UNIVERSITÄT BAYREUTH

Modulhandbuch

für Informatik Bachelor (LA Gym)
in der Fassung vom 15. Februar 2018 ÄS 20. Mai 2026

Fakultät für Mathematik, Physik und Informatik
Universität Bayreuth

Allgemeine Informationen und Lesehinweise

Ein zentraler Baustein des Bologna-Prozesses ist die Modularisierung der Studiengänge, das heißt die Umstellung des vormaligen Lehrveranstaltungssystems auf ein Modulsystem, in dem die Lehrveranstaltungen zu thematisch zusammenhängenden Veranstaltungsblöcken – also Modulen – gebündelt sind.

Dieses Modulhandbuch enthält die Beschreibungen aller Module, die im Studiengang angeboten werden. Das Modulhandbuch dient der Transparenz und versorgt Studierende, Studieninteressierte und andere interne und externe Adressatinnen und Adressaten mit Informationen über die Inhalte der einzelnen Module, ihre Qualifikationsziele sowie qualitative und quantitative Anforderungen.

Inhaltsverzeichnis und Index

Das Inhaltsverzeichnis gibt einen Überblick über die Modulbereiche und Module des Studiengangs. Die Klammerangabe nach dem Titel eines Moduls enthält den Zeitpunkt, zu dem dessen Beschreibung zuletzt aktualisiert wurde. Beispiel zur Notation: 24W steht für das Wintersemester 2024/25, 25S für das darauffolgende Sommersemester 2025.

Der Index am Ende des Modulhandbuchs listet alle Module des Studiengangs in alphabetischer Reihenfolge auf.

Modulbeschreibung

Die Beschreibung eines Moduls enthält dessen Lernziele, Lerninhalte und Leistungsnachweise. Sieht ein Modul mehrere Leistungsnachweise vor, ist deren Anteil an der Modulgesamtnote als Gewichtung angegeben. Eine detaillierte Beschreibung der Prüfungsformen, durch die die Leistungsnachweise erbracht werden, findet sich in § 9 der Allgemeinen Prüfungs- und Studienordnung für die Bachelor- und Masterstudiengänge an der Universität Bayreuth.

Durch Scannen oder Anklicken des QR-Code in der Beschreibung gelangt man zur Webseite des Moduls. Hierüber lassen sich die zum Modul gehörigen Lehrveranstaltungen einsehen und es wird angegeben, in welchen weiteren Studiengängen das Modul vorkommt.

Rechtsverbindlichkeit

Modulbeschreibungen dienen der Erhöhung der Transparenz und der besseren Orientierung über die Module eines Studiengangs. Rechtsverbindlich ist ausschließlich die einschlägige *(Fach-)Prüfungs- und Studienordnung.*

Inhaltsverzeichnis

Alphabetisches Verzeichnis befindet sich auf Seite 25

Fachwissenschaft (FW)
Fak112522: Bachelor-Seminar (25W) ... 4
Fak127933: Konzepte der Programmierung (25W) ... 5
Fak127934: Rechnerarchitektur und Rechnernetze (25W) ... 6
Fak127935: Algorithmen und Datenstrukturen I (25W) ... 7
Fak127936: Betriebssysteme (25W) ... 8
Fak127937: Theoretische Informatik I (25W) ... 9
Fak127938: Parallele und Verteilte Systeme I (25W) ... 10
Fak127951: Datenbanken und Informationssysteme I (25W) ... 11
Fak127952: Software Engineering (25W) ... 12
Wahlmodul
Fak127954: Wissensbasierte Systeme (25W) ... 15
Fak127976: Data Analysis and Deep Learning in Python (25W) ... 16
Fak723587: Reinforcement Learning for Scientists and Engineers (25W) ... 17
Fak128039: Mustererkennung (25W) ... 19
Fak112537: LAI 911: Programmierpraktikum (16S) ... 20
Fak127986: Formale Grundlagen der Informatik (25W) ... 21
Fak112539: Softwarepraktikum für Lehramtsstudierende (20W) ... 22
Fachdidaktik
Fak112949: LAI 211: Informatik – Lehren und Lernen (16S) ... 23
Fak112540: Bachelorarbeit - Informatik Lehramt (16S) ... 24

Fak112522: Bachelor-Seminar

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	30	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	jedes Semester	Buschek, Daniel; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Präsentation	1	5

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Voraussetzungen

Abhängig vom Thema

Lernziele

Der Studierende erwirbt methodische Kompetenzen im Bereich wissenschaftlicher Arbeitstechniken (insbesondere Literaturstudium, Präsentations- und Schreibtechniken) sowie kommunikative Kompetenzen in der mündlichen und schriftlichen Darstellung von wissenschaftlichen Inhalten.

Lerninhalte

Ein ausgewähltes Thema aus der Informatik wird in einer schriftlichen Ausarbeitung dargestellt und mündlich präsentiert.

Art und Umfang der Lehrveranstaltungen

Seminar (2 SWS)

Literatur:

Bernd Weidenmann: Gesprächs- und Vortragstechnik, Beltz-Verlag, 4. Auflage, 2006
Wolfram E. Rossig: Wissenschaftliches Arbeiten, Rossig Verlag, 6. Auflage, 2006
Peter Rechenberg: Technisches Schreiben. (Nicht nur) für Informatiker, Hanser Fachbuchverlag, 3. Auflage, 2006
Weitere Literatur abhängig vom Thema

zum Inhaltsverzeichnis

Fak127933: Konzepte der Programmierung

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	90	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Wintersemester	Mayer, Ruben; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung	1	8

