Modulhandbuch
Bachelor Informatik

Fakultät für Elektrotechnik, Informatik und Mathematik
Universität Paderborn

Version: 18. Mai 2017

Inhaltsverzeichnis

Kapitel 1

Beschreibung des Studiengangs Bachelor Informatik

Der Bachelor-Studiengang Informatik ist in zwei Studienabschnitte aufgeteilt. Er verfolgt das Ziel, breites Wissen in der Informatik zu vermitteln. Um den Studierenden bereits im Bachelor-Studiengang die Möglichkeit zu geben, individuellen Interessen zu folgen, bestehen nach einer Grundausbildung mit Kerninhalten im ersten Studienabschnitt, Wahlmöglichkeiten innerhalb der einzelnen Module im zweiten Studienabschnitt.

1.1 Schema der Modulbeschreibungen

Die Modulbeschreibungen sind nach folgendem Schema einheitlich strukturiert:

Modulname	<Name des Moduls>
Workload	<Gesamtaufwand in Stunden (Workload ECTS)>
Leistungspunkte	<Gesamtaufwand in Leistungspunkten ECTS>
Studiensemester	<Liste der Lehrveranstaltungen in diesem Modul mit Zielsemester>
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
<Liste der im Modul enthaltenen Lehrveranstaltungen mit Aufteilung des Workloads in Kontaktzeit und Selbststudium, Sprache in der die Veranstaltung gehalten wird, Winter- oder Sommersemester und ungefährer Gruppengröße.>	
Wahlmöglichkeiten innerhalb des Moduls	
<Liste der im Modul enthaltenen Wahlmöglichkeiten.>	
Teilnahmevoraussetzungen	
<Voraussetzungen für die Teilnahme am Modul.>	
Empfohlene Kenntnisse	
<Die Angaben sind als Empfehlungen zu verstehen, nicht jedoch als zu überprüfende Voraussetzungen.>	
Inhalte	
<Aufzählung der wesentlichen Inhalte der enthaltenen Lehrveranstaltungen.>	
Lernergebnisse / Fachkompetenzen	
<Aufzählung der erreichten Kenntnisse, Fähigkeiten und Fachkompetenzen.>	
Nichtkognitive Kompetenzen	
<Zusammenfassung aller nichtkognitiver Kompetenzen, die in den Lehrveranstaltungen des Moduls vermittelt werden.>	
Methodische Umsetzung	
<Angaben zu Sozialformen und didaktisch-methodischen Arbeitsweisen in den Veranstaltungen.>	
Prüfungsleistung (Dauer)	
<Form in Dauer der im Modul zu erbringenden Prüfungsleistung.>	
Modulteilprüfungen	
<Form der im Modul zu erbringenden Modulteilprüfung.>	

Studienleistung / qualifizierte Teilnahme
<Form der im Modul zu erbringenden Studienleistungen oder qualifizierter Teilnahmen.>
Voraussetzungen für die Teilnahme an der Prüfung
<Formale Voraussetzungen für Teilnahme an der Modulprüfung.>
Voraussetzungen für die Vergabe von Credits
<Formale Voraussetzungen für die Vergabe von Credits.>
Gewichtung für die Gesamtnote
<Gesamtgewichtung des Moduls bei der Berechnung des Notendurchschnitts.>
Modul wird in folgenden Studiengängen verwendet
<Liste der Studiengänge, in denen dieses Modul verwendet wird.>
Modulbeauftragte/r
<Verantwortlicher für das Modul.>
Lernmaterialien, Literaturangaben
<Angaben zu Literatur, Vorlesungsskripten, etc.>
Sonstige Hinweise
<Sonstige Hinweise.>

1.2 Liste der Organisationsformen

Die folgenden Organisationsformen werden in diesem Studiengang verwendet:

Abschlussarbeit

Proseminar plus wählbare Veranstaltung

Vorlesung mit Übung Eine Kombination aus Vorlesung und begleitenden Übungen, häufig mit praktischen Anteilen und Hausaufgaben.

Vorlesung mit Übung und Praktikum Eine Vorlesung mit Übungen wird mit einem Praktikumsteil kombiniert.

Vorlesungen, Seminare, Projekte Eine beliebige Kombination von Veranstaltungen für das Studium Generale.

1.3 Liste der Prüfungsformen

Die folgenden Organisationsformen werden in diesem Studiengang verwendet:

Abschlussarbeit Eine Abschlussarbeit soll zeigen, dass die Kandidatin oder der Kandidat die Fähigkeit besitzt, innerhalb einer bestimmten Frist ein Problem der Informatik auf der Grundlage wissenschaftlicher Methoden zu bearbeiten.

Klausur In einer Klausuren soll die Kandidatin bzw. der Kandidat nachweisen, dass sie bzw. er in einer vorgegebenen Zeit mit den von der bzw. dem Prüfenden zugelassenen Hilfsmitteln Probleme des Faches erkennen und mit geläufigen Methoden lösen kann. Klausuren dauern in der Regel mindestens 90 und höchstens 180 Minuten.

Mündliche Prüfung In den mündlichen Prüfungen soll die Kandidatin bzw. der Kandidat nachweisen, dass sie bzw. er die Zusammenhänge des Prüfungsgebietes erkennt und spezielle Fragestellungen in diese Zusammenhänge einzuordnen vermag. Durch die mündlichen Prüfungen soll ferner festgestellt werden, ob die Kandidatin bzw. der Kandidat über breites Grundlagenwissen verfügt. Mündliche Prüfungen dauern in der Regel mindestens 25 und höchstens 50 Minuten. Bei Gruppenprüfungen verlängert sich die Gesamtprüfungsdauer entsprechend der Kandidatenzahl.

Prüfung im Studium Generale In einer der im Studium Generale gewählten Veranstaltungen wird eine mündliche oder schriftliche Prüfungsleistung erbracht. Dabei handelt es sich in der Regel um eine Klausur (maximal vier Stunden), eine Hausarbeit (maximal 25 Seiten) oder eine mündliche Prüfung (maximal 45 Minuten).

Seminarvortrag (45-60 Minuten) und schriftliche Ausarbeitung In einem Seminarvortrag sollen die Studierenden nachweisen, dass sie zur wissenschaftlichen Ausarbeitung eines Themas in der Lage sind und die Ergebnisse vortragen können. In der schriftlichen Ausarbeitung sollen sie zeigen, dass sie in der Lage sind Ergebnisse gemäß wissenschaftlichen Standards darzustellen.

Studienleistung Als Studienleistung können Übungsaufgaben verlangt werden, die in der Regel wöchentlich als Hausaufgaben und/oder Präsenzaufgaben gestellt werden. Studienleistungen werden mit bestanden oder nicht bestanden bewertet. Bestanden Studienleistungen sind Voraussetzung zur Teilnahme an einer Modulabschlussprüfung.

Qualifizierte Teilnahme Die qualifizierte Teilnahme wird in der Regel durch eine Praktikumsarbeit mit anschließendem Gespräch nachgewiesen. Eine qualifizierte Teilnahme liegt dann vor, wenn die erbrachten Leistungen erkennen lassen, dass eine mehr als nur oberflächliche Beschäftigung mit den Gegenständen, die einer Aufgabenstellung zugrunde lagen, stattgefunden hat. Qualifizierte Teilnahmen sind zum Bestehen eines Moduls nachzuweisen.

Qualifizierte Teilnahme im Studium Generale In den Veranstaltungen des Studium Generale, bei denen keine Prüfung im Studium Generale abgelegt wird, ist die qualifizierte Teilnahme nachzuweisen.

1.4 Liste der nichtkognitive Kompetenzen

Dieser Studiengang baut die folgenden nichtkognitive Kompetenzen auf:

Einsatz und Engagement

- Gefühl der Verpflichtung informatorische Aufträge zu erfüllen
- Durchhaltevermögen bei der Bearbeitung informatischer Aufträge

Gruppenarbeit
Die Fähigkeit, effektiv und effizient in Gruppen bis zu mittlerer Größe (ca. 15 Personen) zu arbeiten.

Haltung und Einstellung

- Affinität gegenüber informatischen Problemen
- Bereitschaft sich informatorischen Herausforderungen zu stellen
- Sozial-kommunikative Fähigkeiten als bedeutsam beurteilen

Kooperationskompetenz

- Hilfs- und Kooperationsbereitschaft
- Sprachkompetenz
- Kommunikative Fähigkeiten
- Diskussionsbereitschaft gegenüber informatischen Themen
- Informatische Themen präsentieren können
- Fähigkeit und Bereitschaft informatisches Wissen weiterzugeben
- Fähigkeit und Bereitschaft zu konstruktiver Kritik
- Fähigkeit und Bereitschaft Absprachen zu treffen und einzuhalten
- Bereitschaft entlang der Absprachen zu handeln
- Bereitschaft fremde Ideen anzunehmen

Lernkompetenz

- Fähigkeit und Bereitschaft zu lebenslangem Lernen
- Fähigkeit und Bereitschaft zu problemorientiertem Lernen
- Fähigkeit und Bereitschaft kooperativem Lernen
- Fähigkeit zur Selbstorganisation von Lernprozessen und zu selbstständigem Lernen

Lernmotivation

- Bereitschaft informatische Fähigkeiten und informatorisches Wissen zu erweitern
- Bereitschaft infromatische Aufträge zu erfüllen

Medienkompetenz

- Nutzung problemorientierter Lern- und Entwicklungsumgebungen
- Nutzung von Werkzeugen zum wissenschaftlichen Schreiben
- Nutzung von Werkzeugen zum Präsentieren wissenschaftlicher Resultate

Motivationale und volitionale Fähigkeiten

- Offenheit neuen Ideen und Anforderungen gegenüber
- Bereitschaft neue und unvertraute Lösungswege anzuwenden
- Kritikfähigkeit gegenüber einem und reflektierten Umgang mit rezeptartigen Lösungswegen

Schreib- und Lesekompetenz (wissenschaftlich)

- Fähigkeit Quellen zu recherchieren und reflektiert zu beurteilen
- Fähigkeit informatische Sachverhalte sinnvoll zu strukturieren
- Fähigkeit eigene Ideen von anderen korrekt abzugrenzen (Vermeidung von Plagiaten)

Selbststeuerungskompetenz

- Verbindlichkeit
- Disziplin
- Termintreue
- Kompromissbereitschaft
- Übernahe von Verantwortung
- Geduld
- Selbstkontrolle
- Gewissenhaftigkeit
- Zielorientierung
- Motivation
- Aufmerksamkeit

Kapitel 2

Studienrichtungen

2.1 Erster Studienabschnitt

Studienrichtung	Erster Studienabschnitt
Koordination	Prof. Dr. rer. nat. Johannes Blömer Codes und Kryptographie Informatik
Enthaltene Module	• Analysis für Informatiker (S. 20) • Berechenbarkeit und Komplexität (S. 28) • Datenbanksysteme (S. 40) • Datenstrukturen und Algorithmen (S. 43) • Digitaltechnik (S. 46) • Gestaltung von Nutzungsschnittstellen (S. 53) • IT Sicherheit (S. 63) • Lineare Algebra für Informatiker (S. 67) • Modellierung (S. 74) • Programmiersprachen (S. 78) • Programmierung (S. 83) • Rechnerarchitektur (S. 86) • Schlüsselqualifikation (S. 90) • Software Engineering (S. 93) • Softwaretechnikpraktikum (S. 99) • Stochastik für Informatiker (S. 102) • Systemsoftware und systemnahe Programmierung (S. 107)

Beschreibung
Im ersten Studienabschnitt werden die wesentlichen Grundkenntnisse und Fähigkeiten vermittelt, die jeder Absolvent der Informatik kennen und beherrschen sollte. Der erste Studienabschnitt besteht nur aus Pflichtveranstaltungen, die mit einer Ausnahme in den ersten vier Fachsemestern liegen.

