution

Predicate Logic (higher-order):

- quantifier axioms
- natural deduction
- prenex and Skolem forms

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Isabel Valera

Dozent/inn/en Prof. Dr. Isabel Valera

Zulassungsvoraussetzungen The lecture gives a broad introduction into machine learning methods. After the lecture the students should be able to solve and analyze learning problems.

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials.
- 50% of all points of the exercises have to be obtained in order to qualify for the exam.
- Passing 1 out of 2 exams (final, re-exam).

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Determined from the results of the exams, exercises and potential projects. The exact grading modalities are announced at the beginning of the course.

Sprache English

Lernziele / Kompetenzen

The lecture gives a broad introduction into machine learning methods. After the lecture the students should be able to solve and analyze learning problems.

Inhalt

- Bayesian decision theory
- Linear classification and regression
- Kernel methods
- Bayesian learning
- Semi-supervised learning
- Unsupervised learning
- Model selection and evaluation of learning methods
- Statistical learning theory
- Other current research topics

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Peter Druschel, Ph.D.

Dozent/inn/en Prof. Peter Druschel, Ph.D.
Björn Brandenburg, Ph.D

Zulassungsvoraussetzungen For graduate students: none

Leistungskontrollen / Prüfungen Regular attendance at classes and tutorials
Successful completion of a course project in teams of 2 students
Passing 2 written exams (midterm and final exam)
A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

Introduction to the principles, design, and implementation of operating systems

Inhalt

Process management:

- Threads and processes, synchronization
- Multiprogramming, CPU Scheduling
- Deadlock

Memory management:

- Dynamic storage allocation
- Sharing main memory
- Virtual memory

I/O management:

- File storage management
- Naming
- Concurrency, Robustness, Performance

Virtual machines

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Kurt Mehlhorn

Dozent/inn/en Prof. Dr. Kurt Mehlhorn
Dr. Andreas Karrenbauer

Zulassungsvoraussetzungen For graduate students: none

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Solving accompanying exercises, successful participation in midterm and final exam
- Grades: Yes
- The grade is calculated from the above parameters according to the following scheme: 20%, 30%, 50%
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

The students learn to model and solve optimization problems from theory as from the real world

Inhalt

Linear Programming: Theory of polyhedra, simplex algorithm, duality, ellipsoid method * Integer linear programming: Branch-and-Bound, cutting planes, TDI-Systems * Network flow: Minimum cost network flow, minimum mean cycle cancellation algorithm, network simplex method * Matchings in graphs: Polynomial matching algorithms in general graphs, integrality of the matching polytope, cutting planes * Approximation algorithms: LP-Rounding, greedy methods, knapsack, bin packing, steiner trees and forests, survivable network design

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Michael Backes

Dozent/inn/en Prof. Dr. Michael Backes
Prof. Dr. Cas Cremers

Zulassungsvoraussetzungen For graduate students: none

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Passing the final exam
- A re-exam is normally provided (as written or oral examination).

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined by the performance in exams, tutor groups, and practical tasks.
Details will be announced by the lecturer at the beginning of the course.

Sprache English

Lernziele / Kompetenzen

Description, assessment, development and application of security mechanisms, techniques and tools.

Inhalt

- Basic Cryptography,
- Specification and verification of security protocols,
- Security policies: access control, information flow analysis,
- Network security,
- Media security,
- Security engineering

Literaturhinweise

Will be announced on the course website

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Gert Smolka

Dozent/inn/en Prof. Dr. Gert Smolka

Zulassungsvoraussetzungen For graduate students: core lecture Introduction to Computational Logic

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials.
- Passing the midterm and the final exam

Lehrveranstaltungen / SWS

4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand

90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

Understanding of

- Logical structure of programming languages
- Formal models of programming languages
- Type and module systems for programming languages

Inhalt

Theory of programming languages, in particular:

- Formal models of functional and object-oriented languages
- Lambda Calculi (untyped, simply typed, System F, F-omega, Lambda Cube, subtyping, recursive types, Curry-Howard Correspondence)
- Algorithms for type checking and type reconstruction

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr. Sven Apel

Dozent/inn/en Prof. Dr. Sven Apel

Zulassungsvoraussetzungen

- Knowledge of programming concepts as taught in *Programming1* and *Programming2* is recommended.
- Basic knowledge of software processes, design, and testing as taught and applied in the module *Software Engineering Lab* is recommended.