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Lernziele

Ziel der Veranstaltung ist, den Studierenden ein fundiertes Verständnis der Programmierung zu vermitteln, das im weiteren Studium als Fundament für die Informatik-Ausbildung dient. Dabei dient Java als Beispielsprache.
Der Vorkurs richtet sich insbesondere an Studierende ohne Programmiervorkenntnisse, wird aber allen Studierenden empfohlen. Er dient dazu, der fachlichen Diversität der Studierenden zu begegnen. Als Einstiegssprache wird Python verwendet.
Der Schwerpunkt liegt auf dem Erwerb von methodischen Kompetenzen: Durch das Verständnis fundamentaler Konzepte wie Kontroll- und Datenstrukturen, Methoden, Objektorientierung, Syntax, Typkonzept etc. sollen die Studierenden in die Lage versetzt werden, diese Konzepte bei der Umsetzung von Algorithmen in Programme einzusetzen und sich ferner in andere Programmiersprachen einzuarbeiten. Erste algorithmische Kompetenzen werden ebenfalls erworben. Sie legen die Grundlage für weiterführende Veranstaltungen (z.B. Algorithmen und Datenstrukturen).
Die Intensivübungen richten sich an Studierende, die die Modulprüfung im Sommersemester absolvieren und zur individuellen Förderung und Differenzierung.

Lerninhalte

Vorkurs: Grundbegriffe, Algorithmen, einfache Programme in Python
Einführung: Grundbegriffe
Algorithmen: wesentliche Eigenschaften, erste Beispiele
Programme: Umsetzung von Algorithmen in eine Programmiersprache
Syntax: EBNF, Ableitungsbäume, Syntaxdiagramme
Elementare Datentypen: ganze Zahlen, Gleitpunktzahlen, Zeichen, Wahrheitswerte
Ausdrücke: Syntax, Prioritäten, Auswertungsbäume
Anweisungen: Zuweisungen, Kontrollstrukturen, Flussdiagramme, strukturierte Programmierung
Methoden: Syntax, Aufruf, Ausführung, Aufrufstapel
Rekursion: Klassifikation von Rekursionsarten, Ausführung, Elimination von Rekursion
Strukturierte Datentypen: Arrays und Verbunde; Eigenschaften von Objekttypen, Referenzen
Objekte und Klassen: Grundbegriffe, Felder, Methoden, Konstruktoren, Klassen- vs. Objekteigenschaften, abstrakte Klassen
Vererbung: Einfachvererbung, Substituierbarkeit, Polymorphie, Vererbungsregeln
Schnittstellen: Konzept, Abgrenzung gegen abstrakte Klassen, Implementierung von Schnittstellen, Mehrfachvererbung auf Schnittstellen
Generizität: generische Datentypen, Abgrenzung von Generizität und Vererbung
Ausnahmebehandlung: Ausnahmeobjekte, Ausnahmebehandler, geschützte Blöcke
Funktionale Programmierung in Java
Inhalt der Intensivübung: Programmierung ausgewählter Kapitel der Vorlesung

Art und Umfang der Lehrveranstaltungen

Vorlesung (4 SWS)
Übungen (2 SWS)
Intensivübung (2 SWS, optional)
Vorkurs (1 SWS, optional)

Literatur:

K. Echtle, M. Goedicke: Lehrbuch der Programmierung, dpunkt.verlag, Heidelberg, 2000
H.P. Mössenböck: Sprechen Sie Java?, 5. Auflage, dpunkt.Verlag, Heidelberg, 2014
D.J. Barnes, M. Kölling: Objektorientierte Programmierung mit Java - Eine praxisnahe Einführung mit BlueJ, Pearson Studium, München (2003)
H. Balzert: Objektorientierte Programmierung mit Java 5, Spektrum Akademischer Verlag, Heidelberg (2005)
R. Sedgewick, K. Wayne: Einführung in die Programmierung mit Java, Pearson, München, 2011

zum Inhaltsverzeichnis

Fak127934: Rechnerarchitektur und Rechnernetze

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	90	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Wintersemester	Rauber, Thomas; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung	9	7,2
semesterbegleitende Aufgaben	1	0,8

Lernziele

Das Ziel der Veranstaltung besteht in der Vermittlung grundlegender technologischer Kompetenz mit dem Schwerpunkt der Vermittlung von Kenntnissen des Aufbaus von Rechnersystemen mit Speicherhierarchie und Prozessoren. Vermittelt werden auch formale und algorithmische Kompetenzen, die zur Analyse und dem Entwurf digitaler Schaltkreise befähigen, sowie Design- und Realisierungskompetenzen zum Entwurf komplexer Schaltkreise. Durch Erlernen qualitativer Analyseverfahren zur Leistungsbewertung von Rechnersystemen und Rechnernetzen werden grundlegende methodische Kompetenzen im Bereich Rechnersysteme und Rechnernetze erworben, die Grundlagen für weiterführende Veranstaltungen legen.

Lerninhalte

Leistungsbewertung von Rechnern und grundsätzlicher Rechneraufbau
Maschinensprachen als Schnittstelle zwischen Hardware und Software
Zahlendarstellungen und Rechnerarithmetik
Entwurf digitaler Schaltkreise
Kombinatorische Schaltungen
Konstruktion von Speicherelementen
Speicherorganisation und Prozessorganisation,
Grundlagen und Leistungsbewertungen von Rechnernetzen
Schichtenprotokolle und Kommunikationsablauf
Wichtige Protokolle von Verbindungsschicht, Netzwerkschicht und Transportschicht

Art und Umfang der Lehrveranstaltungen

Vorlesung (4 SWS)
Übungen (2 SWS)

Literatur:

Patterson/Hennessy: Computer Organization & Design, Morgan Kaufmann, 6. Auflage, 2020
Hennessy/Patterson: Computer Architecture, Morgan Kaufmann, 7. Auflage, 2026
Oberschelp/Vossen: Rechneraufbau und Rechnerstrukturen, Oldenbourg Verlag, 10. Auflage, 2006
Kurose/Ross: Computer Networking, Addison Wesley, 8. Auflage, 2021
Peterson, Davie: Computer Networks, Morgan Kaufmann, 6. Auflage, 2021.

zum Inhaltsverzeichnis

Fak127935: Algorithmen und Datenstrukturen I

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	90	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Sommersemester	Knauer, Christian; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur	1	8

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Voraussetzungen

Inhalte der Module
- Konzepte der Programmierung
- Formale Grundlagen der Informatik

Lernziele

Die Studenten sollen lernen, Daten zu strukturieren und dynamisch zu repräsentieren. Wichtig ist hierbei die enge Verknüpfung dieser Datenstrukturen und der hierauf angewandten Algorithmen. Ein weiteres Ziel ist die Vermittlung von Kenntnissen zur Komplexitätsanalyse von Algorithmen (methodische Kompetenz).
In der Fragestunde können Lehrinhalte beim Dozenten spezifisch nachgefragt und individuell nachgearbeitet werden. Sie dient der individuellen Förderung.