2.2 Informatikgebiet Algorithmen und Komplexität

Studienrichtung	Informatikgebiet Algorithmen und Komplexität
Koordination	Prof. Dr. rer. nat. Johannes Blömer Codes und Kryptographie Informatik
Enthaltene Module	• Einführung in Kryptographie (S. 48) • Grundlegende Algorithmen (S. 57) • Komplexitätstheorie (S. 65) • Parallelität und Kommunikation (S. 76) • Verteilte Algorithmen und Datenstrukturen (S. 110)
Beschreibung	
Algorithmen bilden die Grundlage jeder Hardware und Software: Ein Schaltkreis setzt einen Algorithmus in Hardware um, ein Programm macht einen Algorithmus für den Rechner verstehbar. Algorithmen spielen daher eine zentrale Rolle in der Informatik. Deshalb steht im Mittelpunkt des Bachelormoduls Algorithmen und Komplexität die Klassifizierung von Problemen bezüglich ihrer algorithmischen Komplexität. Als Maße für Komplexität werden insbesondere Laufzeit und Speicherbedarf, aber auch z.B. Parallelisierbarkeit herangezogen. Module dieses Gebiets behandeln sowohl die Entwicklung und Analyse effizienter Algorithmen und algorithmischer Techniken, als auch die Untersuchung der Problem-inhärenten Komplexität, d.h. den Nachweis unterer Komplexitätsschranken und den Komplexitätsvergleich von Problemen. Weiter ergänzt wird das Gebiet durch ein Modul zur Kryptographie. Hier wird die inhärente Schwierigkeit von Problemen, wie sie die Komplexitätstheorie nachzuweisen versucht, positiv etwa für den Entwurf sicherer Verschlüsselungsverfahren genutzt.	

2.3 Informatikgebiet Computer Systeme

Studienrichtung	Informatikgebiet Computer Systeme
Koordination	Prof. Dr. Marco Platzner Technische Informatik Informatik
Enthaltene Module	• Betriebssysteme (S. 30) • Eingebettete Systeme (S. 50) • Rechnernetze (S. 88) • Verteilte Systeme (S. 112)

Beschreibung

Jede Informatikanwendung benötigt ein technisches System, auf der sie ausgeführt wird. Beispiele für solche Systeme sind auf bestimmte Anwendungsklassen zugeschnittene Computersysteme wie eingebettete Systeme oder verteilte Systeme. Wesentliche Komponenten von Computersystemen sind neben der Rechner-Hardware auch die Betriebssysteme eines Rechners oder die technische Infrastruktur für die Vernetzung mehrerer Rechner.

Die Eignung und Qualität eines Gesamtsystems hängt wesentlich davon ab, dass die Eigenheiten, Vorteile und Einschränkungen dieser technischen Systeme sinnvoll ausgenutzt werden. Zudem entstehen aus der Evolution von Infrastrukturen neue Anwendungsklassen und tragen damit wesentlich zur Innovation in der Informatik und ihren Anwendungen bei. Wesentliche Beispiele sind hier zweifellos das Internet oder der Mobilfunk samt Smartphones, deren tief greifender Einfluss kaum überschätzt werden kann.

Das Verständnis solcher Systeme und die Fähigkeit, solche Systeme gezielt zu benutzen und weiterzuentwickeln, ist daher eine wesentliche Kernkompetenz eines Informatikers. In diesem Gebiet werden diese Fertigkeiten erworben und vertieft. Das Gebiet baut insbesondere auf den Modulen Digitaltechnik, Rechnerarchitektur sowie Systemsoftware und systemnahe Programmierung auf. Die dortigen Grundlagen werden hier methodisch vertieft, die zielgeleitete Konstruktion technischer Systeme wird untersucht und Verfahren zur Bewertung der Leistungsfähigkeit und Eignung solcher Systeme abgeleitet. Dabei fokussieren die einzelnen Module, die innerhalb dieses Gebiets gewählt werden können, auf einzelne Teilbereiche von Computersystemen: Vernetzung von Rechnern, verteilte Systeme, Betriebssysteme und eingebettete Systeme.

2.4 Informatikgebiet Daten und Wissen

Studienrichtung	Informatikgebiet Daten und Wissen
Koordination	Prof. Dr. Stefan Böttcher Electronic Commerce und Datenbanken Informatik
Enthaltene Module	• Computer Graphics Rendering (S. 32) • Data Mining (S. 35) • Databases and Information Systems (S. 37) • Grundlagen Wissensbasierter Systeme (S. 55)

Beschreibung

Intelligente Systeme sind Computersysteme, deren Verhalten durch Methoden und Algorithmen der Künstlichen Intelligenz (KI) gesteuert wird. Solche Systeme gewinnen kontinuierlich an Bedeutung, nicht nur auf wissenschaftlicher Ebene innerhalb der Informatik, sondern auch im sozialen und gesellschaftlichen Kontext: Autonome oder teilautonome Systeme wie Serviceroboter, selbstfahrende PKWs oder medizinische Diagnosesysteme werden unser privates und berufliches Leben in absehbarer Zukunft tiefgreifend verändern. Neben methodischen Fortschritten und einer Steigerung der Rechenleistung durch schnellere Hardware ist die rasante Entwicklung von KI-Systemen in der letzten Dekade vor allem einer Datenexplosion zu verdanken: Die Verfügbarkeit großer Mengen von Daten oder sensorisch erfasster Beobachtungen aus ihrer Umgebung versetzt intelligente Systeme in die Lage, ihr Verhalten durch Adaption und Lernen selbständige zu optimieren.

Dieses Gebiet kombiniert die Themen Daten und Wissen im Sinne einer modern ausgerichteten KI und vermittelt methodische Grundlagen intelligenter Systeme. Die Inhalte des Gebiets erstrecken sich von Aspekten der Wissensrepräsentation über die automatisierte Wissensverarbeitung bis zum Erwerb von Wissen aus Daten. Der effektive Umgang mit Daten durch systematische Organisation und Verarbeitung großer Datenbestände wird in Vorlesungen zum Thema Datenbanken und Informationssysteme erlernt. Ein weiteres Thema dieses Gebiets, die Computergraphik und grafische Datenverarbeitung, widmet sich einer speziellen Form von Daten, nämlich digitalen Bildern, abstrakten Beschreibungen großer 3D Szenen, sowie der visuellen Kommunikation zwischen Mensch und Maschine. Die wichtige Echtzeit-Verarbeitung oft extrem umfangreicher grafischer Datenmengen wird durch die Implementierung von Code auf GPUs (Graphics Processing Units) vermittelt.

2.5 Informatikgebiet Softwaretechnik

Studienrichtung	Informatikgebiet Softwaretechnik
Koordination	Prof. Dr. Heike Wehrheim Spezifikation und Modellierung von Softwaresystemen Informatik
Enthaltene Module	• Angriffssicherer Softwareentwurf (S. 23) • Interaktionsgestaltung (S. 60) • Logik und Deduktion (S. 70) • Modellbasierte Softwareentwicklung (S. 72) • Programmiersprachen und Übersetzer (S. 80) • Softwaremodellierung mit Formalen Methoden (S. 96)
Beschreibung	
Das Gebiet vermittelt einen breiten Überblick über die wichtigsten Konzepte, Notationen und Methoden der Softwaretechnik und ihrer formalen und mathematischen Grundlagen. Die vermittelten Kenntnisse sollen die Studierenden in die Lage versetzen, Softwaresysteme unter vorgegebenen technischen und ökonomischen Randbedingungen zu entwickeln. Darüber hinaus sollen die Studierenden das wissenschaftliche Handwerkszeug beherrschen, um sich im späteren Berufsleben in zukünftige Techniken einzuarbeiten.	

2.6 Informatikgebiet Vertiefung

Studienrichtung	Informatikgebiet Vertiefung
Koordination	Prof. Dr. Gerd Szwillus Mensch-Computer Interaktion Informatik
Enthaltene Module	• Angriffssicherer Softwareentwurf (S. 23) • Betriebssysteme (S. 30) • Computer Graphics Rendering (S. 32) • Data Mining (S. 35) • Databases and Information Systems (S. 37) • Einführung in Kryptographie (S. 48) • Eingebettete Systeme (S. 50) • Grundlagen Wissensbasierter Systeme (S. 55) • Grundlegende Algorithmen (S. 57) • Interaktionsgestaltung (S. 60) • Komplexitätstheorie (S. 65) • Logik und Deduktion (S. 70) • Modellbasierte Softwareentwicklung (S. 72) • Parallelität und Kommunikation (S. 76) • Programmiersprachen und Übersetzer (S. 80) • Rechnernetze (S. 88) • Softwaremodellierung mit Formalen Methoden (S. 96) • Verteilte Algorithmen und Datenstrukturen (S. 110) • Verteilte Systeme (S. 112)

Beschreibung
Dieses Gebiet erlaubt Studierenden eine Vertiefung in einem der vier Gebiete der Informatik (Algorithmen und Komplexität, Computersysteme, Daten und Wissen, Softwaretechnik).

