



UNIVERSITÄT  
DES  
SAARLANDES

FAKULTÄT FÜR MATHEMATIK UND INFORMATIK

MODULHANDBUCH

**Informatik BSc**

20. April 2020

---

## **Liste der Modulbereiche und Module**

<b>1 Ringvorlesungen</b>	<b>3</b>
1.1 Perspektiven der Informatik . . . . .	4
<b>2 Grundlagen der Mathematik</b>	<b>5</b>
2.1 Mathematik für Informatiker 1 . . . . .	6
2.2 Mathematik für Informatiker 2 . . . . .	8
2.3 Mathematik für Informatiker 3 . . . . .	10
<b>3 Grundlagen der Informatik</b>	<b>12</b>
3.1 Big Data Engineering . . . . .	13
3.2 Elements of Machine Learning . . . . .	16
3.3 Grundzüge der Theoretischen Informatik . . . . .	18
3.4 Grundzüge von Algorithmen und Datenstrukturen . . . . .	19
3.5 Nebenläufige Programmierung . . . . .	20
3.6 Programmierung 1 . . . . .	22
3.7 Programmierung 2 . . . . .	23
3.8 Systemarchitektur . . . . .	25
<b>4 Praktika</b>	<b>26</b>
4.1 Softwarepraktikum . . . . .	27
<b>5 Seminare</b>	<b>29</b>
5.1 Proseminar . . . . .	30
5.2 Seminar . . . . .	31
<b>6 Stammvorlesungen</b>	<b>33</b>
6.1 Algorithms and Data Structures . . . . .	34
6.2 Artificial Intelligence . . . . .	35
6.3 Automated Reasoning . . . . .	37
6.4 Compiler Construction . . . . .	38
6.5 Complexity Theory . . . . .	39

6.6	Computer Algebra . . . . .	40
6.7	Computer Graphics . . . . .	41
6.8	Cryptography . . . . .	43
6.9	Data Networks . . . . .	44
6.10	Database Systems . . . . .	46
6.11	Digital Transmission & Signal Processing . . . . .	48
6.12	Distributed Systems . . . . .	50
6.13	Embedded Systems . . . . .	51
6.14	Geometric Modelling . . . . .	53
6.15	Human Computer Interaction . . . . .	55
6.16	Image Processing and Computer Vision . . . . .	56
6.17	Information Retrieval and Data Mining . . . . .	58
6.18	Introduction to Computational Logic . . . . .	59
6.19	Machine Learning . . . . .	60
6.20	Operating Systems . . . . .	61
6.21	Optimization . . . . .	63
6.22	Security . . . . .	64
6.23	Semantics . . . . .	65
6.24	Software Engineering . . . . .	66
6.25	Verification . . . . .	68
<b>7</b>	<b>Vertiefungsvorlesungen</b>	<b>69</b>
7.1	AI Planning . . . . .	70
7.2	Audio/Visual Communication and Networks . . . . .	71
7.3	Automata, Games and Verification . . . . .	73
7.4	Automated Debugging . . . . .	74
7.5	Correspondence Problems in Computer Vision . . . . .	75
7.6	Differential Equations in Image Processing and Computer Vision . . . . .	77
7.7	Introduction to Image Acquisition Methods . . . . .	79
7.8	Multimedia Transport . . . . .	80
7.9	Realistic Image Synthesis . . . . .	82
<b>8</b>	<b>Bachelor-Seminar und -Arbeit</b>	<b>84</b>
8.1	Bachelor-Seminar . . . . .	85
8.2	Bachelor-Arbeit . . . . .	86

**Modulbereich 1**

---

**Ringvorlesungen**

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>1</b>	<b>6</b>	<b>jedes Wintersemester</b>	<b>1 Semester</b>	<b>2</b>	<b>2</b>

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

**Dozent/inn/en** Professoren der Fachrichtung

**Zulassungsvoraussetzungen** keine

**Leistungskontrollen / Prüfungen** Positive Bewertung von mindestens drei schriftlichen Zusammenfassungen verschiedener Vorträge

**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

## Lernziele / Kompetenzen

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

## Inhalt

Querschnitt durch die Forschungsthemen der Saarbrücker Informatik. Die Themen spannen einen attraktiven Bogen von aktuellster Forschung zu anspruchsvollen Problemen der industriellen Praxis.

## Literaturhinweise

**Modulbereich 2**

---

***Grundlagen der Mathematik***

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>1</b>	<b>6</b>	<b>jedes Wintersemester</b>	<b>1 Semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Joachim Weickert

**Dozent/inn/en** Prof. Dr. Joachim Weickert  
 Prof. Dr. Frank-Olaf Schreyer  
 Prof. Dr. Mark Groves

**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

## 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



Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>2</b>	<b>6</b>	<b>jedes Sommersemester</b>	<b>1 Semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Joachim Weickert

**Dozent/inn/en** Prof. Dr. Joachim Weickert  
 Prof. Dr. Frank-Olaf Schreyer  
 Prof. Dr. Mark Groves

**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

## 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

### 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, Austauschatz
- 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

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>3</b>	<b>6</b>	<b>jedes Wintersemester</b>	<b>1 Semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Joachim Weickert

**Dozent/inn/en** Prof. Dr. Joachim Weickert  
Prof. Dr. Frank-Olaf Schreyer  
Prof. Dr. Mark Groves

**Zulassungsvoraussetzungen** Mathematik für Informatiker 1 und 2 (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

## 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.

STOCHASTIK, NUMERIK UND MEHRDIMENSIONALE ANALYSIS

E. NUMERISCHE ERGÄNZUNGEN (3)

- 52. Banachscher Fixpunktsatz (1)
- 53. Interpolation, incl. Splines (2)

F. MEHRDIMENSIONALE ANALYSIS UND NUMERIK (11)

- 54. Stetigkeit und Differentialoperatoren für skalarwertige Funktionen (2)
- 55. Differentialoperatoren für vektorwertige Funktionen (1)
- 56. Totale Differenzierbarkeit (1/2)
- 57. Mittelwertsatz und Satz von Taylor (1 1/2)
- 58. Extrema von Funktionen mehrerer Variabler (1)
- 59. Das Newton-Verfahren (1)
- 60. Extrema mit Nebenbedingungen (1)
- 61. Mehrfachintegrale (1)