Lerninhalte

Listen, Keller, Such- und Sortierverfahren, binäre Bäume, Suchbäume (AVL), Graphen, Hash-Verfahren, Komplexität von Algorithmen, Algorithmentheorie.

Art und Umfang der Lehrveranstaltungen

Vorlesung (4 SWS)
Übung (2 SWS)
Fragestunde (1 SWS, optional)

Literatur:

Goodrich, Tamassia: “Data Structures & Algorithms in Java” (4th Ed.), 2006.
Saake, Sattler: “Algorithmen und Datenstrukturen – Eine Einführung mit Java” (3. Aufl.), 2006.
Cormen, Leiserson, Rivest, Stein: “Introduction to Algorithms” (2nd Ed.), 2001.

zum Inhaltsverzeichnis

Fak127936: Betriebssysteme

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	45	105

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	Wintersemester	Henrich, Dominik; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung zu Betriebssysteme	17	4,25
semesterbegleitende Aufgaben zu Betriebssyteme	3	0,75

Studienleistung: Aktive Teilnahme an den Veranstaltungen und Selbststudium
Prüfungsleistung: Siehe FPSO

Voraussetzungen

- Konzepte der Programmierung
- Rechnerarchitektur und Rechnernetze
- Algorithmen und Datenstrukturen I

Lernziele

Lernziele des Moduls sind das Verständnis des grundsätzlichen Aufbaus von Betriebssystemen, das Verständnis der eingesetzten Verfahren, sowie das Lernen der sinnvollen Auswahl und des Einsatz von Betriebssystemen. Es werden keine Einschränkung auf ein bestimmtes Betriebssystem vorgenommen und auch keine Implementierungsdetails vermittelt. Allgemein werden Methoden zur effizienten Verwaltung von zeitlichen bzw. räumlichen Ressourcen vermittelt.

Lerninhalte

Einleitung: Definition, Schnittstellen, Historie, Aufbau;
Prozessverwaltung: Prozesse/Threads, Prozesssynchronisation, -scheduling, -kommunikation;
Speicherverwaltung: Speicherbelegung, -adressierung, -seiten, Segmentierung, Caches, Schutz;
Dateiverwaltung: Dateisysteme, -namen, -attribute, -funktionen, -organisation;
Ein-/Ausgabeverwaltung: E/A-Aufgaben, Gerätemodelle, Treiber;
Systemsicherheit: Kryptographie, Authentifikation, Angriffe, Schutz;

Art und Umfang der Lehrveranstaltungen

Vorlesung (2 SWS)
Übung (1 SWS)

Anmerkungen

Medienformen: Multimedia-Präsentation als Vortrag und Ausdruck, Übungsblätter mit Korrektur, Tafelübungen

Literatur:

Tanenbaum A. S.: „Moderne Betriebssysteme“. 2. überarbeitete Auflage, München, Pearson Studium, 2002. –> 82/ST 260 T164 M6(3)+3. Dritte Auflage von 2009

zum Inhaltsverzeichnis

Fak127937: Theoretische Informatik I

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch/Englisch	Einsemestrig	90	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Sommersemester	

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung zu Theoretische Informatik I	1	8
semesterbegleitende Aufgaben zu Theoretische Informatik I	0	

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Voraussetzungen

Die Fähigkeit, mathematisch und präzise zu kommunizieren wird benötigt.

Inhalte des Moduls Analysis 1 (Mathematik) oder aller folgende Module:
- Formale Grundlagen der Informatik
- Konzepte der Programmierung
- Algorithmen und Datenstrukturen (spätestens gleichzeitig mit Theoretische Informatik I hören)

Lernziele

Die Studierenden sollen die Grundlagen von regulären, kontextfreien, berechenbaren und effizient berechenbaren formalen Sprachen verstehen und erklären können. Sie sollen in der Lage sein, bestimmte Sprachen in Klassen ein- zuordnen und zu beweisen warum, oder warum nicht sie Mitglied einer Klasse sind.

Die Studierenden sollen in der Lage sein, die Verfahren aus der Vorlesung zu erklären und auf Beispiele anzuwenden. Die Studierenden sollen die Ergebnisse der Vorlesung verstehen und anwenden können und ihre Beweise verstehen.