Kapitel 3

Module

3.1 Pflichtmodul: Analysis für Informatiker

Modulname	Analysis für Informatiker / Calculus for Computer Science
Workload	240 h
Leistungspunkte	8 LP
Studiensemester	• Analysis für Informatiker : 1
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Analysis für Informatiker: Vorlesung (60h / 150h / DE / WS / 450) Analysis für Informatiker: Übung (30h / 0h / DE / WS / 40)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Analysis für Informatiker:	
Inhalte	
Analysis für Informatiker: Das Modul besteht aus der Vorlesung Analysis (für Informatiker) Gliederung „Analysis (V4+Ü2)“ Kapitel I Grundbegriffe 1. Mengen und Abbildungen 2. Vollständige Induktion und Rekursion, Kombinatorik 3. Elementare Zahlentheorie 4. Reelle Zahlen, Körper 5. Die komplexen Zahlen Kapitel II Analysis 1. Konvergenz von Folgen 2. Konvergenz von Reihen und Potenzreihen 3. Stetigkeit 4. Exponentialfunktion und trigonometrische Funktionen 5. Polarkoordinaten, Einheitswurzeln und der Fundamentalsatz der Algebra 6. Differenzierbarkeit 7. Lokale Extrema, Taylor-Formel, Taylor-Reihen 8. Integrierbarkeit (Riemann-Integral) 9. Approximation von Nullstellen und Fixpunkten. Das Newton-Verfahren	
Lernergebnisse / Fachkompetenzen	
Die Studierenden • beschreiben den progressiven Aufbau des Zahlensystems (bis hin zu den komplexen Zahlen) und argumentieren mit dem Permanenzprinzip als formaler Leitidee • verwenden die Begriffe der Konvergenz von Folgen und Reihen sowie der Vollstän-digkeit der reellen Zahlen formal sicher und erläutern diese Begriffe an tragenden Bei-spielen • beschreiben die Begriffe Stetigkeit und Differenzierbarkeit anschaulich und formal und begründen zentrale Aussagen über stetige und differenzierbare Funktionen, verwenden die Idee der Approximation durch Potenzreihen zur Beschreibung von Funktionen • definieren den Begriff des Integrals formal und verwenden ihn in mathematischen Zu-sammenhängen, interpretieren das Integrieren als Flächenmessung und als Mittelwertbildung, • erläutern und begründen den Hauptsatz der Differential- und Integralrechnung • nutzen Software zur Darstellung und Exploration mathematischer Modellierungen und als heuristisches Werkzeug zur Lösung von Anwendungsproblemen	

• kennen und reflektieren Fragen der Umsetzung numerischer Verfahren auf dem Computer (z.B. Komplexität, Genauigkeit)

Nichtkognitive Kompetenzen

• Lernkompetenz
• Selbststeuerungskompetenz

Methodische Umsetzung

Analysis für Informatiker: Die Studierenden
• präsentieren und erklären mathematische Sachverhalte
• denken konzeptionell, analytisch und logisch
• denken und handeln eigenständig
• erarbeiten sich interessengelenkt selbstständig mathematische Einsichten

Prüfungsleistung (Dauer)

Klausur (120 - 180 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 8 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Jürgen Klüners

Lernmaterialien, Literaturangaben

Analysis für Informatiker: keine

Sonstige Hinweise
keine

3.2 Wahlpflichtmodul: Angriffssicherer Softwareentwurf

Modulname	Angriffssicherer Softwareentwurf / Secure Software Engineering
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Angriffssicherer Softwareentwurf : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Angriffssicherer Softwareentwurf: Vorlesung (45h / 105h / EN / WS / 100) Angriffssicherer Softwareentwurf: Übung (30h / 0h / EN / WS / 100)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Angriffssicherer Softwareentwurf: Es wird vorausgesetzt, dass die Teilnehmer den ersten Studienabschnitt des Bachelors abgeschlossen oder nahezu abgeschlossen haben. Hierüberhinaus gibt es keine besonderen Voraussetzungen für diese Veranstaltung.	
Inhalte	
Angriffssicherer Softwareentwurf: Was braucht es, um Softwaresysteme angriffssicher zu entwickeln? Dies ist die Schlüsselfrage, der wir in dieser Veranstaltung auf den Grund gehen. Um sie zu beantworten ist es erforderlich, ein Verständnis der folgenden Kernbereiche des angriffssicheren Softwareentwurfs zu entwickeln: Bedrohungsmodellierung, sicheres Design, sichere Programmierung, Sicherheitsvalidierung, sicheres Deployment und sichere Wartung. Diese Bereiche werden in dieser Veranstaltung auf beispielorientierte Weise abgedeckt. Diskutiert werden aktuelle Techniken, die auf diese Bereiche anwendbar sind, sowie die Lektionen, die aus konkreten Sicherheitsvorfällen gelernt werden können.	
Lernergebnisse / Fachkompetenzen	
Ziel der Veranstaltung ist es, dass die Teilnehmer ein fundiertes Verständnis der allerwichtigsten Aspekte des angriffssicheren Softwareentwurfs erhalten. Das schließt die Fähigkeit ein, Bedrohungen von Softwaresystemen zu identifizieren und zu modellieren, um die gängigsten Klassen von Schwachstellen zu vermeiden, sowie Techniken und Werkzeuge zu identifizieren und anzuwenden, um das Einführen? von Sicherheitsschwachstellen zu verhindern oder zu identifizieren.	
Nichtkognitive Kompetenzen	

- Lernkompetenz
- Lernmotivation

Methodische Umsetzung

Angriffssicherer Softwareentwurf:

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

keine

Voraussetzungen für die Teilnahme an der Prüfung

keine

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Eric Bodden

Lernmaterialien, Literaturangaben

Angriffssicherer Softwareentwurf: Empfohlenes Buch: Gary McGraw: Software Security: Building Security In (2006, Addison-Wesley Professional) Über UPB als e-book verfügbar

Sonstige Hinweise

Inhalte: 1. Einführung in die Bedrohungsmodellierung und -analyse 2. Buffer Overflows: Prinzip, Exploits und Gegenmaßnahmen 3. Andere Code Injection-Schwachstellen: Prinzipien, Exploits und Gegenmaßnahmen 4. Crypto: gängige Algorithmen und Fallstricke 5. Zugriffskontrolle in Java und Android 6. Informationsfluss und Nutzungskontrolle 7. Automatische Erkennung von Schwachstellen: Codeanalyse, Fuzz Testing, modellbasiertes Testen 8. Systematische Sicherheitsanalyse 9. Softwareaktualisierung und -wartung

3.3 Pflichtmodul: Bachelor-Abschlussarbeit

Modulname	Bachelor-Abschlussarbeit / Bachelor Thesis
Workload	450 h
Leistungspunkte	15 LP
Studiensemester	• Bachelorarbeit Abschlussarbeit : 6 • Bachelorarbeit Arbeitsplan : 6

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Bachelorarbeit Abschlussarbeit: Kontaktzeiten, Ergebnispräsentation (15h / 345 h / DE / WS oder SS / 1)
Bachelorarbeit Arbeitsplan: Kontaktzeiten, Präsentation Arbeitsplan (15h / 75 h / DE / WS oder SS / 1)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Erfolgreicher Abschluss aller Module des erstem Studienabschnitts sowie des Moduls Schlüsselqualifikation.

Empfohlene Kenntnisse

Bachelorarbeit Abschlussarbeit: Je nach gewähltem Thema.

Bachelorarbeit Arbeitsplan: Je nach gewähltem Thema.

Inhalte

Bachelorarbeit Abschlussarbeit: Eine Bachelorarbeit umfasst die Bearbeitung eines Themas mit schriftlicher Ausarbeitung und einer mündlicher Präsentation der Ergebnisse. Der Studierende soll zeigen, dass er innerhalb einer vorgegebenen Frist ein Thema der Informatik auf der Grundlage wissenschaftlicher Methoden bearbeiten kann. Die Aufgabe einer Bachelorarbeit kann beispielsweise die Entwicklung von Software, Hardware, eine Beweisführung oder eine Literaturrecherche umfassen.

Bachelorarbeit Arbeitsplan: Nach Themenabsprache mit dem Betreuer erfolgt eine erste grobe Einarbeitung. Auf dieser Grundlage und einer ersten Literaturrecherche ist durch den Studierenden ein Arbeitsplan vorzulegen, der die zu erzielenden Ergebnisse samt Meilensteine für die Arbeit dokumentiert.

Lernergebnisse / Fachkompetenzen

Im Rahmen ihrer Abschlussarbeit bearbeiten die Studierenden ein Problem nach wissenschaftlichen Methoden innerhalb einer bestimmten Frist. Die im Zuge des Studiums erworbenen fachlich-methodischen sowie fachübergreifenden Kompetenzen sollen dazu entsprechend eingesetzt werden. Dazu gehören insbesondere auch die Strukturierung und Planung der einzelnen Arbeitsschritte sowie die Präsentation der Ergebnisse nach Abschluss der Arbeit.

Nichtkognitive Kompetenzen
• Einsatz und Engagement • Lernmotivation • Motivationale und volitionale Fähigkeiten • Schreib- und Lesekompetenz (wissenschaftlich) • Selbststeuerungskompetenz
Methodische Umsetzung
Bachelorarbeit Abschlussarbeit: Selbständiges Arbeiten unterstützt durch individuelle Betreuung Bachelorarbeit Arbeitsplan: Direkte Absprache mit Betreuer.
Prüfungsleistung (Dauer)
Abschlussarbeit Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
keine
Voraussetzungen für die Teilnahme an der Prüfung
keine
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 48 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. rer. nat. Johannes Blömer
Lernmaterialien, Literaturangaben
Bachelorarbeit Abschlussarbeit: Je nach gewähltem Thema. Bachelorarbeit Arbeitsplan: Je nach gewähltem Thema.
Sonstige Hinweise
Die Bachelorarbeit soll zeigen, dass die Kandidatin oder der Kandidat die Fähigkeit besitzt, innerhalb einer bestimmten Frist ein Problem der Informatik auf der Grundlage wissenschaftlicher Methoden zu bearbeiten. Die Aufgabenstellung soll so gestaltet werden, dass sie einem Arbeitsaufwand von neun

Wochen Vollzeitarbeit entspricht. Die Arbeit wird studienbegleitend erstellt und muss fünf Monate nach der Ausgabe abgegeben werden. Sie soll einen Umfang von in der Regel nicht mehr als 60 DIN A4-Seiten haben.