62. Die Umkehrfunktion und die Transformationsregel (1)

63. Variationsrechnung (1)

G. STOCHASTIK (16)

64. Grundbegriffe (Ws., Stichprobenraum) (1/3)

65. Kombinatorik (2/3)

66. Erzeugende Funktionen (1)

67. Bedingte Wahrscheinlichkeiten (1)

68. Zufallsvariable, Erwartungswert, Varianz (2) (Systemzuverlässigkeit, Varianz, Kovarianz, Jensen)

69. Abschätzungen für Abweichungen vom Mittelwert (1) (Momente, Schranken von Markov, Chebyshev, Chernoff, schwaches Gesetz der grossen Zahlen)

70. Wichtige diskrete Verteilungen (1)

71. Wichtige kontinuierliche Verteilungen (1) (incl. zentraler Grenzwertsatz)

72. Multivariate Verteilungen und Summen von Zufallsvariablen (1)

73. Parameterschätzung und Konfidenzintervalle (1)

74. Hypothesentests (1)

75. Methode der kleinsten Quadrate (1)

76. Robuste Statistik (2/3)

77. Fehlerfortpflanzung (1/3)

78. Markowketten (2)

79. Pseudozufallszahlen und Monte-Carlo-Simulation (1)

## Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet



Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>4</b>	<b>6</b>	<b>jedes Sommersemester</b>	<b>1 Semester</b>	<b>4</b>	<b>6</b>

**Modulverantwortliche/r** Prof. Dr. Jens Dittrich

**Dozent/inn/en** Prof. Dr. Jens Dittrich

**Zulassungsvoraussetzungen** keine (siehe aber „Empfohlene Vorkenntnisse“ weiter unten)

**Leistungskontrollen / Prüfungen** Erfolgreiche Teilnahme an den Übungen/Projekt berechtigt zur Teilnahme an der Abschlussklausur (bzw. Studienarbeit).

**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 (alternativ Studienarbeit), Übungen, ggf. Projekt ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekanntgegeben.

**Sprache** Deutsch

## 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.

**Empfohlene Vorkenntnisse:** Programmierung 1, Programmierung 2, Softwarepraktikum oder Projektpraktikum, Mathematik für Informatiker 1, sowie Grundzüge von Algorithmen und Datenstrukturen.



Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>4</b>	<b>6</b>	<b>jedes Wintersemester</b>	<b>1 Semester</b>	<b>4</b>	<b>6</b>

**Modulverantwortliche/r** Prof. Dr. Tobias Marschall

**Dozent/inn/en** Prof. Dr. Tobias Marschall  
Prof. Dr. Bernt Schiele  
Prof. Dr. Jilles Vreeken

**Zulassungsvoraussetzungen** Die Vorlesung setzt grundlegende Kenntnisse in Statistik und linearer Algebra voraus. Ist deshalb ratsam, MfI2 und das Statistics Lab erfolgreich abgeschlossen zu haben. Die Übungen verwenden die Programmiersprache R und grundlegende Kenntnisse sind hilfreich. Zur Vorbereitung sind die folgenden Materialien nützlich: „R for Beginners“ von Emmanuel Paradis (insbesondere Kapitel 1, 2, 3 und 6) und „An introduction to R“ (Venables/Smith).

**Leistungskontrollen / Prüfungen** Voraussetzung zur Zulassung zur Prüfung sind 50% der Punkte der theoretischen und praktischen Aufgaben auf den Übungsblättern. Die Prüfungen finden, je nach Teilnehmerzahl, schriftlich oder mündlich statt. Die genauen Modalitäten werden in den ersten zwei Wochen der Vorlesung bekannt gegeben.

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

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

**Modulnote** Siehe "Leistungskontrollen/Prüfungen"

**Sprache** English

## Lernziele / Kompetenzen

In diesem Kurs werden grundlegende Konzepte des maschinellen Lernens behandelt, wobei der Schwerpunkt auf statistischen Methoden liegt. Der Kurs vermittelt die nötigen Fähigkeiten um für einen gegebenen Datensatz geeignete statistische Methoden für dessen Analyse auszuwählen, anzuwenden, und die Qualität der Resultate zu bewerten. Der Kurs behandelt sowohl theoretische als auch praktische Aspekte des maschinellen Lernens, legt den Fokus jedoch auf praktische Aspekte.

Die Vorlesung folgt im Großen und Ganzen dem Buch "An Introduction to Statistical Learning with Applications in R (2013)". In einigen Fällen erhält der Kurs zusätzliches Material aus dem Buch The Elements of Statistical Learning, Springer (second edition, 2009). Das erste Buch ist der einleitende Text, das zweite behandelt fortgeschrittenere Themen. Beide Bücher sind als kostenlose PDFs erhältlich. Es wird durchschnittlich eine Vorlesung pro Woche (90 Minuten) und alle zwei Wochen (90 Minuten) ein Tutorium angeboten.

## Inhalt

Die Vorlesung behandelt grundlegende Methoden des maschinellen Lernens, insbesondere folgende Inhalte:

- 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

## **Literaturhinweise**

## **Weitere Informationen**

Die Inhalte sind größtenteils identisch mit der Vorlesung, die in der Vergangenheit unter dem Namen *Elements of Statistical Learning* angeboten wurde.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>3</b>	<b>6</b>	<b>jedes Wintersemester</b>	<b>1 Semester</b>	<b>6</b>	<b>9</b>

**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** *Programmierung 1 und 2 und Mathematik für Informatiker 1 und 2* oder vergleichbare Veranstaltungen der Mathematik (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** Deutsch

## 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.  $\mu$ -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.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>3</b>	<b>6</b>	<b>jedes Wintersemester</b>	<b>1 Semester</b>	<b>4</b>	<b>6</b>

**Modulverantwortliche/r** Prof. Dr. Raimund Seidel

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

**Zulassungsvoraussetzungen** *Programmierung 1 und 2, und Mathematik für Informatiker 1 und 2 oder vergleichbare Veranstaltungen der Mathematik (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** Deutsch

## 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.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>4</b>	<b>6</b>	<b>jedes Sommersemester</b>	<b>1 Semester</b>	<b>4</b>	<b>6</b>

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

**Leistungskontrollen / Prüfungen** Zwei Klausuren (Mitte und Ende der Vorlesungszeit), praktisches Projekt.  
 Nachklausuren finden innerhalb der letzten Wochen vor Vorlesungsbeginn des Folgesemesters statt..