Lerninhalte

- Formale Sprachen
- Berechenbarkeit
- Automaten, Grammatiken und die Chomsky-Hierarchie
- Theoretische Berechnungsmodelle
- Entscheidbarkeit
- Komplexitätstheorie

Art und Umfang der Lehrveranstaltungen

Vorlesung (4 SWS)
Übungen (2 SWS)
Fragestunde (1 SWS, optional)

Literatur:

Hopcroft, Motwani, Ullman: „Introduction to Automata Theory, Languages, and Computation“, 2000.
Asteroth, Baier: „Theoretische Informatik. Eine Einführung in Berechenbarkeit, Komplexität und formale Sprachen mit 101 Beispielen“, 2002.
Wegener: „Theoretische Informatik - Eine algorithmenorientierte Einführung“, 2. Aufl., 1999.
Schöning: „Theoretische Informatik kurzgefasst“, 4. Aufl, 2001.
Sipser: „Introduction to the Theory of Computation“, 2nd ed., 2006.

zum Inhaltsverzeichnis

Fak127938: Parallele und Verteilte Systeme I

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	45	105

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	Wintersemester	Rauber, Thomas; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung zu Parallele und Verteilte Systeme I	9	4,5
semesterbegleitende Aufgaben zu Parallele und Verteilte Systeme I	1	0,5

Voraussetzungen

Inhalte der Module
– Konzepte der Programmierung
– Rechnerarchitektur und Rechnernetze

Lernziele

Ziel der Veranstaltung ist es, den Studenten grundlegende Techniken der parallelen und verteilten Programmierung zu vermitteln. Dabei werden besondere methodische Kompetenzen erworben: Durch das Verständnis grundlegender Problemstellungen wie Lastverteilung und Skalierbarkeit und die Vermittlung von Synchronisations- und Kommunikationstechniken werden die Studenten in die Lage versetzt, parallele Algorithmen zu entwerfen und mit Hilfe von Kommunikations- und Threadbibliotheken in effiziente parallele und verteilte Programme umzusetzen. Dabei werden sowohl gemeinsame als auch verteilte Adressräume erlernt.

Lerninhalte

Architektur und Verbindungsnetzwerke für parallele Systeme
Leistung, Laufzeitanalyse und Skalierbarkeit paralleler Programme
Programmier- und Synchronisationstechniken für gemeinsamen Adressraum mit Multi-Threading
Koordination paralleler und verteilter Programme
Anwendung der Programmiertechniken auf komplexe Beispiele aus verschiedenen Anwendungsgebieten
Programmiertechniken für verteilte Adressräume und Message-Passing und Realisierung typischer Kommunikationsmuster

Art und Umfang der Lehrveranstaltungen

Vorlesung (2 SWS)
Übung (1 SWS)

Literatur:

Hennessy/Patterson: Computer Architecture, Morgan Kaufmann, 6. Auflage, 2017
Rauber/Rünger: Parallele Programmierung, Springer, 3. Auflage 2012 (deutsch)
Rauber/Rünger: Parallel Programming, Springer, 3. Auflage 2023 (englisch)
Kurose/Ross: Computer Networking, Addison Wesley, 8. Auflage, 2021
Coulouris/Dollimore/Kindberg: Distributed Systems, 5th Edition, Addison Wesley, 2011

zum Inhaltsverzeichnis

Fak127951: Datenbanken und Informationssysteme I

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	90	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Sommersemester	Jablonski, Stefan; Prof. Dr.-Ing.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung	1	8

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Lernziele

Ziel ist die Vermittlung grundlegender Kenntnisse zum Entwurf von (relationalen) Datenbanken. Die Studierenden sollen Analyse-, Entwurfs und Realisierungskompetenzen vermittelt bekommen, so dass sie selbständig eine Anwendungssituation analysieren und darauf aufbauend ein datenbankgestützte Anwendungen entwickeln können.
Daneben sollen Grundkenntnisse bezüglich des Aufbaus und des Betriebs von Datenbanksystemen vermittelt werden, so dass die Studierenden einen prinzipiellen Einblick in die Technologie von Datenbanksystemen bekommen.
Über den Übungsbetrieb sollen die Studierenden den praktischen Umgang mit Datenbanken und deren Anwendungen erlernen.
In den Intensivübungen werden darüber hinaus programmiertechnische Fähigkeiten vermittelt und Studierende individuell gefördert.

Lerninhalte

Entwurf von Datenbanksystemen: Aufbau konzeptioneller Schemata (Von Entity-Relationship-Diagrammen zu Relationen), Normalisierung, Relationenalgebra, Einführung in SQL, Verwendung von Datenbanksystemen (SQL als DB-Schnittstelle), Objektrelationale Datenbanksysteme; Aufbau von Datenbanksystemen (Architektur), Einführung ins Transaktionsmanagement; Aufbau von Informationssystemen (Arten von Informationssystemen), Anwendungen von Datenbanken in den Bereichen Bio-, Ingenieur- und Umweltinformatik; Vorstellung von Beispielen und Fallstudien.
Inhalt der Intensivübung: Programmierung ausgewählter Kapitel der Vorlesung.

Art und Umfang der Lehrveranstaltungen

Vorlesung (4 SWS)
Übung (2 SWS)
Intensivübung (2 SWS, optional)

Literatur:

Härder, T.; Rahm, E.: Architektur von Datenbanksystemen. Springer-Verlag
Gray, J.; Reuter, A.: Transaction Systems. Morgan Kaufman
Conolly, T.; Begg, C.: Database Systems: A Practical Approach to Design, Implementation and Management. Addison-Wesley Longman

zum Inhaltsverzeichnis

Fak127952: Software Engineering

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	90	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Sommersemester	Baltes, Sebastian; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung zu Software Engineering	9	7,2
semesterbegleitende Aufgaben zu Software Engineering	1	0,8

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Voraussetzungen

Inhalte der Module
- Konzepte der Programmierung
- Bachelor-Praktikum

Lernziele

Das Modul vermittelt ein systematisches und vertieftes Verständnis von fundamentalen Konzepten sowie ausgewählten Verfahren, Methoden und Werkzeugen des modernen Software-Engineerings. Die Studierenden sollen eine Auswahl an Verfahren, Methoden und Werkzeugen im Rahmen von Beispielen und Fallstudien anwenden. Insbesondere werden so Analyse- und Design-Kompetenzen vermittelt, die für die Entwicklung von Softwaresystemen von zentraler Bedeutung sind. Darüber hinaus werden methodische Kompetenzen unter anderem in Projektmanagement und Qualitätssicherung vermittelt.