3.4 Pflichtmodul: Berechenbarkeit und Komplexität

Modulname	Berechenbarkeit und Komplexität / Computability and Complexity
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Berechenbarkeit und Komplexität : 3
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Berechenbarkeit und Komplexität: Vorlesung (45h / 105h / DE / WS / 200) Berechenbarkeit und Komplexität: Übung (30h / 0h / DE / WS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Erfolgreicher Abschluss der Module Modellierung und Datenstrukturen und Algorithmen	
Empfohlene Kenntnisse	
Berechenbarkeit und Komplexität: Inhalte der Vorlesungen Modellierung sowie Datenstrukturen und Algorithmen.	
Inhalte	
Berechenbarkeit und Komplexität: Einführung in grundlegende Methoden und Techniken zur Charakterisierung der Schwierigkeit von Berechnungsproblemen. Als formales Rechenmodel werden Turingmaschinen definiert. Ausgehend hiervon werden die wichtigsten Begriffe und Techniken der Berechenbarkeitstheorie (wie z.B. Entscheidbarkeit, Unentschuldbarkeit, Diagonalisierung, Reduktionen) und der Komplexitätstheorie (wie z.B. Zeitkomplexiät, Klassen P und NP, NP-Vollständigkeit, polynomielle Reduktionen) definiert und erläutert.	
Lernergebnisse / Fachkompetenzen	
Studierende kennen wesentliche Konzepte und Methoden der Berechenbarkeitstheorie und der Komplexitätstheorie. Sie können selbständig Probleme analysieren und klassifizieren. Studierende können Hypothesen zur Komplexität von Problemen entwickeln und diese anschließend verifizieren oder falsifizieren und darauf aufbauend neue Hypothesen formulieren.	
Nichtkognitive Kompetenzen	
• Haltung und Einstellung • Selbststeuerungskompetenz	
Methodische Umsetzung	
Berechenbarkeit und Komplexität: Die Vorlesung nutzt Tafelanschrieb und Folien sowie kleine Aufgaben für die Studierenden während der Vorlesung. Sie wird sowohl durch Tafelübung als auch durch Kleingruppentutorien begleitet. Studierende haben in den Kleingruppen Gelegenheit, Aufgaben in der	

Gruppe zu bearbeiten und Übungsblätter durch Tutoren benoten zu lassen.
Prüfungsleistung (Dauer)
Klausur (90 - 120 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 6 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. rer. nat. Johannes Blömer
Lernmaterialien, Literaturangaben
Berechenbarkeit und Komplexität: Michael Sipser, Introduction to the Theorey of Computation, Uwe Schöning, Theoretische Informatik - kurz gefasst; Foliensatz der VL; Übungsblätter.
Sonstige Hinweise
keine

3.5 Wahlpflichtmodul: Betriebssysteme

Modulname	Betriebssysteme / Operating Systems
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Betriebssysteme : 5
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Betriebssysteme: Vorlesung (45h / 105h / EN / WS / 30) Betriebssysteme: Übung (30h / 0h / EN / WS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Betriebssysteme: Vorlesung Systemsoftware und systemnahe Programmierung.	
Inhalte	
Betriebssysteme: Im Rahmen der Veranstaltung werden grundlegende Konzepte von Betriebssystemen besprochen, sowie spezifische Eigenschaften von Echtzeitbetriebssystemen und Betriebssystemen für eingebettete Systeme. Themen: • Parallelismus • Scheduling • Synchronisation • Inter-Process Communication • Memory Management • Security • Eingebettete Systeme • Echtzeitsysteme	
Lernergebnisse / Fachkompetenzen	
Lernziel ist das Verständnis fundamentaler Konzepte von Betriebssystemen. Die Studierenden verstehen diese Konzepte und sind in der Lage, diese an Beispielen anzuwenden.	

Nichtkognitive Kompetenzen
• Einsatz und Engagement • Lernkompetenz
Methodische Umsetzung
Betriebssysteme: Vorlesung mit praktischen Übungen
Prüfungsleistung (Dauer)
Klausur (90 - 120 Minuten) oder mündliche Prüfung (ca. 40 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Falko Dressler
Lernmaterialien, Literaturangaben
Betriebssysteme: Folien, Lehrbücher
Sonstige Hinweise
keine

3.6 Wahlpflichtmodul: Computer Graphics Rendering

Modulname	Computer Graphics Rendering / Computer Graphics Rendering
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Computer Graphics Rendering : 5

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Computer Graphics Rendering: Vorlesung (45h / 105h / DE / WS / 90)
Computer Graphics Rendering: Übung (30h / 0h / DE / WS / 20)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.

Empfohlene Kenntnisse

Computer Graphics Rendering: Grundkenntnisse in Linearer Algebra und Vektorrechnung, sowie ein sattelfeste Programmierausbildung, werden vorausgesetzt.

Inhalte

Computer Graphics Rendering: Computergrafik wird oft als übergeordneter Begriff verwendet, um die Erzeugung und Manipulation von digitalen Bildern zu beschreiben. Sie ist das Fachgebiet, welche visuelle Kommunikation durch Berechnung ermöglicht. In diesem Modul geht es konkret um die Generierung von digitalen Bildern und Bildsequenzen aus (mathematisch beschriebenen) 3D Szenen. Dieser Prozess wird Rendering genannt. Durch moderne Hardware und neue informatische Methoden unterstützt, wird Echtzeit-Rendering immer komplexerer 3D Szenen möglich. Um Studierende auf diesen Weg zu führen, werden folgende Themen bearbeitet:
• Geometrische Modellierung einer 3D Szene durch mathematische Beschreibungen, z.B. Punkte, Ebenen, Vektoren, Polyeder, oder gekrümmte Flächen.
• Die moderne Rendering Pipeline mit Transformationen (Translation, Skalierung, Rotation, Projektion), lokaler Reflektion und Schattierung, Sichtbarkeit, Rasterung, Texturen und Anti-aliasing.
• Fortgeschrittene Rendering Verfahren wie Scene Graph, Echtzeit-Schattenalgorithmen, Bildbasiertes Rendering (Image-Based Rendering), globale Reflexion, inkl. rekursives Raytracing, Radiosity, und andere Näherungen der Rendering Gleichung, Non-Photorealistic Rendering, oder Partikel Systeme.
Eine moderne Shader-basierte API wird die Vorstellung der Algorithmen begleiten und den Studierenden Erfahrungen mit GPU Architekturen ermöglichen.

Lernergebnisse / Fachkompetenzen

Studierende benennen und erklären relevante Verfahren entlang der Rendering Pipeline. Sie beherrschen entsprechende Rechentechniken um wichtige Verfahren (z.B. Transformation, Projektion, Cohen-Sutherland Clipping, Culling, Beleuchtungsmodelle, Gouraud-Schattierung, Rasterung von Linien und Kreisen, mathematische Faltung (convolution), B-Splines) auch in Rechenschritten nachzuvollziehen. Dasselbe (erklären, beherrschen Rechentechniken) gilt für alternative Verfahren zur Rendering Pipeline (z.B. Raytracing, Radiosity). Studierende demonstrieren die Fähigkeit, mit einem modernen API (z.B. OpenGL, WebGL) 3D Szenen nach bestimmten Vorgaben (Kamera, Beleuchtung, Modelle) mit unterschiedlichen Rendering Effekten (z.B. Schattenwurf, Bump Mapping, Environment Mapping) umzusetzen. Sie entwickeln auch Grafikprogrammen, welche die GPUs optimal ausnutzen. Studierende sind in der Lage, Rendering Software in Bezug auf Ihre Mächtigkeit an Rendering Funktionen zu bewerten.

Nichtkognitive Kompetenzen

- Einsatz und Engagement
- Lernmotivation
- Selbststeuerungskompetenz

Methodische Umsetzung

Computer Graphics Rendering: Die Vorlesung nutzt Beamer und Tafel wechselweise. Die Studierenden bearbeiten kurze In-Class Aufgaben und diskutieren dann mit der Dozentin über unterschiedliche Lösungen bzw. Probleme bei den Lösungen. In den Übungen werden in Kleingruppen Aufgaben (mathematische Aufgaben, Algorithmen, Progammieraufgaben) bearbeitet und Hausaufgaben vorbereitet und nachbesprochen.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Gitta Domik-Kienegger
Lernmaterialien, Literaturangaben
Computer Graphics Rendering: Vorlesungsfolien; Textbücher werden in der Vorlesung vorgestellt. Verteilung der Materialien über koaLA. Da die Computergrafik ein sehr dynamisches Fachgebiet ist, ändern sich die Materialien zeitlich entsprechend. Zur Zeit der Erstellung des Modulhandbuchs ist das Textbuch İnteractive Computer Graphics"von E. Angel und D. Schreiner, 6. Edition, Pearson Verlag, in Gebrauch.
Sonstige Hinweise
keine

3.7 Wahlpflichtmodul: Data Mining

Modulname	Data Mining / Data Mining
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Data Mining : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Data Mining: Vorlesung (45h / 105h / DE / SS / 120) Data Mining: Übung (30h / 0h / DE / SS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Data Mining: Grundlegende Kenntnisse in Mathematik (lineare Algebra, Statistik), Programmierung, Algorithmen und Datenstrukturen.	
Inhalte	
Data Mining: Die Vorlesung gibt eine Einführung in Verfahren zur Wissensentdeckung mithilfe von Methoden zur systematischen Suche nach Mustern in Daten. Der Schwerpunkt liegt auf effizienten algorithmischen Ansätzen für potenziell sehr große Datenbestände.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden verfügen über ein grundlegendes Verständnis der formalen statistischen und informationstheoretischen Grundlagen der Datenanalyse. Sie sind in der Lage, Datenanalyse- und Data Mining Probleme formal zu modellieren, Rohdaten eines speziellen Anwendungskontextes adäquat aufzubereiten, und geeignete Methoden auf die Daten anzuwenden. Darüber hinaus sind sie in der Lage, die Ergebnisse zu interpretieren und entsprechende Rückschlüsse zu ziehen. Insbesondere haben die Studierenden ein Bewusstsein für die Grenzen datenanalytischer Verfahren und die Gefahr von Fehlinterpretationen entwickelt.	
Nichtkognitive Kompetenzen	
• Lernkompetenz • Lernmotivation • Schreib- und Lesekompetenz (wissenschaftlich)	

Methodische Umsetzung
Data Mining: Theoretische Grundlagen und Konzepte des Data Mining werden im Rahmen einer Vorlesung eingeführt und anschließend in praktischen Übungen in Kleingruppen sowie in Heimübungen vertieft ergänzt.
Prüfungsleistung (Dauer)
Klausur (90 - 120 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Eyke Hüllermeier
Lernmaterialien, Literaturangaben
Data Mining: Begleitmaterial in Form eines Skripts, Übungszettel, zusätzliche Literatur in Form von Lehrbüchern.
Sonstige Hinweise
keine

3.8 Wahlpflichtmodul: Databases and Information Systems

Modulname	Databases and Information Systems / Databases and Information Systems
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Databases and Information Systems : beliebig

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Databases and Information Systems: Vorlesung (45h / 105h / EN / WS / 120)
Databases and Information Systems: Übung (30h / 0h / EN / WS / 30)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.

Empfohlene Kenntnisse

Databases and Information Systems: Studierende sollten Vorkenntnisse in relationalen Datenbanken und SQL haben, die in etwa denen der Vorlesung "Datenbanksystemeëntsprechen, sowie Vorkenntnisse im Programmieren, die in etwas denen der Vorlesungen "Programmierungünd "Grundlagen der Programmiersprachenëntsprechen.

Inhalte

Databases and Information Systems: Datenspeicherung und Datenmanagement spielen eine zentrale Rolle in Unternehmen, weil ein Großteil des Wissens von Unternehmen in Daten abgelegt ist. Zukünftige Anwendungsentwickler brauchen zur Verarbeitung großer Datenmengen Kenntnisse, die über traditionelle Datenbanken hinausgehen, insbesondere über NoSQL und Nicht-Standard-Datenmodelle, Hauptspeicher-Datenbanken, Kompression, Indizierung und effiziente Suche, um Anwendungen oder Informationssysteme zu entwickeln, die akzeptable Antwortzeiten haben. Dieses Modul behandelt schwerpunktmäßig Anfragen und Verarbeitung von Massendaten einschließlich baumstrukturierter Daten, Textdaten, Datenströmen, NoSQL-Daten und Hauptspeicher-Datenbanken. Das Modul umfasst Algorithmen, Technologien und praktische Fertigkeiten in diesen Gebieten.