Leistungskontrollen / Prüfungen

- Beside the lecture there are voluntary weekly practical exercises and tutorials covering the lecture topics.
- There will be a mandatory exam admission test in the middle of the semester that students have to pass in order to be admitted to the final examination. The exam admission test covers basic concepts of the lecture material.
- The final examination is a written exam. The exam will cover the topics of the lecture and the exercises.
- Further examination details will be announced at the beginning of the course.

Lehrveranstaltungen / SWS

4 h lectures
+ 2 h exercises
= 6 h (weekly)

Arbeitsaufwand

90 h of classes and exercises
+ 180 h private study and assignments
= 270 h (= 9 ECTS)

Modulnote

- The grade is determined by the written exam.
- The practical exercises, tutorials and the exam admission test do not contribute to the final grade.
- Passing the exam admission test is a prerequisite for taking the final exam.
- Further examination details will be announced by the lecturer at the beginning of the course.

Sprache English

Lernziele / Kompetenzen

- The students know and apply modern software development techniques.
- They are aware of key factors contributing to the complexity of real-world software systems, in particular, software variability, configurability, feature interaction, crosscutting concerns, and how to address them.
- They know how to apply established design and implementation techniques to master software complexity.
- They are aware of advanced design and implementation techniques, including collaboration-based design, parametric polymorphism, traits and mixins.
- They are aware of advanced quality assurance techniques that take the complexity of real-world software systems into account: variability-aware analysis, sampling, feature-interaction detection, predictive performance modeling, etc.
- They appreciate the role of non-functional properties and know how to predict and optimize software systems regarding these properties.
- They are able to use formal methods to reason about key techniques and properties covered in the lecture

Inhalt

- Domain analysis, feature modeling
- Automated reasoning about software configuration using SAT solvers
- Runtime parameters, design patterns, frameworks
- Version control, build systems, preprocessors
- Collaboration-based design
- Parametric polymorphism, traits, mixins
- Expression problem, preplanning problem, code scattering & tangling, tyranny of the dominant decomposition, inheritance vs. delegation vs. mixin composition
- Feature interaction problem (structural, control- & data-flow, behavioral, non-functional feature interactions)
- Variability-aware analysis and variational program representation (with applications to type checking and static program analysis)
- Sampling (random, coverage)
- Machine learning for software performance prediction and optimization

Literaturhinweise

See: https://www.infomath-bib.de/tmp/vorlesungen/info-core_software-engineering.html

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
5-6	6	at least every two years	1 semester	6	9

Modulverantwortliche/r Prof. Dr.-Ing. Holger Hermanns

Dozent/inn/en Prof. Dr.-Ing. Holger Hermanns
Prof. Bernd Finkbeiner, Ph.D

Zulassungsvoraussetzungen For graduate students: none

Leistungskontrollen / Prüfungen

- Regular attendance of classes and tutorials
- Passing the final exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

Lehrveranstaltungen / SWS 4 h lectures
+ 2 h tutorial
= 6 h (weekly)

Arbeitsaufwand 90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)

Modulnote Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

Sprache English

Lernziele / Kompetenzen

The students become familiar with the standard methods in computer-aided verification. They understand the theoretical foundations and are able to assess the advantages and disadvantages of different methods for a specific verification project. The students gain first experience with manual correctness proofs and with the use of verification tools.