**Lehrveranstaltungen / SWS** **Element T – Theorie (2 SWS):**  
 8 Vorlesungen: 6 Wochen  
 4 Übungen: 6 Wochen  
**Element A – Anwendung (2 SWS):**  
 9 Vorlesungen: 6 Wochen  
 4 Übungen: 6 Wochen  
**Element P – Praxis (Eigenstudium):**  
 Semesterbegleitend 8 schriftliche Reflektionen (Prüfungsvorleistungen), anschließend Projektarbeit über ca. 2 Wochen  
**= 4 SWS**

**Arbeitsaufwand** **Element T:**  
 24 h Präsenz, 36h Selbststudium  
**Element A:**  
 26 h Präsenz, 34h Selbststudium  
**Element P:**  
 60 h Selbststudium  
 50 h Präsenzstudium  
 + 130 h Eigenstudium  
 = 180 h (= 6 ECTS)

**Modulnote** Wird aus Leistungen in Klausuren (im Anschluss an die Elemente T und A), sowie den Prüfungsvorleistungen (Element P) ermittelt. Die genauen Modalitäten werden vom Modulverantwortlichen bekannt gegeben. Alle Modulelemente sind innerhalb eines Prüfungszeitraumes erfolgreich zu absolvieren.

**Sprache** Deutsch

## Lernziele / Kompetenzen

Die Teilnehmer lernen die Nebenläufigkeit von Prozessen 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. Außerdem 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 verschiedene Phänomene des nebenläufigen Programmierens in den formalen Modellen zu beschreiben und mit deren Hilfe konkrete Lösungen für die Praxis abzuleiten. Des Weiteren werden die Teilnehmer in der Praxis existierende Konzepte auf diese Art auf ihre Verlässlichkeit hin untersuchen.

## 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 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
- 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
- Shared Objects und Threads in Java
- Shared Objects und Threads als Transitionssysteme

### Analyse und Programmierunterstützung

- 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.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>1</b>	<b>6</b>	<b>jedes Wintersemester</b>	<b>1 Semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Gert Smolka

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

**Zulassungsvoraussetzungen** keine

**Leistungskontrollen / Prüfungen**

- zwei Klausuren (Mitte und Ende der Vorlesungszeit)
- Die Note wird aus den Klausuren gemittelt und kann durch Leistungen in den Übungen verbessert werden.
- Eine Nachklausur findet innerhalb der letzten beiden Wochen vor Vorlesungsbeginn des Folgesemesters statt.

**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

- höherstufige, getypte funktionale Programmierung anwenden können
- Verständnis rekursiver Datenstrukturen und Algorithmen, Zusammenhänge mit Mengenlehre
- Korrektheit beweisen und Laufzeit abschätzen
- Typabstraktion und Modularisierung verstehen
- Struktur von Programmiersprachen verstehen
- einfache Programmiersprachen formal beschreiben können
- einfache Programmiersprachen implementieren können
- anwendungsnahe Rechenmodelle mit maschinennahen Rechenmodellen realisieren können
- Praktische Programmiererfahrung, Routine im Umgang mit Interpretern und Übersetzern

## Inhalt

- Funktionale Programmierung
- Algorithmen und Datenstrukturen (Listen, Bäume, Graphen; Korrektheitsbeweise; asymptotische Laufzeit)
- Typabstraktion und Module
- Programmieren mit Ausnahmen
- Datenstrukturen mit Zustand
- Struktur von Programmiersprachen (konkrete und abstrakte Syntax, statische und dynamische Syntax)
- Realisierung von Programmiersprachen (Interpreter, virtuelle Maschinen, Übersetzer)

## Literaturhinweise

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>2</b>	<b>6</b>	<b>jedes Sommersemester</b>	<b>1 Semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Sebastian Hack

**Dozent/inn/en** Prof. Dr. Sebastian Hack  
Prof. Dr. Jörg Hoffmann

**Zulassungsvoraussetzungen** *Programmierung 1* und *Mathematik für Informatiker 1* und Mathematikveranstaltungen im Studiensemester oder vergleichbare Kenntnisse aus sonstigen Mathematikveranstaltungen (empfohlen)

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

Im **Praktikumsteil** müssen die Studierenden eine Reihe von Programmieraufgaben selbstständig implementieren. Diese Programmieraufgaben ermöglichen das Einüben der Sprachkonzepte und führen außerdem komplexere Algorithmen und Datenstrukturen ein. Automatische Tests prüfen die Qualität der Implementierungen. Die Note des Praktikumsteils wird maßgeblich durch die Testergebnisse bestimmt.

Im **Vorlesungsteil** müssen die Studierenden Klausuren absolvieren und Übungsaufgaben bearbeiten. Die Aufgaben vertiefen dabei den Stoff der Vorlesung. Die Zulassung zu der Klausur hängt von der erfolgreichen Bearbeitung der Übungsaufgaben ab.

Im Praktikumsteil kann eine Nachaufgabe angeboten werden

**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 lernen die Grundprinzipien der imperativen /objektorientierten Programmierung kennen. Dabei wird primär Java als Programmiersprache verwendet.

In dieser Vorlesung lernen sie:

- wie Rechner Programme ausführen
- Die Grundlagen imperativer und objektorientierter Sprachen
- kleinere, wohlstrukturierte Programme in C zu schreiben
- mittelgroße objektorientierte Systeme in Java zu implementieren und zu testen
- sich in wenigen Tagen eine neue imperative/objektorientierte Sprache anzueignen, um sich in ein bestehendes Projekt einzuarbeiten Inhalt



## **Inhalt**

- Imperatives Programmieren
- Objekte und Klassen
- Klassendefinitionen
- Objektinteraktion
- Objektsammlungen
- Objekte nutzen und testen
- Vererbung
- Dynamische Bindung
- Fehlerbehandlung
- Klassendesign und Modularität
- Systemnahe Programmierung

sowie spezifische Vorlesungen für die Programmieraufgaben.

## **Literaturhinweise**

Bekanntgabe jeweils vor Beginn der Vorlesung auf der Vorlesungsseite im Internet

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>2</b>	<b>6</b>	<b>jedes Sommersemester</b>	<b>1 Semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Jan Reineke

**Dozent/inn/en** Prof. Dr. Jan Reineke

**Zulassungsvoraussetzungen** *Programmierung 1* und *Mathematik für Informatiker 1* oder vergleichbare Veranstaltungen der Mathematik (empfohlen)

**Leistungskontrollen / Prüfungen** **Studienleistungen:** die Vorlesungen hören, nach bearbeiten und gegebenenfalls verstehen; die Übungen allein oder in Gruppen bearbeiten; erfolgreich bearbeitete Übungen in der Übungsgruppe vortragen.