Lernergebnisse / Fachkompetenzen

Nach Abschluss des Moduls sind die Studierenden in der Lage XML-Verarbeitung in Softwaresystemen zu verstehen, zu entwerfen, zu implementieren und in Bezug auf ihren Zeit- und Platz-Bedarf zu beurteilen. Sie kennen wesentliche Such- und Anfragetechniken zur Informationsbeschaffung in unkom-

primierten oder komprimierten XML-Datenbeständen. Sie sind in der Lage, unendliche Datenströme geeignet zu verarbeiten. Die Studenten sind in der Lage, sich neueste Forschungsergebnisse anhand von wissenschaftlichen Papers zu erarbeiten.

Nichtkognitive Kompetenzen

- Gruppenarbeit
- Lernkompetenz
- Lernmotivation

Methodische Umsetzung

Databases and Information Systems: Grundlegende Konzepte werden in einer Vorlesung präsentiert. Zusätzlich werden theoretische Konzepte in Tutorien in Kleingruppen vertieft, insbesondere für Kernkonzepte von Datenbanken wie die Suche ini und Anfragen auf Big Data, verteilten Datenbanken und mobilem Datenmanagement. Zudem erwerben Studierende praktische Kenntnisse durch Computergestützte Übungen, in denen sie aufbauend auf den in der Vorlesung erläuterten Konzepten ihre eigenen Informatonssysteme, Such- oder Kompressionsalgrithmen entwickeln.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Stefan Böttcher

Lernmaterialien, Literaturangaben

Databases and Information Systems: Verweise auf aktuelles Lehrnmaterial werden in der Vorlesung gegeben
Sonstige Hinweise
keine

3.9 Pflichtmodul: Datenbanksysteme

Modulname	Datenbanksysteme /
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Datenbanksysteme : 2

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Datenbanksysteme: Vorlesung (30h / 90h / DE / SS / 400)
Datenbanksysteme: Übung (30h / 0h / DE / SS / 40)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

keine

Empfohlene Kenntnisse

Datenbanksysteme: Kenntnisse in der Programmierung werden in dem Umfang vorausgesetzt, wie sie in den Veranstaltungen Programmierung und GPS gelehrt werden. Elementare Kenntnisse der Logik der Modellierung aus der Vorlesung Modellierung werden ebenfalls vorausgesetzt.

Inhalte

Datenbanksysteme: Datenbanken spielen eine zentrale Rolle in Unternehmen, weil ein Großteil des Wissen der Unternehmen als Daten in Datenbanken gespeichert wird. Für das Unternehmen ist es entscheidend, dass diese Daten korrekt, insbesondere konsistent, sind und dass sie effizient erfragt und aktualisiert werden können. Weiterhin sind die in Datenbanken abgelegten Datenbestände die wesentliche Datenquelle für eine Vielzahl von Anwendungsprogrammen, sie werden aber auch durch Anwendungsprogramme aktualisiert. Deshalb kommt der Organisation und Verarbeitung großer Datenbestände sowie der Einbindung von Datenbanken in Anwendungen eine zentrale Rolle bei der Erstellung korrekter und effizienter Anwendungen zu. Dieses Modul erschließt die Grundlagen für Datenbanksysteme, die in nahezu allen Unternehmen in der Praxis eingesetzt werden.

Lernergebnisse / Fachkompetenzen

Studierende lernen Faktenwissen
- Theorie und Konzepte relationaler Anfragesprachen kennen
- Konzepte des Datenbankentwurfs kennen
- Konzepte der Synchronisation und Recovery von Transaktionen kennen

Vermittlung von methodischem Wissen in Kleingruppen-Präsenz-Übungen:
- Komplexe Anfragen an relationale Datenbanken korrekt zu formulieren
- ein Datenbankschema möglichst redundanzfrei zu entwerfen

in praktischen Übungen am Rechner:
- eigene SQL-Anfragen an existierende relationale Datenbanken stellen

- Programme zu schreiben, die Datenbestände aus Datenbanken lesen oder verändern
- eigene Datenbanken zu definieren und aufzubauen

Vermittlung von Transferkompetenz
- die erworbenen Kompetenzen und Fertigkeiten auf andere Datenquellen oder andere Datenbanksysteme zu übertragen
- Umgang mit Zugriffsrechten

Vermittlung von normativ-bewertenden Kompetenzen
- die Eignung und Grenzen des relationalen Datenmodells bewerten und einzuschätzen
- den Programmieraufwand für Datenbankanfragen und Datenbankprogrammierung einzuschätzen
- die Folgen einer Datenbankschema-Änderung zu erkennen und abzuschätzen
- die Risiken eines schlecht entworfenen Datenbankschemas zu bewerten
- den Aufwand und Nutzen von Synchronisation und Recovery abzuschätzen

Nichtkognitive Kompetenzen

- Gruppenarbeit
- Lernkompetenz
- Lernmotivation

Methodische Umsetzung

Datenbanksysteme: Die Grundlagen und Konzepte von Datenbanksystemen werden im Rahmen einer Vorlesung eingeführt und anschließend in Präsenzübungen in Kleingruppen sowie in Heimübungen vertieft und durch praktische übungen ergänzt.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 5 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Stefan Böttcher
Lernmaterialien, Literaturangaben
Datenbanksysteme: Lehrbuch: Kemper, Eickler: Datenbanksysteme , Oldenbourg-Verlag, neueste Ausgabe. Lehrbuch: Garcia-Molina, Ullman, Widom: Database Systems: The Complete Book, Prentice Hall, neueste Ausgabe.
Sonstige Hinweise
keine

3.10 Pflichtmodul: Datenstrukturen und Algorithmen

Modulname	Datenstrukturen und Algorithmen / Data Structures and Algorithms
Workload	270 h
Leistungspunkte	9 LP
Studiensemester	• Datenstrukturen und Algorithmen : 2 • Praktikum: Datenstrukturen und Algorithmen : 2
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Datenstrukturen und Algorithmen: Vorlesung (60h / 135h / DE / SS / 400) Datenstrukturen und Algorithmen: Übung (30h / 0h / DE / SS / 25) Datenstrukturen und Algorithmen: Zentralübung (15h / 0 h / DE / SS / 400) Praktikum: Datenstrukturen und Algorithmen: (0h / 30h / DE / SS / 3)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Datenstrukturen und Algorithmen: Bereitschaft und Fähigkeit, den kreativen Prozess des Algorithmenentwurfs und die Effizienzanalyse u.a. mit mathematischen Methoden zu erlernen. Praktikum: Datenstrukturen und Algorithmen: Grundlagen der Programmierung.	
Inhalte	
Datenstrukturen und Algorithmen: Algorithmen bilden die Grundlage jeder Hardware und Software: Ein Schaltkreis setzt einen Algorithmus in Hardware um, ein Programm macht einen Algorithmus "für den Rechner verstehbar". Algorithmen spielen daher eine zentrale Rolle in der Informatik. Wesentliches Ziel des Algorithmenentwurfs ist die (Ressourcen-) Effizienz, d.h. die Entwicklung von Algorithmen, die ein gegebenes Problem möglichst schnell oder mit möglichst geringem Speicherbedarf lösen. Untrennbar verbunden mit effizienten Algorithmen sind effiziente Datenstrukturen, also Methoden, große Datenmengen im Rechner so zu organisieren, dass Anfragen wie Suchen Einfügen, Löschen aber auch komplexere Anfragen effizient beantwortet werden können. Die in dieser Veranstaltung vorgestellten Entwurfs- und Analysemethoden für effiziente Algorithmen und Datenstrukturen, sowie die grundlegenden Beispiele wie Sortierverfahren, dynamische Suchstrukturen und Graphenalgorithmen gehören zu den wissenschaftlichen Grundlagen für Algorithmenentwicklung und Programmierung in weiten Bereichen der Informatik. Praktikum: Datenstrukturen und Algorithmen: Begleitend zur Vorlesung Datenstrukturen und Algorithmen werden in diesem Programmierpraktikum einige wichtige Algorithmen und Datenstrukturen exemplarisch implementiert. Studierende werden in konkreten Projekten das Problem analysieren,	

geeignete Programmiertechniken auswählen, praktisch realisieren und eine quantitative Leistungsbewertung durchführen.

Lernergebnisse / Fachkompetenzen

Die Studierenden kennen effiziente Datenstrukturen und Algorithmen für ausgewählte grundlegende Probleme. Sie sind in der Lage Methoden zum Korrektheitsbeweis und zur Effizienzanalyse von Algorithmen und Datenstrukturen einzusetzen. Sie können selbstständig und kreative Algorithmen und Datenstrukturen entwickeln (Wie gestalte ich den kreativen Prozess vom algorithmischen Problem zum effizienten Algorithmus?). Sie sind in der Lage mathematischer Methoden zum Korrektheitsbeweis und zur Effizienzanalyse einzusetzen. Sie können die Wechselwirkung zwischen Algorithmus und Datenstruktur an wesentlichen Beispielen erläutern. Sie können die Qualität von Algorithmen und algorithmischen Ansätzen unter Effizienzaspekten einschätzen. Sie können sich neue Algorithmen, Datenstrukturen und algorithmischen Ideen und Analysen eneignen.

Nichtkognitive Kompetenzen

- Einsatz und Engagement
- Gruppenarbeit
- Haltung und Einstellung
- Lernkompetenz
- Motivationale und volitionale Fähigkeiten
- Selbststeuerungskompetenz

Methodische Umsetzung

Datenstrukturen und Algorithmen:
- Vorlesung mit Beamer und Tafelanschrieb
- Übungen in Kleingruppen
- erwartete Aktivitäten der Studierenden: aktive Mitarbeit bei Präsenzübungen, Hausaufgaben
- Übungsblätter, Musterlösungen werden in Zentralübungen vorgestellt
- In Übungen und Hausaufgaben werden Entwurf und Analyse von Algorithmen an ausgewählten Beispielen geübt.

Praktikum: Datenstrukturen und Algorithmen: Programmieraufgaben in kleineren Teams.