Lerninhalte

Software Engineering: Geschichte, Begriffe, Programmieren im Kleinen vs. Programmieren im Großen, “No Silver Bullet”.

Software Development Processes: Software Development Life Cycle, historische Prozess-modelle (z.B., Wasserfallmodell, Spiralmodell), V-Modell, agile Prozessmodelle (Scrum, SA-Fe), DevOps.

Software Development Tools: Versionsverwaltungssysteme, kollaborative Softwareentwick-lung, integrierte Entwicklungsumgebungen, Build Automation, statische Analyse, Continuous Integration und Delivery, AI Coding Assistants.

Requirements Engineering: Requirements Elicitation, Analysis, Specification, Validation, Management; Stakeholder Groups, Anwendungsfälle, User Stories, Akzeptanzkriterien.

Software Architecture: Top-down vs. bottom-up Architecture, Module, Komponenten, Dekomposition, Architekturmuster, 4+1 Model.
Requirements to Code: Domain-driven design, Behavior-Driven Development, Test-Driven Development.

Software Documentation and Visualization: Dokumentationsökosysteme, Documentation as Code, Architectural Decision Records, Einführung in die Softwarevisualisierung.

Object-Oriented Modeling: Unified Modeling Language, Entwurfsmuster.

Software Quality: Antipatterns, Clean Code, funktionale vs. nicht-funktionale Qualität, Per-formance, Sicherheit, Usability.

Code Comprehension and Code Reviews: Mentale Modelle, kognitive Komplexität, lesba-ren Code schreiben, Coding Conventions, Kommentare im Code, Pull Request Reviews, Code Review Best Practices.

Software Testing: Begriffe, Regressionstests, Black-Box- vs. Glass-Box-Testing, risikoba-siertes Testen, Testbarkeit von Anforderungen, Testpyramide, manuelles vs. automatisiertes Testen, exploratives vs. strukturiertes Testen, Unit vs. Integration vs. System Testing, Code Coverage, Test Doubles, testbaren Code schreiben.

Managing Software Projects: Menschen managen, Risiken managen, Projekte managen, Aufwandsschätzung, Kanban Boards, Definition of Ready/Done, Gantt-Diagramme, Critical Path Method, Ethische Aspekte.

Open Source and Inner Source: Definition, Urheberrecht, Softwarelizenzen, Open-Source-Ökosysteme, Software Supply Chains, Open Source von und in Unternehmen.

Software Evolution and Maintenance: Evolution vs. Wartung, technische Schulden, De-bugging, Reverse-engineering, Reengineering, Refactoring, gute Bug Reports verfassen, Lehman’s Laws, Migrationen.

Software Measurement and Metrics: Softwaremetriken (insbesondere Komplexitätsmetri-ken), Geschäftsmetriken, Goodhart’s Law.

Cloud-native Software Development: Skalierbarkeit, Elastizität, Virtualisierung, Containe-rization.

Software Analytics: Einführung ins Empirical Software Engineering, Software Observability.

Art und Umfang der Lehrveranstaltungen

Vorlesung (4 SWS)
Übung (2 SWS)

Literatur:

Die Vorlesung basiert auf eigenen Materialien und einer Vielzahl von Quellen.
T. Winters, T. Manshreck, H. Wright: Software Engineering at Google - Lessons Learned from Programming Over Time, O'Reilly Media, 2020.
K. Wiegers, J. Beatty: Software Requirements, 3rd Edition, Microsoft Press, 2013.
P. Clements, F. Bachmann, L. Bass, D. Garlan, J. Ivers, R. Little, P. Merson, R. Nord, J. Stafford: Documenting Software Architectures - Views and Beyond, 2nd Edition, Addison-Wesley Professional, 2010.
M. Fowler: Patterns of Enterprise Application Architecture, Pearson International, 2002.
G. Hohpe. B. Woolf: Enterprise Integration Patterns - Designing, Building, and Deploying Messaging Solutions, Addison Wesley, 2003.
G. Booch, R.A. Maksimchuk. M.W. Engle, B.J. Young, J. Conallen, K.A. Houston: Object-Oriented Analysis and Design with Applications, Addison-Wesley Professional, 2007.
E. Gamma, R. Helm, R. Johnson, J. Vlissides: Design Patterns - Elements of Reusable Object-Oriented Software, Addison-Wesley Professional, 1994.
M. Fowler: Refactoring - Improving the Design of Existing Code, 2nd Edition, Addison-Wesley Professional, 2018.
M. Aniche: Effective Software Testing - A Developer's Guide, Manning Publications, 2022.
E. Evans: A Domain-Driven Design - Tackling Complexity in the Heart of Software, Addison-Wesley Professional, 2003.
C. Sims, H.L. Johnson: Scrum - A Breathtakingly Brief and Agile Introduction, Dymaxicon, 2012.
R.C. Martin: Clean Code - A Handbook of Agile Software Craftsmanship, Prentice Hall, 2008.
S. McConnell: Code Complete - A Practical Handbook of Software Construction, Microsoft Press, 2004.
Hunt, D. Thomas: The Pragmatic Programmer - From Journeyman to Master, Addison-Wesley Professional, 1999.
G. Hohpe: The Software Architect Elevator - Redefining the Architect's Role in the Digital Enterprise, O'Reilly Media, 2020.
J. Ousterhout: A Philosophy of Software Design, Yaknyam Press, 2021.

zum Inhaltsverzeichnis

Fak127954: Wissensbasierte Systeme

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	45	105

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	Wintersemester	Guthe, Michael; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung	17	4,25
semesterbegleitende Aufgaben	3	0,75

Voraussetzungen

- Algorithmen und Datenstrukturen I

Lernziele

In der Veranstaltung werden Fertigkeiten und Kenntnisse der wichtigsten KI-Methoden und deren Anwendung in der Praxis vermittelt. Dabei soll unter anderem die Programmierung wissensbasierter Inferenzsysteme in Prädikatenlogik, mit der Programmiersprache Prolog erlernt werden. Des Weiteren werden Wissensrepräsentationsformen sowie Problemlösungs-, Such- und Planungsalgorithmen vermittelt. Die Studenten sollen einen Überblick über gebräuchliche Methoden des Schätzens, wie zum Beispiel Bayes'sche Inferenz und Dempster-Shafer Theorie erhalten.