**Prüfungsleistungen:** erfolgreiche Bearbeitung von 50% der Übungsaufgaben berechtigt zur Teilnahme an den Klausuren. Bestehen von zwei aus drei Klausuren.

**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 sollen die Funktionsweise, die Eigenschaften und die Entwurfsprinzipien von Rechnerarchitekturen und Betriebssystemen kennen lernen.

## Inhalt

1. Hardware
  - a. Boole'sche Algebra und Schaltkreise
  - b. Elementare Rechnerarithmetik
  - c. ALU (Konstruktion und Korrektheit)
  - d. Sequentieller vereinfachter DLX-Prozessor (Konstruktion und Korrektheit)
2. Betriebssystemkern
  - a. Virtualisierung
  - b. Ressourcen-Verwaltung, Speicher, Prozessor
  - c. Scheduling
  - d. Datei-System

## Literaturhinweise

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

## **Modulbereich 4**

---

### ***Praktika***

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>2-3</b>	<b>6</b>	<b>Vorlesungsfr. Zeit nach dem SS</b>	<b>7 Wochen</b>	<b>BLOCK</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Sven Apel

**Dozent/inn/en** Prof. Dr. Sven Apel

**Zulassungsvoraussetzungen** Die Teilnahme am Softwarepraktikum setzt umfangreiche Programmierkenntnisse voraus, wie sie in den Vorlesungen *Programmierung 1* und *Programmierung 2* vermittelt werden.

Für die Teilnahme am Softwarepraktikum werden eigene Laptops benötigt, die selbst mitgebracht werden müssen.

**Leistungskontrollen / Prüfungen** Das Ziel des Softwarepraktikums ist es, in einer Gruppe von Studierenden ein nicht-triviales Softwaresystem zu erstellen. Dazu müssen eine Reihe von Dokumenten (Entwurf, Quellcode, Tests, etc.) erstellt und abgegeben werden. Bewertet wird die Korrektheit und Qualität der Dokumente sowie die fristgerechte Abgabe.

Das Softwarepraktikum gliedert sich in eine Übungsphase, eine Gruppenphase und eine Einzelphase. In der Übungsphase werden täglich Minitests zu den aktuellen Vorlesungsinhalten durchgeführt und bewertet.

In der Gruppenphase wird ein substantielles Softwaresystem im Team geplant, entworfen, implementiert und getestet. Um zur Gruppenphase zugelassen zu werden, müssen die Studierenden die Minitests der Übungsphase bestehen.

In der Einzelphase wird ein kleineres Softwaresystem oder eine Erweiterung eines bestehenden Systems (z.B. aus der Gruppenphase) von den Studierenden jeweils allein entwickelt. Voraussetzung für die Einzelphase ist die erfolgreiche Absolvierung der Gruppenphase.

Die Softwaresysteme der Gruppen- und Einzelphase, sowie die zugehörigen Dokumente (Entwurf, Quellcode, Tests, etc.), werden auf Basis der Prinzipien und Qualitätsstandards der Vorlesung bewertet. Genauere Prüfungsmodalitäten werden zu Beginn des Softwarepraktikums in der Vorlesung bekannt gegeben.

**Lehrveranstaltungen / SWS** täglich Projektarbeit mit Betreuung  
teilweise Vorlesung

**Arbeitsaufwand** 25 h Vorlesung  
+ 245 h Projektarbeit  
= 270 h (= 9 ECTS)

**Modulnote** unbenotet

**Sprache** Deutsch

## Lernziele / Kompetenzen

Die Studierenden erwerben die Fähigkeit, im Team zu arbeiten und Probleme des Software Engineerings zu lösen.

Die Studierenden wissen, welche Probleme beim Durchführen eines Softwareprojekts auftreten können, und wie diese gelöst werden können.

Sie können eine komplexe Aufgabenstellung eigenständig in ein Softwareprodukt umsetzen, das den Anforderungen des Kunden entspricht. Hierfür wählen sie einen passenden Entwicklungsprozess, der Risiken früh erkannt und minimiert, und wenden diesen an.

Sie sind vertraut mit Grundzügen des Softwareentwurfs wie schwache Kopplung, hohe Kohäsion, Geheimnisprinzip sowie Entwurfs- und Architekturmustern und sind in der Lage, einen Entwurf anhand dieser Kriterien zu erstellen, zu beurteilen und zu verbessern.

Sie beherrschen Techniken der Qualitätssicherung wie Testen und Debugging und wenden diese an.

## **Inhalt**

- Softwareentwurf
- Softwaretesten
- Teamarbeit
- 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.
- Pattern-Oriented Software Architecture: A System of Patterns. F. Buschmann, et al., Wiley, 1996.
- 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.

*Modulbereich 5*

---

*Seminare*

# Proseminar

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>4</b>	<b>6</b>	<b>jedes Semester</b>	<b>1 Semester</b>	<b>2</b>	<b>5</b>

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

**Dozent/inn/en** Professoren der Fachrichtung

**Zulassungsvoraussetzungen** Grundlegende Kenntnisse im jeweiligen Teilbereich des Studienganges.

**Leistungskontrollen / Prüfungen**

- Diskussion in der Gruppe
- thematischer Vortrag
- kurze schriftliche Ausarbeitung

**Lehrveranstaltungen / SWS** 2 SWS Proseminar

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

**Modulnote** Die Modalitäten der Notenvergabe werden vom jeweiligen verantwortlichen Hochschullehrer festgelegt.

**Sprache** Deutsch

## Lernziele / Kompetenzen

Die Studierenden haben am Ende der Veranstaltung ein profundes Verständnis aktueller oder fundamentaler Aspekte eines spezifischen Teilbereiches des Studienganges erlangt.

Sie haben Kompetenz im Verstehen einfacher wissenschaftlicher Aufsätze und im Präsentieren von wissenschaftlichen Erkenntnissen erworben.

## Inhalt

Unter Anleitung werden folgende Punkte praktisch geübt:

- Lesen und Verstehen wissenschaftlicher Aufsätze
- Diskutieren der Aufsätze in der Gruppe
- Analysieren, Zusammenfassen und Wiedergeben des spezifischen Themas
- Präsentationstechnik
- Spezifische Vertiefung in Bezug auf das individuelle Thema des Seminars.