Prüfungsleistung (Dauer)

Klausur (120 - 180 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Qualifizierte Teilnahme: Praktikumsarbeit mit anschließendem Gespräch
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 9 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Friedhelm Meyer auf der Heide
Lernmaterialien, Literaturangaben
Datenstrukturen und Algorithmen: Standardlehrbücher, Foliensatz der Vorlesung, Übungsblätter Praktikum: Datenstrukturen und Algorithmen: Aufgabenstellung, man pages, eigenständige Recherche zu weiterführender Literatur.
Sonstige Hinweise
keine

3.11 Pflichtmodul: Digitaltechnik

Modulname	Digitaltechnik / Digital Design
Workload	150 h
Leistungspunkte	5 LP
Studiensemester	• Digitaltechnik : 2
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Digitaltechnik: Vorlesung (30h / 90h / DE / SS / 300) Digitaltechnik: Übung (30h / 0h / DE / SS / 25)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Digitaltechnik: Kenntnisse aus der Lehrveranstaltung Modellierung sind hilfreich.	
Inhalte	
Digitaltechnik: Die Vorlesung gibt eine Einführung in den Entwurf digitaler Schaltungen und Systeme. Dabei wird der Bogen vom Logikentwurf auf Gatterebene bis hin zu komplexeren Systemen auf Register-Transfer-Ebene gespannt. Die vermittelten Techniken und Methoden werden in den Übungen an Beispielen vertieft und mit modernen Entwurfswerkzeugen umgesetzt.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden sind nach dem Besuch der Lehrveranstaltung in der Lage, • den Entwurfsablauf in der Digitaltechnik von der Spezifikation bis zur technischen Realisierung zu beschreiben, • die zugrunde liegenden mathematischen Modelle aus der Booleschen Algebra und der Automatentheorie zu erklären und anzuwenden, • Entwürfe im Hinblick auf vorgegebene Entwurfsziele zu analysieren und zu bewerten, sowie • einfache Systeme selbständig zu konzipieren und mit den entsprechenden Entwurfswerkzeugen technisch zu realisieren.	
Nichtkognitive Kompetenzen	
• Gruppenarbeit • Lernkompetenz	
Methodische Umsetzung	
Digitaltechnik: • Vorlesung mit Beamer und Tafelanschrieb	

• Präsenzübungen in kleinen Gruppen mit Übungsblättern zu den theoretischen Grundlagen, Präsentation der Lösungen durch Übungsteilnehmer • Rechnerübungen zum Thema Hardware-Entwurf (Teamarbeit)
Prüfungsleistung (Dauer)
Klausur (60 - 90 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
keine
Voraussetzungen für die Teilnahme an der Prüfung
keine
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 5 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Marco Platzner
Lernmaterialien, Literaturangaben
Digitaltechnik: • Vorlesungsfolien und Übungsblätter • Aufgabenblätter und technische Dokumentation für die Rechnerübungen • J. F. Wakerly, “Digital Design,” 4th Edition, Upper Saddle River, NJ: Pearson, Prentice Hall, 2007 • Aktuelle Hinweise auf alternative und ergänzende Literatur, sowie Lehrmaterialien auf der Webseite und in den Vorlesungsfolien
Sonstige Hinweise
keine

3.12 Wahlpflichtmodul: Einführung in Kryptographie

Modulname	Einführung in Kryptographie / Introduction to Cryptography
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Einführung in Kryptographie : 6
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Einführung in Kryptographie: Vorlesung (45h / 105h / DE / SS / 100) Einführung in Kryptographie: Übung (30h / 0h / DE / SS / 25)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Einführung in Kryptographie: Datenstrukturen und Algorithmen sowie Berechenbarkeit und Komplexität	
Inhalte	
Einführung in Kryptographie: In dieser Vorlesung werden die wichtigsten Aufgaben und Methoden der modernen Kryptographie vorgestellt. Weiter werden einige der wichtigsten Sicherheitsanforderungen moderner Kryptographie informell diskutiert. Es werden die Vor- und Nachteile symmetrischer und asymmetrischer Kryptographie erläutert. Wichtige kryptographische Basiskonstruktionen wie Verschlüsselungsverfahren und digitale Signaturen werden vorgestellt.	
Lernergebnisse / Fachkompetenzen	
Studierende sind in der Lage Sicherheitsanforderungen von Anwendungen und Kryptosystemen präzise zu formulieren. Sie beherrschen die wichtigsten kryptographischen Basistechniken und können ihre Einsatzmöglichkeiten einschätzen und erläutern. Studierende können einschätzen, ob umgesetzte kryptographische Lösungen gegebenen Anforderungen genügen und sie können für gegebene Sicherheitsanforderungen die geeigneten kryptographischen Verfahren auszuwählen. Studierende können einschätzen, welche Anpassungen an kryptographische Verfahren unproblematisch sind und welche sicherheitskritisch sind.	
Nichtkognitive Kompetenzen	
• Lernkompetenz	

- Selbststeuerungskompetenz

Methodische Umsetzung

Einführung in Kryptographie: Eine Mischung aus Folien und Tafelanschrieb. Alle wichtigen Konzepte und Techniken werden in Übungen anhand von Beispielen weiter vertieft.

Prüfungsleistung (Dauer)

Mündliche Prüfung (ca. 40 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. rer. nat. Johannes Blömer

Lernmaterialien, Literaturangaben

Einführung in Kryptographie: Jonathan Katz, Yehuda Lindell, Introduction to Modern Cryptography, Chapman and Hall, Johannes Buchmann: Einführung in Kryptographie, Springer Verlag, Vorlesungsfolien, Übungsaufgaben

Sonstige Hinweise

keine

3.13 Wahlpflichtmodul: Eingebettete Systeme

Modulname	Eingebettete Systeme / Embedded Systems
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Eingebettete Systeme : 4
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Eingebettete Systeme: Vorlesung (45h / 105h / DE / SS / 50) Eingebettete Systeme: Übung (30h / 0h / DE / SS / 25)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Eingebettete Systeme: Kenntnisse aus der Lehrveranstaltung Rechnerarchitektur sind hilfreich.	
Inhalte	
Eingebettete Systeme: Die Veranstaltung bietet eine Einführung in Eingebettete Systeme und vermittelt Grundlagen zu Spezifikationsmodellen, eingebetteten Zielarchitekturen, grundlegenden Syntheseverfahren für Software und Hardware, sowie Methoden für die Bewertung und Analyse von Prozessor-Performance und -Energie.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden sind nach dem Besuch der Lehrveranstaltung in der Lage, • die Eigenschaften eingebetteter Systeme zu benennen, • die Entwurfsziele und Eigenschaften wesentlicher Typen von eingebetteten Zielarchitekturen zu erklären, • Methoden der Codegenierung und -optimierung für eingebettete Prozessoren anzugeben, • die Zusammenhänge an der Hardware/Software-Grenze zwischen Codegenerator und Prozessorarchitektur zu beschreiben, • grundlegende Verfahren von Software- und Hardware-Synthese zu erklären, • die Bedeutung von Performance- und Energie-Metriken einzuschätzen und • Methoden zur Bestimmung der Worst-Case-Execution-Time anzuwenden.	

Nichtkognitive Kompetenzen
• Gruppenarbeit • Lernkompetenz
Methodische Umsetzung
Eingebettete Systeme: • Vorlesung mit Beamer und Tafelanschrieb • Interaktive Übungen im Hörsaal • Rechnerübungen mit eingebetteten Zielarchitekturen (DSP, ARM, FPGA)
Prüfungsleistung (Dauer)
Mündliche Prüfung (ca. 40 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Marco Platzner
Lernmaterialien, Literaturangaben
Eingebettete Systeme: • Vorlesungsfolien und Übungsblätter • Aufgabenblätter und technische Dokumentation für die Rechnerübungen • Peter Marwedel: Embedded System Design, Springer, 2011. • Aktuelle Hinweise auf alternative und ergänzende Literatur, sowie Lehrmaterialien auf der Webseite und in den Vorlesungsfolien

Sonstige Hinweise
keine

3.14 Pflichtmodul: Gestaltung von Nutzungsschnittstellen

Modulname	Gestaltung von Nutzungsschnittstellen / Designing User Interfaces
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Gestaltung von Nutzungsschnittstellen : 4
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Gestaltung von Nutzungsschnittstellen: Vorlesung (45h / 105h / DE / SS / 300) Gestaltung von Nutzungsschnittstellen: Übung (30h / 0h / DE / SS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Gestaltung von Nutzungsschnittstellen:	
Inhalte	
Gestaltung von Nutzungsschnittstellen: Erlernbarkeit, Beeinträchtigungsfreiheit sowie die barrierefreie Erschließbarkeit von Softwaresystemen sind heute Pflichtanforderungen an die Systemgestaltung. Grundlegende Herausforderungen sind, mit Hilfe geeigneter Gestaltungsmaßnahmen Verständnisprozesse bei Nutzern zu fördern und unnötige Belastungen bei der Arbeit mit Softwaresystemen zu vermeiden. Dazu ist ein methodisches Repertoire erforderlich, um schon während des Entwurfs die Gebrauchstauglichkeit sichern zu können. Die dazu erforderlichen Kenntnisse und Fertigkeiten reichen von physiologischen und psychologischen Grundlagen über Methoden und Techniken der Systemgestaltung bis hin zu rechtlichen Anforderungen.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden sind in der Lage, grundlegende Problembereiche der Mensch-Rechner-Interaktion zu erkennen und sie konstruktiv gestaltend umzusetzen. Sie erwerben zugleich anschlussfähiges Wissen, das vor allem für die Zusammenarbeit mit Designern und Psychologen erforderlich ist, aber auch für den Diskurs mit Medienwissenschaftlern und Pädagogen (E-Learning) hilfreich ist. Die erworbenen Kenntnisse und Fertigkeiten bilden zugleich die Grundlage für vertiefende Veranstaltungen im Bereich der Mensch-Maschine-Wechselwirkung wie etwa Usability Engineering, Computergrafik oder auch Medien-Ergonomie, d.h. die Teilnehmer können Gebrauchstauglichkeit evaluieren, graphische Darstellungen erzeugen und manipulieren, sowie digitale Medien einordnen und bewerten.	
Nichtkognitive Kompetenzen	
• Haltung und Einstellung • Medienkompetenz	

Methodische Umsetzung
Gestaltung von Nutzungsschnittstellen: Eine Mischung aus Folien und Tafelanschrieb. Alle wichtigen Konzepte und Techniken werden in Übungen anhand von Beispielen weiter vertieft.
Prüfungsleistung (Dauer)
Klausur (90 - 120 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 6 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr.-Ing. Reinhard Keil
Lernmaterialien, Literaturangaben
Gestaltung von Nutzungsschnittstellen: Benyon, David (2010): Designing interactive systems. A comprehensive guide to HCI and interaction design. 2nd ed. Harlow: Addison Wesley. Leventhal, Laura M.; Barnes, Julie A. (2008): Usability engineering. Process, products, and examples. Upper Saddle River, NJ: Pearson/Prentice Hall. Nielsen, Jakob (1996): Usability engineering. 3. [print]. Boston: Acad. Press. Raskin, Jef (2009): The Humane Interface. New directions for designing interactive systems. 11. print. Boston, Mass.: Addison Wesley. Shneiderman, Ben; Plaisant, Catherine (2010): Designing the user interface. Strategies for effective human-computer interaction. 5. ed., internat. ed. Upper Saddle River, NJ: Addison-Wesley/Pearson. Tognazzini, Bruce (1992): Tog on interface. 2. printing. Reading, Mass.: Addison-Wesley.
Sonstige Hinweise
keine

3.15 Wahlpflichtmodul: Grundlagen Wissensbasierter Systeme

Modulname	Grundlagen Wissensbasierter Systeme / Foundations of Knowledge-Based Systems
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Grundlagen Wissensbasierter Systeme : beliebig

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Grundlagen Wissensbasierter Systeme: Vorlesung (45h / 105h / DE / WS / 120)
Grundlagen Wissensbasierter Systeme: Übung (30h / 0h / DE / WS / 30)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.