Lerninhalte

Die Veranstaltung beschäftigt sich zunächst mit dem Programmieren in Prolog, der Prädikatenlogik und Zwangsbedingungen. Dann werden Wissen, Wissensrepräsentation und Inferenz sowie die Struktur wissensbasierter Systeme besprochen. Zum Abschluss werden verschiedene Verfahren zum wahrscheinlichkeitsbasierten Schließen, wie zum Beispiel Bayes'sche Inferenz und Dempster-Shafer Theorie vorgestellt und untersucht.

Art und Umfang der Lehrveranstaltungen

Vorlesung (2 SWS),
Übung (1 SWS)

Literatur:

Nilsson: Artificial Intelligence (Morgan); Luger: Künstliche Intelligenz (Pearson); Thayse: From Standard Logic to Logic Programming (Wiley)

zum Inhaltsverzeichnis

Fak127976: Data Analysis and Deep Learning in Python

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Englisch	Einsemestrig	60	90

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	jedes Semester	Müller, Jörg; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung	17	4,25
semesterbegleitende Aufgaben	3	0,75

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Lernziele

Die Studierenden lernen, numerische Programme in Python zu entwickeln. Dazu lernen Sie Python als Programmiersprache und als Umgebung für wissenschaftliches Rechnen. Verwendete Bibliotheken sind NumPy, SciPy, Matplotlib, Pandas, und TensorFlow/Keras.

Lerninhalte

- Die Python Programmiersprache, Datentypen, Kontrollstrukturen, Funktionen, Objektorientierte Programmierung, Debugging.
- Algorithmen: Rekursion, Dynamische Programmierung, Newton's Methode.
- Rechnen mit Matrizen: Lineare Algebra mit NumPy, Matrixfaktorisierungen, Eigenvektoren und –werte, Diagonalisierung, SVD, Methode der kleinsten Quadrate, Pseudoinverse.
- Datenanalyse: Pandas, Clustering, Plotten. Neuronale Netze und Deep Learning.

Art und Umfang der Lehrveranstaltungen

Vorlesung (2 SWS)
Übung (2 SWS)

Literatur:

- Wes McKinney: „Python for Data Analysis", O'Reilly, Second Edition

zum Inhaltsverzeichnis

Fak723587: Reinforcement Learning for Scientists and Engineers

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Englisch	Einsemestrig	45	105

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	Wintersemester	Faisal, Aldo; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur/Hausarbeit/Präsentation	1	5

Voraussetzungen

- Höhere Mathematik I
- Höhere Mathematik II
- Statistische Methoden I und II, oder

Mathematische Grundlagen der Datenanalyse

Lernziele

Upon completion of this module, students will be able to:
Describe the core principles of learning in autonomous systems and specifically learning through interaction.
Calculate mathematical solutions to problems using reinforcement learning theory.
Compare and contrast a range of reinforcement learning approaches.
Propose solutions to decision making problems using knowledge of the state-of-the-art.
Translate mathematical concepts into software to solve practical problems using Python and PyTorch. Given basic programming knowledge students will be eased into using PyTorch (a world-standard for deep learning use) and upported by providing code templates to get going quickly.
Evaluate the performance of a range of methods and propose appropriate improvements.
Prepare clear visualisations of complex data to assist with evaluation.

Lerninhalte

Reinforcement learning (RL) is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act optimally through experience. RL is at the heart of current developments in AI ranging from robot learning, self-driving cars, and healthcare, with famous exemplars including
The course provides both an introduction in reinforcement learning and then advances to state-of-the-art reinforcement learning across three core skills: theory, implementation, and evaluation. Students will learn the fundamentals of both tabular reinforcement learning and deep reinforcement learning, and will gain experience in designing and implementing these methods for practical applications.
Specifically, students will:
Learn the theoretical foundations of reinforcement learning Markov Decision Processes
& Dynamic Programming).
Learn the algorithmic foundations of reinforcement learning (Temporal Difference and Monte-Carlo learning).
Gain experience in framing low-dimensional problems and implementing solutions using tabular reinforcement learning.
Learn about the motivation behind deep reinforcement learning and its relevance to high-dimensional applications, such as robotics and healthcare applications (Deep Q Networks (DQN).
Discover the state-of-the-art deep reinforcement learning algorithms such as Deep Determinstic Policy Gradient (DDPG).
Proximal Policy Optimisation (PPO), and Soft Actor Critic (SAC).
Learn to Implement and experiment with a range of different algorithms in Python and PyTorch, and learn how to visualise and evaluate their performance.
The first half of the course will include (RL basics):
Introduction to Reinforcement Learning and its Mathematical Foundations
The Markov Decision Process Framework
Markov Reward Processes
The Policy
Markov Decision Processes
Dynamic Programming
Model-Free Learning & Control
Monte-Carlo Learning
Temporal Difference Learning
The second half of the course will include (Deep RL):
Motivation for function approximation:
High-dimensional state and action spaces
Continuous state and action spaces
Deep Q-learning:
Q update through back propagation
Experience replay buffer
Target and Q networks
Policy gradients:
The REINFORCE algorithm
Policy update through back propagation
Advanced topics:

Model-based reinforcement learning

Art und Umfang der Lehrveranstaltungen

Vorlesung (2 SWS)
Übung (2 SWS)

Literatur:

Literature recommendations for preparation:
Mathematics for machine learning, Deisenroth, Faisal & Ong, Cambridge University Press, PDF freely available under https://mml-book.github.io/