## Literaturhinweise

dem Thema entsprechend

## Weitere Informationen

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

# Seminar

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>5</b>	<b>6</b>	<b>jedes Semester</b>	<b>1 Semester</b>	<b>3</b>	<b>7</b>

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

**Dozent/inn/en** Professoren der Fachrichtung

**Zulassungsvoraussetzungen** Grundlegende Kenntnisse im jeweiligen Teilbereich der Studienganges.

**Leistungskontrollen / Prüfungen**

- Beiträge zur Diskussion
- Thematischer Vortrag
- Schriftliche Ausarbeitung
- Mündliche Abschlussprüfung über das gesamte Themengebiet

**Lehrveranstaltungen / SWS** 2 SWS Seminar

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

**Modulnote** Die Modalitäten der Notenvergabe werden vom verantwortlichen Hochschullehrer festgelegt.

**Sprache** Deutsch oder Englisch

## Lernziele / Kompetenzen

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

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

## Inhalt

Praktisches Einüben von

- reflektierender wissenschaftlicher Arbeit,
- Analyse und Bewertung wissenschaftlicher Aufsätze,
- Verfassen eigener wissenschaftlicher Zusammenfassungen
- Diskussion der Arbeiten in der Gruppe
- Erarbeiten gemeinsamer Standards für wissenschaftliches Arbeit
- Präsentationstechnik

Spezifische Vertiefung in Bezug auf das individuelle Thema des Seminars.

Der typische Ablauf eines Seminars ist wie folgt:

- Vorbereitende Gespräche zur Themenauswahl
- Regelmäßige Treffen mit Diskussion ausgewählter Beiträge
- Vortrag und Ausarbeitung zu einem der Beiträge
- Mündliche Prüfung über das erarbeitete Themengebiet



## **Literaturhinweise**

dem Thema entsprechend

## **Weitere Informationen**

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

**Modulbereich 6**

---

**Stammvorlesungen**

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Jörg Hoffmann

**Dozent/inn/en** Prof. Dr. Jörg Hoffmann  
Prof. Dr. Jana Köhler

**Zulassungsvoraussetzungen** For graduate students: none

**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

Problem-solving:

- Uninformed- and informed search procedures
- Adversarial search

Knowledge and reasoning:

- Propositional logic
- SAT
- First-order logic, Inference in first-order logic
- Knowledge representation, Semantic Web
- Default logic, rule-based mechanisms

Planning:

- STRIPS formalism and complexity
- Delete relaxation heuristics

Probabilistic reasoning:

- Basic probabilistic methods
- Bayesian networks

## **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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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 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

## Literaturhinweise

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

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Michael Backes

**Dozent/inn/en** Prof. Dr. Markus Bläser  
Dr. Nico Döttling

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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**

- Regular attendance of classes and tutorials
- Qualification for final exam through mini quizzes during classes
- Possibility to get bonus points through excellent homework
- 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

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
- Protokols 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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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*), *Programmierung 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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Bernd Finkbeiner, Ph.D

**Dozent/inn/en** Prof. Bernd Finkbeiner, Ph.D  
Prof. Dr. Martina Maggio

**Zulassungsvoraussetzungen** keine

**Leistungskontrollen / Prüfungen**

- Written exam at the end of the course.
- Demonstration of the implemented system.
- 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)

The course is accompanied by a laboratory project, in which a non-trivial embedded system has to be realized.

**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 should learn methods for the design, the implementation, and the validation of safety-critical embedded systems.

## Inhalt

Embedded Computer Systems are components of a technical system, e.g. an air plane, a car, a household machine, a production facility. They control some part of this system, often called the plant, e.g. the airbag controller in a car controls one or several airbags. Controlling means obtaining sensor values and computing values of actuator signals and sending them.

Most software taught in programming courses is transformational, i.e. it is started on some input, computes the corresponding output and terminates. Embedded software is reactive, i.e. it is continuously active waiting for signals from the plant and issuing signals to the plant.

Many embedded systems control safety-critical systems, i.e. malfunctioning of the system will in general cause severe damage. In addition, many have to satisfy real-time requirements, i.e. their reactions to input have to be produced within fixed deadlines.

According to recent statistics, more than 99% of all processors are embedded. Processors in the ubiquitous PC are a negligible minority. Embedded systems have a great economical impact as most innovations in domains like avionics, automotive are connected to advances in computer control. On the other hand, failures in the design of such systems may have disastrous consequences for the functioning of the overall system. Therefore, formal specification techniques and automatic synthesis of software are used more than in other domains.

The course will cover most aspects of the design and implementation of embedded systems, e.g. specification mechanisms, embedded hardware, operating systems, scheduling, validation 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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Jürgen Steimle

**Dozent/inn/en** Prof. Dr. Jürgen Steimle

**Zulassungsvoraussetzungen** undergraduate students: *Programmierung 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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Joachim Weickert

**Dozent/inn/en** Prof. Dr. Joachim Weickert

**Zulassungsvoraussetzungen** For graduate students: none

**Leistungskontrollen / Prüfungen**

- Regular attendance of classes and tutorials.
- At least 50% of all possible points from the weekly assignments have to be gained to qualify for the final exam.
- 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

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. Image Transformations
  - 2.1 Fourier Transform
  - 2.2 Image Pyramids
  - 2.3 Wavelet Transform
3. Colour Perception and Colour Spaces
4. Image Enhancement
  - 4.1 Point Operations
  - 4.2 Linear Filtering
  - 4.3 Wavelet Shrinkage, Median Filtering, M-Smothers
  - 4.4 Mathematical Morphology
  - 4.5 Diffusion Filtering
  - 4.6 Variational Methods

- 4.7 Deblurring
- 5. Feature Extraction
  - 5.1 Edges
  - 5.2 Corners
  - 5.3 Lines and Circles
- 6. Texture Analysis
- 7. Segmentation
  - 7.1 Classical Methods
  - 7.2 Variational Methods
- 8. Image Sequence Analysis
  - 8.1 Local Methods
  - 8.2 Variational Methods
- 9. 3-D Reconstruction
  - 9.1 Camera Geometry
  - 9.2 Stereo
  - 9.3 Shape-from-Shading
- 10. Object Recognition
  - 10.1 Eigenspace Methods
  - 10.2 Moment Invariances

## **Literaturhinweise**

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

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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

# Semantics

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Sven Apel

**Dozent/inn/en** Prof. Dr. Sven Apel

**Zulassungsvoraussetzungen**

- Knowledge of programming concepts (as taught in the lectures *Programmierung 1* and *Programmierung 2*)
- Basic knowledge of software processes, design, and testing (as taught and applied in the lecture *Softwarepraktikum*)