Inhalt

- models of computation and specification languages: temporal logics, automata over infinite objects, process algebra
- deductive verification: proof systems (e.g., Floyd, Hoare, Manna/Pnueli), relative completeness, compositionality
- model checking: complexity of model checking algorithms, symbolic model checking, abstraction case studies

Literaturhinweise

Will be announced before the start of the course on the course page on the Internet.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
6	6	jedes Semester	variabel	2	9

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Dozent/inn/en der Fachrichtung

Zulassungsvoraussetzungen Erwerb von mindestens 120 CP

Leistungskontrollen / Prüfungen

- Schriftliche Ausarbeitung der Aufgabenstellung der Bachelorarbeit und der relevanten wissenschaftlichen Literatur
- Vortrag über die geplante Aufgabenstellung mit anschliessender Diskussion
- Aktive Teilnahme an der Diskussion

Lehrveranstaltungen / SWS 2 SWS Seminar

Arbeitsaufwand

- 30 h Präsenzstudium (Seminar)
- + 30 h Betreuung durch den Lehrstuhl
- + 210 h Eigenstudium
- = 270 h (= 9 ECTS)

Modulnote Wird aus den Leistungen im Vortrag und der schriftlichen Ausarbeitung ermittelt. Die genauen Modalitäten werden von dem/der jeweiligen Dozenten/in bekannt gegeben.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Im Bachelorseminar erwirbt der Studierende unter Anleitung die Fähigkeit zum wissenschaftlichen Arbeiten im Kontext eines angemessenen Themengebietes.

Am Ende des Bachelorseminars sind die Grundlagen für eine erfolgreiche Anfertigung der Bachelorarbeit gelegt, und wesentliche Lösungsansätze bereits eruiert.

Das Bachelorseminar bereitet somit die Themenstellung und Ausführung der Bachelorarbeit vor.

Es vermittelt darüber hinaus praktische Fähigkeiten des wissenschaftlichen Diskurses. Diese Fähigkeiten werden durch die aktive Teilnahme an einem Lesekreis vermittelt, in welchem die Auseinandersetzung mit wissenschaftlich anspruchsvollen Themen geübt wird.

Inhalt

Einarbeitung in ein wissenschaftliches Themengebiet innerhalb der Informatik.

Anfertigung einer schriftlichen Ausarbeitung der Aufgabenstellung der Bachelorarbeit und der relevanten wissenschaftlichen Literatur.

Fachvortrag über das Themengebiet und die geplante Aufgabenstellung der Bachelorarbeit.

Das Thema wird in enger Absprache mit dem anleitenden Dozenten definiert.

Literaturhinweise

Dem Themengebiet entsprechende wissenschaftliche Artikel in enger Absprache mit dem Dozenten

Bachelor-Arbeit

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
6	6	jedes Semester	3 Monate	-	12

Modulverantwortliche/r Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

Dozent/inn/en Professoren der Fachrichtung

Zulassungsvoraussetzungen Erfolgreicher Abschluss des *Bachelor-Seminars*

Leistungskontrollen / Prüfungen Schriftliche Ausarbeitung. Sie beschreibt sowohl das Ergebnis der Arbeit als auch den Weg, der zu dem Ergebnis führte. Der eigene Anteil an den Ergebnissen muss klar erkennbar sein. Außerdem Präsentation der Bachelorarbeit in einem Kolloquium, in dem auch die Eigenständigkeit der Leistung des Studierenden überprüft wird.

Lehrveranstaltungen / SWS keine

Arbeitsaufwand 30 h Betreuung durch den Lehrstuhl
+ 330 h Eigenstudium
= 360 h (= 12 ECTS)

Modulnote Beurteilung der Bachelorarbeit durch die Gutachter.

Sprache Deutsch oder Englisch

Lernziele / Kompetenzen

Die Bachelor-Arbeit ist eine Projektarbeit, die unter Anleitung ausgeführt wird. Sie soll der Kandidaten/die Kandidatin in der Lage versetzen, innerhalb einer vorgegebenen Frist ein Problem aus dem Gebiet der Informatik selbständig zu lösen und die Ergebnisse in wissenschaftlich angemessener Form zu dokumentieren.

Inhalt

Bearbeitung einer aktuellen Problemstellung aus der Informatik unter Anleitung. Adäquate Dokumentation der Ergebnisse in Form einer wissenschaftlichen Abschlussarbeit.

Das Thema wird in enger Absprache mit dem anleitenden Dozenten definiert.

Literaturhinweise

Dem Themengebiet entsprechende wissenschaftliche Artikel in enger Absprache mit dem anleitenden Dozenten.