Empfohlene Kenntnisse

Grundlagen Wissensbasierter Systeme:

Inhalte

Grundlagen Wissensbasierter Systeme: Intelligente Systeme sind Computersysteme, deren Verhalten durch Methoden und Algorithmen der Künstlichen Intelligenz (KI) gesteuert wird. Solche Systeme gewinnen kontinuierlich an Bedeutung, nicht nur auf wissenschaftlichen Ebene sondern auch im sozialen und gesellschaftlichen Kontext: Autonome oder teilautonome Systeme wie Serviceroboter, selbstfahrende PKWs oder medizinische Diagnosesysteme werden unser privates und berufliches Leben in absehbarer Zukunft tiefgreifeng verändern. Diese Vorlesung gibt eine Einführung in Methoden und Konzepte der Künstlichen Intelligenz. Der inhaltliche Schwerpunkt liegt dabei auf Wissensbasierten Systemen im Sinne von Systemen, die mithilfe adäquater Ansätze zur Repräsentation und Verarbeitung von Wissen die Problemlösungskompetenz eines Fachexperten in einer bestimmten Anwendungsdomäne approximieren. Neben Methoden der Wissensrepräsentation und -verarbeitung gibt die Vorlesung auch einen ersten Einblick in den automatisierten Erwerb von Wissen mithilfe maschineller Lernverfahren.

Lernergebnisse / Fachkompetenzen

Die Studierenden verstehen den Unterschied zwischen klassischen Softwaresystemen und wissensbasierten Systemen bzw. klassischer Programmierung und dem Entwurf wissensbasierter Systeme. Sie sind mit der Architektur wissensbasierter Systeme sowie grundlegenden Methoden und Techniken zum Entwurf solcher Systeme vertraut und können sie auf konkrete Probleme anwenden. Die Studierenden

verstehen das Zusammenspiel von Wissen, Daten und Inferenz.

Nichtkognitive Kompetenzen

- Lernkompetenz
- Lernmotivation
- Schreib- und Lesekompetenz (wissenschaftlich)

Methodische Umsetzung

Grundlagen Wissensbasierter Systeme: Theoretische Grundlagen und Konzepte der Künstlichen Intelligenz werden im Rahmen einer Vorlesung eingeführt und anschließend in praktischen Übungen in Kleingruppen sowie in Heimübungen vertieft ergänzt.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Eyke Hüllermeier

Lernmaterialien, Literaturangaben

Grundlagen Wissensbasierter Systeme: Skript und weitere Literatur in Form von Lehrbüchern.

Sonstige Hinweise

keine

3.16 Wahlpflichtmodul: Grundlegende Algorithmen

Modulname	Grundlegende Algorithmen / Fundamental Algorithms
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Grundlegende Algorithmen : 5

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Grundlegende Algorithmen: Vorlesung (45h / 105h / DE / WS / 100)
Grundlegende Algorithmen: Übung (30h / 0h / DE / WS / 30)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.

Empfohlene Kenntnisse

Grundlegende Algorithmen: Bereitschaft und Fähigkeit, den kreativen Prozess des Algorithmenentwurfs und die Effizienzanalyse mit mathematischen Methoden zu erlernen. Grundkenntnisse einiger grundlegender Algorithmen und Datenstrukturen und deren Analysen werden vorausgesetzt.

Inhalte

Grundlegende Algorithmen: Dieser Kurs präsentiert Algorithmen und algorithmische Paradigmen für grundlegenden Probleme. Paradigmen wie Teile und Herrsche, dynamische Programmierung und Greedy-Algorithmen werden erläutert und durch Beispiele veranschaulicht. Ferner werden Graphenalgorithmen, Netzwerk-Fluss-Algorithmen und Hash-Verfahren vorgestellt. In allen Fällen werden Korrektheitsbeweise und Laufzeitanalysen angegeben.

Lernergebnisse / Fachkompetenzen

Die Studierenden wenden Entwurfsmethoden für effiziente Datenstrukturen und Algorithmen für schwierige Probleme wie Matching, Netzwerk-Fluß u.a. an. Sie nutzen mathematisch fundierte Methoden zum Korrektheitsbeweis und zur Effizienzanalyse von Algorithmen und Datenstrukturen.Zudem wählen sie zielgerichtet die zum gegebenen Problem passende generische Entwurfsmethode - z.B. Greedy-Algorithmen, Divide and Conquer, Dynamische Programmierung u.v.a.- aus. Darüber hinaus entwickeln sie selbstständig, kreativ Algorithmen und Datenstrukturen (Wie gestalte ich den kreativen Prozess vom algorithmischen Problem zum effizienten Algorithmus?) unter Nutzung von Entwurfsmethoden und ihrem Verständnis für die Struktur des algorithmischen Problems. Zudem nutzen sie einfache Varianten von fortgeschrittenen algorithmische Modellen wie online, approximative oder randomisierte Algorith-

men.

Nichtkognitive Kompetenzen

- Haltung und Einstellung
- Selbststeuerungskompetenz

Methodische Umsetzung

Grundlegende Algorithmen:
- Vorlesung mit Beamer und Tafelanschrieb.
- Übungen in Kleingruppen.
- erwartete Aktivitäten der Studierenden: aktive Mitarbeit bei Präsenzübungen, Hausaufgaben.
- Übungsblätter, Lösungen werden in Übungsgruppen vorgestellt und diskutiert.
- In Übungen und Hausaufgaben werden Entwurf und Analyse von Algorithmen an ausgewählten Beispielen geübt.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Friedhelm Meyer auf der Heide

Lernmaterialien, Literaturangaben

Grundlegende Algorithmen: Standardlehrbücher, Foliensatz der Vorlesung, Übungsblätter

Sonstige Hinweise
keine

3.17 Wahlpflichtmodul: Interaktionsgestaltung

Modulname	Interaktionsgestaltung / Interaction Design
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Interaktionsgestaltung : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Interaktionsgestaltung: Vorlesung (45h / 105h / DE / WS / 70) Interaktionsgestaltung: Übung (30h / 0h / DE / WS / 70)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Interaktionsgestaltung: Die Vorlesung baut auf der vorangegangenen Pflichtvorlesung Gestaltung von Nutzungsschnittstellen auf, in der bereits grundlegene Inhalte des Usability Engineering eingeführt werden.	
Inhalte	
Interaktionsgestaltung: Die Vorlesung behandelt die Gestaltung von Interaktion mit klassischen, graphischen Desktopsystemen genauso wie die Interaktionsgestaltung im Web. Dabei baut sie auf der Pflichtvorlesung ”Gestaltung von Nutzungsschnittstellenäuf, in der erste Grundkenntnisse des Usability Engineering vermittelt wurden. Hier werden die Inhalte vertieft und besonders intensiv die existierenden Usability-Regeln, sowie die darauf aufbauenden Verfahren behandelt und auch praktisch eingeübt. Es wird anschließend deutlich gemacht, was sich ändert, wenn wir statt klassischer graphischer Oberflächen die Situation bei der Gestaltung von Webauftritten betrachten. Die deutlich existierenden Gemeinsamkeiten beider Bereiche, aber auch ihre Abgrenzungen werden deutlich gemacht. Im Web tritt an die Stelle des Dialogdesigns das Navigationsdesign, was alternative Entwicklungsstrategien erfordert. Besondere Betonung wird in der Vorlesung auf die stets wachsende Nutzung mobiler Geräte mit ihrer andersartigen technischen Ausrüstung gelegt. Dadurch und durch den sehr anderen Nutzungskontext entstehen neue Benutzungsanforderungen, die durch neuartige Entwicklungsprozesse und -techniken abgefangen werden müssen. Um das Thema abzurunden, wird schließlich die bedeutende Rolle von Farbe und Typographie im Web thematisiert.	
Lernergebnisse / Fachkompetenzen	

Die Studierenden lernen die wesentlichen, aktuellen Verfahren kennen und anwenden, die im klassischen Usability Engineering, aber auch im Web eine Rolle spielen. Die Studierenden lernen, sich in der umfangreichen und komplexen Welt von Usability-Regeln zu orientieren und diese anzuwenden. Bezogen auf die Gestaltung von Interaktion im Web lernen die Hörer der Vorlesung relevante Aspekte der Webgestaltung zu trennen und adäquat zu untersuchen: Inhaltsstrukturen, visuelle Anordnung, Navigationsstrukturen und Auswahl von Typographie und Farbe.

Nichtkognitive Kompetenzen

- Einsatz und Engagement
- Gruppenarbeit
- Lernkompetenz
- Lernmotivation

Methodische Umsetzung

Interaktionsgestaltung: Die Vorlesung wird klassisch mit Beamerpräsentation gehalten und intensiv durch die E-Learning-Umgebung koaLA der Universität Paderborn unterstützt. Hier werden vor der Vorlesung die Folien veröffentlicht, schriftliche Übungsaufgaben gestellt, Software und Videoaufzeichnungen aller Vorlesungen zur Verfügung gestellt. Im Rahmen der Vorlesung selbst findet immer wieder an geeigneter Stelle interaktive Gruppenarbeit statt – etwa zum Durchführen von Usability-Tests oder Anwendung von Inspektionsmethoden. In den Übungsgruppen stellen Hörer die von ihnen erarbeiteten Lösungen vor und zur Diskussion. Dazu müssen die Vortragenden ihre Lösung anhand von geeigneten Vortragsfolien präsentieren.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Gerd Szwillus
Lernmaterialien, Literaturangaben
Interaktionsgestaltung: • Vorlesungsfolien als PDF zum Herunterladen • Schriftliche Hausaufgaben • Diverse, während der Vorlesung eingesetzte Softwarewerkzeuge • Ben Shneiderman: Designing the User Interface: Strategies for Effective Human-Computer Interaction, 2009 • David Benyon: Designing Interactive Systems: A Comprehensive Guide to HCI, UX and Interaction Design, 2009 • Jakob Nielsen und Raluca Budiu: Mobile Usability, 2012
Sonstige Hinweise
keine

3.18 Pflichtmodul: IT Sicherheit

Modulname	IT Sicherheit / IT Security
Workload	150 h
Leistungspunkte	5 LP
Studiensemester	• IT Sicherheit : 5

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

IT Sicherheit: Vorlesung (30h / 90h / DE / WS / 150)
IT Sicherheit: Übung (30h / 0h / DE / WS / 25)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Erfolgreicher Abschluss des Moduls Systemsoftware und systemnahe Programmierung

Empfohlene Kenntnisse

IT Sicherheit: Rechnernetze, Programmierung, Systemsoftware

Inhalte

IT Sicherheit: In der Vorlesung werden die wesentlichen Begriffe und Probleme der IT Sicherheit vorgestellt. Es werden klassische und moderne Angriffstechniken auf Netzwerkprotokolle, Passwort-Datenbanken, Computersysteme und Webanwendungen werden vorgestellt und geeigenete Gegenmaßnahmen diskutiert. Hierzu gehört auch die Vorstellung praxisrelevanter kryptographischer Protokolle und Algorithmen, sowie deren Sicherheitseigenschaften.