**Leistungskontrollen / Prüfungen** Beside the lecture and weekly practical exercises, there will be a number of assignments in the form of mini-projects for each student to work on (every two to three weeks). The assignments will be assessed based on the principles covered in the lecture. Passing all assignments is a prerequisite for taking the final written exam. The final grade is determined only by the written exam. Further examination details will be announced by the lecturer at the beginning of the course. In short:

- Passing all assignments (prerequisite for the written exam)
- Passing the written exam

**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. Passing all assignments is a prerequisite for taking the written exam. The assignments do not contribute to the final grade. 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, mixins/traits, aspects, pointcuts, advice.
- 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
- Aspects, pointcuts, advice
- 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**

- Feature-Oriented Software Product Lines: Concepts and Implementation. S. Apel, et al., Springer, 2013.
- Generative Programming: Methods, Tools, and Applications: Methods, Techniques and Applications. K. Czarnecki, et al., Addison-Wesley, 2000.
- Mastering Software Variability with FeatureIDE. J. Meinicke, et al., Springer, 2017.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>5</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**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.

***Modulbereich 7***

---

***Vertiefungsvorlesungen***

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

**Modulverantwortliche/r** Prof. Dr. Jörg Hoffmann

**Dozent/inn/en** Prof. Dr. Jörg Hoffmann

**Zulassungsvoraussetzungen** For graduate students: none

**Leistungskontrollen / Prüfungen** Regular attendance of classes and tutorial  
Paper as well as programming exercises for exam qualification  
Final exam  
A re-exam takes place 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 will gain a deep understanding of algorithms used in Automatic Planning for the efficient exploration of large state spaces, from both a theoretical and practical point of view. The programming exercises will familiarize them with the main implementation basis in Automatic Planning. The search algorithms are generic and are relevant also in other CS sub-areas in which large transition systems need to be analyzed.

## Inhalt

Automatic Planning is one of the fundamental sub-areas of Artificial Intelligence, concerned with algorithms that can generate strategies of action for arbitrary autonomous agents in arbitrary environments. The course examines the technical core of the current research on solving this kind of problem, consisting of paradigms for automatically generating heuristic functions (lower bound solution cost estimators), as well as optimality-preserving pruning methods. Apart from understanding these techniques themselves, the course explains how to analyze, combine, and compare them.

Starting from an implementation basis provided, students implement their own planning system as part of the course. The course is concluded by a competition between these student systems.

## Literaturhinweise

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

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

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

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

**Zulassungsvoraussetzungen** Solid foundation of mathematics (differential and integral calculus) and probability theory. The course will build on the mathematical concepts and tools taught in TC I while trying to enable everyone to follow and to fill gaps by an accelerated study of the accompanying literature. *Signals and Systems* as well as *Digital Transmission and Signal Processing (TC I)* are strongly recommended but not required. Related core lecture *Digital Transmission and Signal Processing (TC I)*

**Leistungskontrollen / Prüfungen** Regular attendance of classes and tutorials Passing the final exam  
Oral exam directly succeeding the course. Eligibility: Weekly excersises / 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

AVCN will deepen the students' knowledge on modern communications systems and will focus on wireless systems.

Since from a telecommunications perspective the combination of audio/visual data – meaning inherently high data rate and putting high requirements on the realtime capabilities of the underlying network – and wireless transmission – that is unreliable and highly dynamic with respect to the channel characteristics and its capacity – is the most demanding application domain.

## Inhalt

As the basic principle the course will study and introduce the building blocks of wireless communication systems. Multiple access schemes like TDMA, FDMA, CDMA and SDMA are introduced, antennas and propagation incl. link budget calculations are dealt with and more advanced channel models like MIMO are investigated. Modulation and error correction technologies presented in Telecommunications I will be expanded by e.g. turbo coding and receiver architectures like RAKE and BLAST will be introduced. A noticeable portion of the lecture will present existing and future wireless networks and their extensions for audio/visual data. Examples include 802.11n and the terrestrial DVB system (DVB-T2).

## 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 II*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>4</b>	<b>6</b>

**Modulverantwortliche/r** Prof. Bernd Finkbeiner, Ph.D

**Dozent/inn/en** Prof. Bernd Finkbeiner, Ph.D

**Zulassungsvoraussetzungen** keine

**Leistungskontrollen / Prüfungen**

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

**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, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

**Sprache** English

## Lernziele / Kompetenzen

The students will gain a deep understanding of the automata-theoretic background of automated verification and program synthesis.

## Inhalt

The theory of automata over infinite objects provides a succinct, expressive and formal framework for reasoning about reactive systems, such as communication protocols and control systems. Reactive systems are characterized by their nonterminating behaviour and persistent interaction with their environment.

In this course we study the main ingredients of this elegant theory, and its application to automatic verification (model checking) and program synthesis.

- Automata over infinite words and trees (omega-automata)
- Infinite two-person games
- Logical systems for the specification of nonterminating behavior
- Transformation of automata according to logical operations

## Literaturhinweise

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

# Automated Debugging

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>4</b>	<b>6</b>

**Modulverantwortliche/r** Prof. Dr. Andreas Zeller

**Dozent/inn/en** Prof. Dr. Andreas Zeller

**Zulassungsvoraussetzungen** *Programmierung 1, Programmierung 2 and Softwarepraktikum*

**Leistungskontrollen / Prüfungen** Projects and mini-tests

**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** The module is passed in its entirety if the examination performance has been passed.

**Sprache** English

## Lernziele / Kompetenzen

Finding and fixing software bugs can involve lots of effort. This course addresses this problem by automating software debugging, specifically identifying failure causes, locating bugs, and fixing them. Students learn the basics of systematic debugging, and explore tools and techniques for automated debugging.