Literature recommendations for study:
Reinforcement learning : an introduction, Sutton, Richard S., Second edition., The MIT Press
Algorithms for reinforcement learning, Szepesva#ri, Csaba, Morgan & Claypool Publishers
Deep learning , Goodfellow, Ian, author., The MIT Press

zum Inhaltsverzeichnis

Fak128039: Mustererkennung

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch/Englisch	Einsemestrig	45	105

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	Wintersemester	Henrich, Dominik; Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur oder mündliche Prüfung	17	4,25
semesterbegleitende Aufgaben	3	0,75

Voraussetzungen

Inhalte der Module
- Höhere Mathematik I
- Höhere Mathematik II
- Statistische Methoden I und II oder Mathematische Grundlagen der Datenanalyse
- Konzepte der Programmierung
- Algorithmen und Datenstrukturen I

Lernziele

Das Modul vermittelt ein systematisches und vertieftes Verständnis der Methoden zur Erkennung bzw. Klassifikation von Mustern in einer Menge von Daten. Die Anwendungen liegen beispielsweise in den Bereichen der Objekterkennung, Schrifterkennung, Spracherkennung, Gestenerkennung und Gesichtserkennung.

Lerninhalte

- Bayes'sche Klassifikation
- Parameterschätzung
- Parameterfreie Klassifikation
- Lineare Klassifikation
- Vorwärtsgerichtete Neuronale Netze
- Rückgekoppelte Neuronale Netze
- Nicht-metrische Klassifikation
- Überwachtes Lernen
- Unüberwachtes Lernen

Art und Umfang der Lehrveranstaltungen

Vorlesung (2 SWS)
Übung (1 SWS)

Anmerkungen

Medienformen: Multimedia-Präsentation als Vortrag und Ausdruck, Übungsblätter mit Korrektur, Tafelübungen

Literatur:

Duda R., Hart P., Stork D.: „Pattern Classification", Wiley, 2. Auflage, 2001. Signatur: 819 ST 282 D 844 (2)

zum Inhaltsverzeichnis

Fak112537: LAI 911: Programmierpraktikum

Gültig ab: 01.04.2016

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig		

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
5	Sommersemester	Korch, Matthias; apl. Prof. Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Programmierpraktikum	1	5

150 h gesamt (60 h Präsenz, 90 h Vorbereitung und Übungsbearbeitung)

Voraussetzungen

Keine

Lernziele

Vermittlung grundlegender Programmiertechniken; Einführung in die funktionale Programmierung; Einführung in die Programmierung mit Skriptsprachen

Lerninhalte

Vermittlung grundlegender Programmiertechniken; Einführung in die funktionale Programmierung; Einführung in die Programmierung mit Skriptsprachen

Art und Umfang der Lehrveranstaltungen

Praktikum

zum Inhaltsverzeichnis

Fak127986: Formale Grundlagen der Informatik

Gültig ab: 01.10.2025

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	90	150

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Sommersemester	Stehn, Fabian; Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Klausur	1	8

Studien-/Prüfungsleistungen: Aktive Teilnahme an den Veranstaltungen und Selbststudium

Voraussetzungen

Umgang mit Gleichungssystemen, insb. Gauß-Verfahren

Lernziele

Die Studierenden kennen die grundlegenden Definitionen und Eigenschaften der Logiken, die in der Vorlesung behandelt wurden. Sie können die Verfahren aus der Vorlesung auf Beispiele anwenden. Sie sind in der Lage, umgangssprachliche Texte formal zu modellieren.
Die Studierenden kennen formale Methoden aus dem Bereich der diskreten Mathematik. Sie kennen Anwendungen dieser Methoden auf Probleme der Informatik. Die Studierenden können die vorgestellten Methoden selber einsetzen. Sie sind in der Lage zu erkennen, wann ähnliche Situationen für die Anwendung der bekannten Verfahren vorliegen. Die Studierenden können einfache formale Beweise aus dem Bereich der diskrete Mathematik durchführen.
In der Fragestunde können Lehrinhalte beim Dozenten spezifisch nachgefragt und individuell nachgearbeitet werden.
Im Mathematik-Vorkurs werden die wichtigsten mathematischen Grundlagen aus der gymnasialen Oberstufe wiederholt und vertieft. Ziel ist, potentielle Unterschiede in den Vorkenntnissen der Studenten auszugleichen und eventuell vorhandene Wissenslücken aufzufüllen.

Lerninhalte

- Diskrete Strukturen:
Beweismethoden (Direkter Beweis, Indirekte Beweise: Kontraposition und Widerspruch, vollständige Induktion, Ringschluss, Beweis durch Zeugen), Mengen, Relationen, Funktionen, Landau-Symbole, Abzählbarkeit und Überabzählbarkeit, elementare Zahlentheorie, ggT, kgV, Fundamentalsatz der Zahlentheorie, Kombinatorik.
- Logik und Modellierung:
Aussagenlogik, Modallogik und Prädikatenlogik. Syntax, Semantik und Eigenschaften. Übungen in Modellierung.
- Mathematik-Vorkurs:
Aussagen- und Prädikatenlogik; Lösen von Gleichungen; Mengenlehre; Funktionen; Komplexe Zahlen; Kombinatorik; Relationen

Art und Umfang der Lehrveranstaltungen

Vorlesung 'Diskrete Strukturen' (2 SWS)
Übung 'Diskrete Strukturen' (1 SWS)
Vorlesung 'Logik und Modellierung' (2 SWS)
Übung 'Logik und Modellierung' (1 SWS)
Fragestunde (1 SWS, optional)
Mathematik-Vorkurs (1 SWS als Block vor Vorlesungsbeginn, optional)

Literatur:

Diskrete Strukturen:
- Kurt Ullrich Witt: Mathematische Grundlagen der Informatik
- Kurt Ullrich Witt: Elementare Kombinatorik für die Informatik
Logik und Modellierung:
- Martin Kreuzer, Stefan Kühling: Logik für Informatiker
- Uwe Schöning: Logik für Informatiker

zum Inhaltsverzeichnis

Fak112539: Softwarepraktikum für Lehramtsstudierende

Gültig ab: 01.10.2020

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Einsemestrig	60	120

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
7	jedes Semester	Ehmann, Matthias; Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Softwarepraktikum für Lehramtsstudierende	1	7

Modulprüfung

Voraussetzungen

INF 107 - Konzepte der Programmierung
INF 109 - Algorithmen und Datenstrukturen I

Lernziele

Die Studierenden entwickeln unter Anleitung kleinere Softwaresysteme in kleinen Gruppen. Im Vordergrund steht der Erwerb von algorithmischen, Design- und Realisierungskompetenzen. Die Studierenden sollen in die Lage versetzt werden, Software beschränkten Umfangs und beschränkten Schwierigkeitsgrads syste-matisch zu entwickeln (methodische Kompetenz) sowie die von ihnen erarbeitete Lösung zu präsentieren (kommunikative Kompetenz).

Lerninhalte

Probleme werden analysiert, Anforderungen definiert, ein Systementwurf erstellt, und die Komponenten des Systementwurfs werden implementiert und getestet. Hinzu kommt die Präsentation der Lösungskonzepte.

Anmerkungen

Präsentation der Aufgabenstellung und der Lösungskonzepte

Literatur:

Wolfram E. Rossig: Wissenschaftliches Arbeiten, Rossig Verlag, 6. Auflage, 2006

zum Inhaltsverzeichnis

Fak112949: LAI 211: Informatik – Lehren und Lernen

Gültig ab: 01.04.2016

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch	Mehrere Semester	120	120

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
8	Wintersemester	Ehmann, Matthias; Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Modulprüfung Informatik - Lehren und Lernen	1	8

Voraussetzungen

Keine

Lernziele

Kennen und Anwenden der Konzepte des Lehrens und Lernens im Fach Informatik
Beherrschen der Denkweisen und Methoden der Informatik und ihre Übertragung auf den Schulunterricht
Konzeption und Gestaltung von Informatikunterricht

Lerninhalte

Informatische Bildung, Didaktische Prinzipien, Unterrichtsplanung und -gestaltung, Unterrichtsmethodik, eigenverantwortliches und eigenständiges Lernen, Einsatz von Informatiksystemen, Analyse und Bewertung von Lehr- und Lernprozessen im Unterricht
Einsatz von Informatiksystemen im unterrichtlichen Kontext
Methoden der Informatik, Techniken und Hilfsmittel zur Vermittlung informatischer Inhalte anhand geeigneter Praxisfelder des Informatikunterrichts

Art und Umfang der Lehrveranstaltungen

Vorlesung, Übung , Seminar

zum Inhaltsverzeichnis

Fak112540: Bachelorarbeit - Informatik Lehramt

Gültig ab: 01.04.2016

Lehrsprache	Dauer	Präsenzstudium (in Stunden)	Selbststudium (in Stunden)
Deutsch/Englisch	Einsemestrig	0	300

Leistungspunkte (LP)	Turnus	Modulverantwortliche(r)
10	jedes Semester	Ehmann, Matthias; Dr.

Leistungsnachweise

Titel:	Gewichtung:	LP:
Bachelorarbeit - Lehramt Informatik	10	10

Lernziele

In der Bachelorarbeit werden methodische Kompetenzen zum Lösen von Informatikproblemen erworben. Der Studierende erarbeitet das zu bearbeitende Thema selbständig und systematisch (Selbstkompetenz) und wird dabei vom Betreuer angeleitet und unterstützt. Die schriftliche Ausarbeitung dient dazu, die Ergebnisse der Arbeit in wissenschaftlicher Weise angemessen darzustellen (kommunikative Kompetenz). Dem Studierenden wird hierzu eine fachspezifische Einführung in das Verfassen wissenschaftlicher Arbeiten vermittelt.

Lerninhalte

Abhängig vom anbietenden Lehrstuhl wird ein Thema der Informatik bearbeitet und hinsichtlich einer konkreten Aufgabenstellung untersucht und beschrieben.

Art und Umfang der Lehrveranstaltungen

Selbständig unter Betreuung durchzuführende schriftliche Ausarbeitung

Literatur:

Marcus Deininger, Horst Lichter, Jochen Ludewig, und Kurt Schneider: Studienarbeiten, Vdf Hochschulverlag, 5. Auflage, 2005
Weitere Literatur abhängig vom gewählten Thema

zum Inhaltsverzeichnis

Index

Fak127935: Algorithmen und Datenstrukturen I (25W) ... 7
Fak112540: Bachelorarbeit - Informatik Lehramt (16S) ... 24
Fak112522: Bachelor-Seminar (25W) ... 4
Fak127936: Betriebssysteme (25W) ... 8
Fak127976: Data Analysis and Deep Learning in Python (25W) ... 16
Fak127951: Datenbanken und Informationssysteme I (25W) ... 11
Fak127986: Formale Grundlagen der Informatik (25W) ... 21
Fak127933: Konzepte der Programmierung (25W) ... 5
Fak112949: LAI 211: Informatik – Lehren und Lernen (16S) ... 23
Fak112537: LAI 911: Programmierpraktikum (16S) ... 20
Fak128039: Mustererkennung (25W) ... 19
Fak127938: Parallele und Verteilte Systeme I (25W) ... 10
Fak127934: Rechnerarchitektur und Rechnernetze (25W) ... 6
Fak723587: Reinforcement Learning for Scientists and Engineers (25W) ... 17
Fak127952: Software Engineering (25W) ... 12
Fak112539: Softwarepraktikum für Lehramtsstudierende (20W) ... 22
Fak127937: Theoretische Informatik I (25W) ... 9
Fak127954: Wissensbasierte Systeme (25W) ... 15