## Inhalt

- Tracking Problems
- The Scientific Method
- Cause-Effect Chains
- Building a Debugger
- Tracking Inputs
- Assertions and Sanitizers
- Detecting Anomalies
- Statistical Fault Localization
- Generating Tests
- Reducing Failure-Inducing Inputs
- Mining Software Archives
- Fixing the Defect
- Repairing Bugs Automatically
- Managing Bugs

## Literaturhinweise

The teaching material consists of text, Python code, and Jupyter Notebooks from the textbook “The Debugging Book” (<https://www.debuggingbook.org/>), also in English.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>4</b>	<b>6</b>

**Modulverantwortliche/r** Prof. Dr. Joachim Weickert

**Dozent/inn/en** Prof. Dr. Joachim Weickert

**Zulassungsvoraussetzungen** Related core lecture *Computer Vision*,  
*Mathematik für Informatiker 1 - 3*

**Leistungskontrollen / Prüfungen**

- Regular attendance of lecture and tutorial
- Written or oral exam and the end of the course

**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, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

**Sprache** English

## Lernziele / Kompetenzen

Correspondence problems are a central topic in computer vision. Thereby, one is interested in identifying and matching corresponding features in different images/views of the same scene. Typical correspondence problems are the estimation of motion information from consecutive frames of an image sequence (optic flow), the reconstruction of a 3-D scene from a stereo image pair and the registration of medical image data from different modalities (e.g. CT and MRT). Central part of this lecture is the discussion of the most important correspondence problems as well as the modelling of suitable algorithms for solving them.

## Inhalt

1. Introduction and Overview
2. General Matching Concepts
  - 2.1 Block Matching
  - 2.2 Correlation Techniques
  - 2.3 Interest Points
  - 2.4 Feature-Based Methods
3. Optic Flow I
  - 3.1 Local Differential Methods
  - 3.2 Parameterisation Models
4. Optic Flow II
  - 4.1 Global Differential Methods
  - 4.2 Horn and Schunck
5. Optic Flow III
  - 5.1 Advanced Constancy Assumptions
  - 5.2 Large Motion
6. Optic Flow IV
  - 6.1 Robust Data Terms

- 6.2 Discontinuity-Preserving Smoothness Terms
- 7. Optic Flow V
  - 7.1 High Accuracy Methods
  - 7.2 SOR and Liemar Multigrid
- 8. Stereo Matching I
  - 8.1 Projective Geometry
  - 8.2 Epipolar Geometry
- 9. Stereo Matching II
  - 9.1 Estimation of the Fundamental Matrix
- 10. Stereo Matching III
  - 10.1 Correlation Methods
  - 10.2 Variational Approaches
  - 10.3 Graph Cuts
- 11. Medical Image Registration
  - 11.1 Mutual Information
  - 11.2 Elastic and Curvature Based Registration
  - 11.3 Landmarks
- 12. Particle Image Velocimetry
  - 12.1 Div-Curl-Regularisation
  - 12.2 Incompressible Navier Stokes Prior

## **Literaturhinweise**

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

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Joachim Weickert

**Dozent/inn/en** Prof. Dr. Joachim Weickert

**Zulassungsvoraussetzungen** Related core lecture *Computer Vision*

**Leistungskontrollen / Prüfungen**

- Regular attendance of lecture and tutorial
- 50% of all possible points from weekly assignments to be eligible for the final exam are needed
- Passing the final exam or the re-exam
- The 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)

50% theoretical exercises and 50% practical programming assignments

**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

Many modern techniques in image processing and computer vision make use of methods based on partial differential equations (PDEs) and variational calculus. Moreover, many classical methods may be reinterpreted as approximations of PDE-based techniques. In this course the students will get an in-depth insight into these methods. For each of these techniques, they will learn the basic ideas as well as theoretical and algorithmic aspects. Examples from the fields of medical imaging and computer aided quality control will illustrate the various application possibilities.

## Inhalt

1. Introduction and Overview
2. Linear Diffusion Filtering
  - 2.1 Basic Concepts
  - 2.2 Numerics
  - 2.3 Limitations and Alternatives
3. Nonlinear Isotropic Diffusion Filtering
  - 3.1 Modeling
  - 3.2 Continuous Theory
  - 3.2 Semidiscrete Theory
  - 3.3 Discrete Theory
  - 3.4 Efficient Sequential and Parallel Algorithms
4. Nonlinear Anisotropic Diffusion Filtering
  - 4.1 Modeling
  - 4.2 Continuous Theory

- 4.3 Discrete Aspects
- 5. Parameter Selection
- 6. Variational Methods
  - 6.1 Basic Ideas
  - 6.2 Discrete Aspects
  - 6.3 TV Denoising, Equivalence Results
  - 6.4 Mumford-Shah Segmentation and Diffusion-Reaction Filters
- 7. Vector- and Matrix-Valued Images
- 8. Image Sequence Analysis
  - 8.1 Global Methods
  - 8.2 Local Methods
  - 8.3 Combined Local-Global Methods
  - 8.4 Numerical Techniques
- 9. Continuous-Scale Morphology
  - 9.1 Basic Ideas
  - 9.2 Applications
- 10. Curvature-Based Morphology
  - 10.1 Basic Ideas
  - 10.2 Applications

## **Literaturhinweise**

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

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>2</b>	<b>4</b>

**Modulverantwortliche/r** Prof. Dr. Joachim Weickert

**Dozent/inn/en** N.N.

**Zulassungsvoraussetzungen** Related core lecture *Computer Vision*

**Leistungskontrollen / Prüfungen**

- Written or oral exam at end of course
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

**Lehrveranstaltungen / SWS** 2 h lectures (weekly)

**Arbeitsaufwand** 30 h of classes  
+ 90 h private study  
= 120 h (= 4 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 course is designed as a supplement for image processing lectures, to be attended before, after or parallel to them.

Participants shall understand

- what are digital images
- how they are acquired
- what they encode and what they mean
- which limitations are introduced by the image acquisition.

This knowledge will be helpful in selecting adequate methods for processing image data arising from different methods.

## Inhalt

A broad variety of image acquisition methods is described, including imaging by virtually all sorts of electromagnetic waves, acoustic imaging, magnetic resonance imaging and more. While medical imaging methods play an important role, the overview is not limited to them.

Starting from physical foundations, description of each image acquisition method extends via aspects of technical realisation to mathematical modelling and representation of the data.

## Literaturhinweise

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



Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>winter semester</b>	<b>1 semester</b>	<b>4</b>	<b>9</b>

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

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

**Zulassungsvoraussetzungen** For graduate students: none

**Leistungskontrollen / Prüfungen** Regular attendance of classes and tutorial.  
Paper as well as programming exercises for exam qualification  
Final exam  
A re-exam takes place before the start of lectures in the following semester

**Lehrveranstaltungen / SWS** 3 h lectures  
+ 1 h tutorial  
= 4 h (weekly)

**Arbeitsaufwand** 60 h of classes  
+ 90 h private study  
+ 120 h programming exercise  
= 270 h (= 9 ECTS)

**Modulnote** Graded absolute 1.0-n.b. and relative A-F

**Sprache** English

## Lernziele / Kompetenzen

The course deals with Media Transport over the Internet. After the course students know how data- and mediatransport is solved in today's Internet and have a good understanding of so called erasure channels.

Besides the pure transport protocol design the course complements the fundamentals laid in TCI and TCII by introducing state-of-the-art error codes (Van-der-Monde-Codes, Fountain Codes) and by engineering tasks like the design of a Digital PLL.

## Inhalt

The course introduces media transmission over packet channels, specifically the Internet. After establishing a Quality of Service framework built on ITU requirements the course models erasure channels without and with memory. Key characteristics like the channel capacity and the minimum redundancy information are derived.

The second part of the course introduces current media transport protocol suites (TCP, UDP, RTP, RTSP) and middleware (ISMA, DLNA, UPnP, DVB-IPI).

In the second half of the course audiovisual coders used in the Internet are introduced (H.264, AAC), state-of-the-art forward error coding schemes (Van-der-Monde-Codes, Fountain Codes) are explained and essential elements like a Digital Phase-locked Loop are developed.

## Literaturhinweise

The course will come with a self contained manuscript. The most essential monographs used for and referenced within the manuscript are available in the Computer Science Library of Saarland University.

## **Weitere Informationen**

This module was formerly also known as *Future Media Internet*.

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>at least every two years</b>	<b>1 semester</b>	<b>6</b>	<b>9</b>

**Modulverantwortliche/r** Prof. Dr. Philipp Slusallek

**Dozent/inn/en** Prof. Dr. Philipp Slusallek  
Dr. Karol Myszkowski  
Guprit Singh

**Zulassungsvoraussetzungen** Related core lecture: *Computer Graphics*.

**Leistungskontrollen / Prüfungen**

- Theoretical and practical exercises (50% of the final grade)
- Final oral exam (other 50%)
- A minimum of 50% of needs to be achieved in each part to pass.
- 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** The final grade is be based on the assessments above. Any changes will be announced at the beginning of the semester.

**Sprache** English

## Lernziele / Kompetenzen

At the core of computer graphics is the requirement to render highly realistic and often even physically-accurate images of virtual 3D scenes. In this lecture students will learn about physically-based lighting simulation techniques to compute the distribution of light even in complex environment. The course also covers issues of perception of images, including also HDR technology, display technology, and related topics.

After this course students should be able to build their own highly realistic but also efficient rendering system.

## Inhalt

- Rendering Equation
- Radiosity and Finite-Element Techniques
- Probability Theory
- Monte-Carlo Integration & Importance Sampling
- Variance Reduction & Advanced Sampling Techniques
- BRDFs and Inversion Methods
- Path Tracing & \* Bidirectional Path Tracing
- Virtual Point-Light Techniques
- Density Estimation & Photon Mapping
- Vertex Connection & Merging
- Path Guiding
- Spatio-Temporal Sampling & Reconstruction
- Approaches for Interactive Global Illumination
- Machine Learning Techniques in Rendering

- Human Perception
- HDR & Tone-Mapping
- Modern Display Technology
- Perception-Based Rendering

## Literaturhinweise

Literature will be announced in the first lecture of the semester.

But here are some relevant text books:

- Pharr, Jakob, Humphreys, Physically Based Rendering : From Theory to Implementation, Morgan Kaufmann
- Shirley et al., Realistic Ray Tracing, 2. Ed., AK. Peters, 2003
- Jensen, Realistic Image Synthesis Using Photon Mapping, AK. Peters, 2001
- Dutre, et al., Advanced Global Illumination, AK. Peters, 2003
- Cohen, Wallace, Radiosity and Realistic Image Synthesis, Academic Press, 1993
- Apodaca, Gritz, Advanced Renderman: Creating CGI for the Motion Pictures, Morgan Kaufmann, 1999
- Ebert, Musgrave, et al., Texturing and Modeling, 3. Ed., Morgan Kaufmann, 2003
- Reinhard, Ward, Pattanaik, Debevec, Heidrich, Myszkowski, High Dynamic Range Imaging, Morgan Kaufmann Publishers, 2nd edition, 2010.
- Myszkowski, Mantiuk, Krawczyk. High Dynamic Range Video. Synthesis Digital Library of Engineering and Computer Science. Morgan & Claypool Publishers, San Rafael, USA, 2008.
- Glassner, Principles of Digital Image Synthesis, 2 volumes, Morgan Kaufman, 1995

**Modulbereich 8**

---

***Bachelor-Seminar und -Arbeit***

# Bachelor-Seminar

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>jedes Semester</b>	<b>variabel</b>	<b>5</b>	<b>9</b>

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

**Dozent/inn/en** Professoren der Fachrichtung

**Zulassungsvoraussetzungen** Erwerb von mindestens 120 CP

**Leistungskontrollen / Prüfungen**

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

**Lehrveranstaltungen / SWS** 2 SWS Seminar

**Arbeitsaufwand** 30 h Präsenzstudium (Seminar)  
+ 20 h Betreuung durch den Lehrstuhl  
+ 220 h Eigenstudium  
= 270 h (= 9 ECTS)

**Modulnote** benotet

**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

Auf der Grundlage des "state-of-the-art" werden die Methoden der Informatik systematisch unter Anleitung angewendet.

## Literaturhinweise

Dem Themengebiet entsprechende wissenschaftliche Artikel in enger Absprache mit dem Dozenten

# Bachelor-Arbeit

Studiensem.	Regelst.sem.	Turnus	Dauer	SWS	ECTS
<b>6</b>	<b>6</b>	<b>jedes Semester</b>	<b>3 Monate</b>	<b>-</b>	<b>12</b>

**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** 20 h Betreuung durch den Lehrstuhl  
+ 340 h Eigenstudium  
= 360 h (= 12 ECTS)

**Modulnote** Beurteilung der Bachelorarbeit

**Sprache** Deutsch oder Englisch

## Lernziele / Kompetenzen

Die Bachelor-Arbeit ist eine Projektarbeit, die unter Anleitung ausgeführt wird. Sie zeigt, dass der Kandidat/die Kandidatin in der Lage ist, innerhalb einer vorgegebenen Frist ein Problem aus dem Gebiet der Informatik unter Anleitung zu lösen und die Ergebnisse zu dokumentieren.

## Inhalt

Auf der Grundlage des "state-of-the-art" wird die systematische Anwendung der Methoden der Informatik dokumentiert.

## Literaturhinweise

Je nach Thema in Absprache mit dem betreuenden Hochschullehrer