Lernergebnisse / Fachkompetenzen

Studierende verstehen die wesentlichen Konzepte, Methoden und Mechanismen zum Schutz von Daten und Systemen vor Manipulation und Missbrauch auf einem grundlegenden, praxis-orientierten, wissenschaftlichen Niveau. Sie sind in der Lage, die Konzepte zur Erhöhung der Systemsicherheit korrekt einzusetzen, einfache Sicherheitsprotokolle zu entwickeln und diese zu bewerten. Sie verstehen die Ursachen von Sicherheits-Problemen heutiger Systeme, sind in der Lage, grundlegende Konzepte auch in neuen Anwendungskontexten einzusetzen und besitzen ein generelles Bewusstsein für mögliche Sicherheitsbedrohungen und Risiken.

Nichtkognitive Kompetenzen

- Einsatz und Engagement
- Lernkompetenz
- Lernmotivation

Methodische Umsetzung

IT Sicherheit:

- Vorlesung mit Beamer und Tafelanschrieb
- Präsenzübungen in kleinen Gruppen mit Übungsblättern zu den theoretischen Grundlagen, Präsentation der Lösungen durch Übungsteilnehmer
- Praktische Übungen zur IT Sicherheit

Prüfungsleistung (Dauer)

Klausur (60 - 90 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 5 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr., Tibor Jager

Lernmaterialien, Literaturangaben

IT Sicherheit:
- Vorlesungsfolien und Übungsblätter
- Sicherheit und Kryptographie im Internet, Jörg Schwenk
- Computer Security, William Stallings und Lawrie Brown

Sonstige Hinweise

keine

3.19 Wahlpflichtmodul: Komplexitätstheorie

Modulname	Komplexitätstheorie / Complexity Theory
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Komplexitätstheorie : 5
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Komplexitätstheorie: Vorlesung (45h / 105h / DE / WS oder SS / 30) Komplexitätstheorie: Übung (30h / 0h / DE / WS oder SS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Komplexitätstheorie: Berechenbarkeit und Komplexität	
Inhalte	
Komplexitätstheorie: Die Komplexitätstheorie ist eine wichtige Ergänzung der Theorie der Algorithmen. Ihr Ziel ist es zu verstehen, warum gewisse Berechnungsprobleme schwierig sind und diese anhand ihrer Schwierigkeit zu klassifizieren. Das bekannteste und wichtigste Beispiel ist die Theorie der NP-Vollständigkeit.	
Lernergebnisse / Fachkompetenzen	
Studierende können die Eigenschaften wesentlicher Komlexitätsklassen wie P, NP und PSPACE benennen und erläutern. Studierende können Probleme in geeignete Komplexitätsklassen einordnen. Studierende beherrschen die wichtigsten Techniken, um Probleme gemäß ihrer Komplexität vergleichen zu können (Reduktionen). Sie können diese Techniken auf neue Probleme anwenden.	
Nichtkognitive Kompetenzen	
• Einsatz und Engagement • Lernmotivation • Selbststeuerungskompetenz	
Methodische Umsetzung	

Komplexitätstheorie: Eine Mischung aus Folien und Tafelanschrieb. Alle wichtigen Konzepte und Techniken werden in Übungen anhand von Beispielen weiter vertieft.
Prüfungsleistung (Dauer)
Mündliche Prüfung (ca. 40 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. rer. nat. Johannes Blömer
Lernmaterialien, Literaturangaben
Komplexitätstheorie: Michael Sipser, Introuction to the Theory of Computation, S. Arora, B. Barak, Computational Complexity - A Modern Approach, Cambridge University Press, Vorlesungsfolien, Übungsaufgaben
Sonstige Hinweise
keine

3.20 Pflichtmodul: Lineare Algebra für Informatiker

Modulname	Lineare Algebra für Informatiker / Linear Algebra for Computer Science
Workload	240 h
Leistungspunkte	8 LP
Studiensemester	• Lineare Algebra für Informatiker : 2
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Lineare Algebra für Informatiker: Vorlesung (60h / 150h / DE / SS / 300) Lineare Algebra für Informatiker: Übung (30h / 0h / DE / SS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Lineare Algebra für Informatiker: Analysis für Informatiker	

Inhalte

Lineare Algebra für Informatiker: Einführung in die Grundlagen der Linearen Algebra, die während des Informatikstudiums benötigt werden. Die Lineare Algebra thematisiert, auf unterschiedlichen begrifflichen Ebenen, praktisch und theoretisch das Lösen linearer Gleichungssysteme und darüber hinausgehend das Konzept der Linearität als universell einsetzbares mathematisches Lösungswerkzeug. Dessen Rolle für das weitere Studium liegt in der großen Bedeutung, welche die Linearisierung (oder Lineare Ap-proximation) für alle Sparten der Mathematik, für die mathematische Modellbildung und für die mathematischen Anwendungen hat.
Das Modul besteht aus der Vorlesung Lineare Algebra (für Informatiker) Gliederung Lineare Algebra (V4+Ü2)

1. Grundbegriffe
2. Vektorräume
3. lineare Abbildungen
4. Basis
5. Dimension
6. Matrizen
7. lineare Gleichungssysteme
8. Determinanten

9. Eigenwerte
10. charakteristisches Polynom
11. Normalformenproblem

Lernergebnisse / Fachkompetenzen

Die Studierenden
• verstehen und erläutern, wie abstrakte Vektorräume als koordinatenfreie Verallgemeinerung ein- bis dreidimensionaler Räume zustande kommen, und geben Beispiele aus der Mathematik und Anwendungsgebieten an, die in diesem konzeptionellen Rahmen verstanden werden können
• begreifen lineare Abbildungen von Vektorräumen als strukturverträgliche Abbildun-gen und erläutern, wie lineare Gleichungssysteme koordinatenfrei durch sie beschrieben werden
• verstehen den abstrakten Basis- und Dimensionsbegriff und erklären, wie dieser als Verallgemeinerung des naiven Koordinaten- und Dimensionsbegriff verstanden werden kann
• stellen lineare Abbildungen durch Matrizen dar und begreifen diese als koordinatenabhängige Realisierung
• verstehen und erläutern, wie sich die (eindeutige) Lösbarkeit solcher Gleichungssysteme charakterisieren lässt; lösen lineare Gleichungssysteme und erklären Lösungsverfahren
• verstehen die Determinante als alternierende Multilinearform und erläutern sie anhand ihrer geometrischen Bedeutung; begreifen ihre Rolle für die Inversion von Matrizen und kennen die Verfahren zu ihrer Bestimmung
• kennen den Begriff des Eigenwerts; verstehen und erklären das Normalformenproblem, kennen Kriterien für Diagonalisierbarkeit

Nichtkognitive Kompetenzen

• Lernkompetenz
• Selbststeuerungskompetenz

Methodische Umsetzung

Lineare Algebra für Informatiker: Die Studierenden
• reflektieren eigene Lernerfahrungen
• präsentieren und erklären mathematische Sachverhalte
• denken konzeptionell, analytisch und logisch
• erarbeiten sich interessengeleitet selbständig neue Erkenntnisse
• denken und handeln eigenständig

Prüfungsleistung (Dauer)

Klausur (120 - 180 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt

gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 8 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Jürgen Klüners
Lernmaterialien, Literaturangaben
Lineare Algebra für Informatiker: keine
Sonstige Hinweise
keine

3.21 Wahlpflichtmodul: Logik und Deduktion

Modulname	Logik und Deduktion / Logic and Deduction
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Logik und Deduktion : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Logik und Deduktion: Vorlesung (45h / 105h / DE / SS / 60) Logik und Deduktion: Übung (30h / 0h / DE / SS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Logik und Deduktion: Die Bachelorvorlesung Modellierung.	
Inhalte	
Logik und Deduktion: In dieser Vorlesung werden Grundlagen der mathematischen Logik vermittelt, die für die Informatik relevant sind. Die Studierenden lernen wesentliche Eigenschaften der klassische Aussagen- und Prädikatenlogik kennen. Darüber hinaus werden neuere Logiken, wie Modallogik und Fuzzy Logik, und deren Anwendungen in der Informatik eingeführt. Für diese logischen Systeme untersuchen wir Syntax, Semantik und verschiedene Beweiskalküle.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden lernen verschiedene logische Systeme kennen. Sie kennen wesentliche Eigenschafte dieser Systeme und sind in der Lage mit unterschiedlichen Beweiskalkülen zu arbeiten. Die Studierenden erwerben die Fähigkeit zur Modellierung mit Hilfe der Logik.	
Nichtkognitive Kompetenzen	
• Lernkompetenz • Lernmotivation	
Methodische Umsetzung	

Logik und Deduktion: In den Vorlesungen werden alle Themen behandelt. Während der Übungen werden die Studierenden ausführlich mit den neuen Themen arbeiten. Es gibt kein Computerpraktikum.
Prüfungsleistung (Dauer)
Klausur (90 - 120 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Dr. Lorijn van Rooijen
Lernmaterialien, Literaturangaben
Logik und Deduktion: Wird noch bekannt gegeben.
Sonstige Hinweise
keine

3.22 Wahlpflichtmodul: Modellbasierte Softwareentwicklung

Modulname	Modellbasierte Softwareentwicklung / Model-Based Software Development
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Modellbasierte Softwareentwicklung : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Modellbasierte Softwareentwicklung: Vorlesung (45h / 105h / EN / SS / 75) Modellbasierte Softwareentwicklung: Übung (30h / 0h / EN / SS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Modellierung, Programmierung und Software Engineering	
Empfohlene Kenntnisse	
Modellbasierte Softwareentwicklung: Modellierung, Programmierung und Software Engineering	
Inhalte	
Modellbasierte Softwareentwicklung: In der modellbasierten Softwareentwicklung steht das Modell einer Software im Mittelpunkt. Es wird dabei nicht nur zu Dokumentationszwecken, sondern auch zur Entwicklung selbst verwendet (auch modellgetriebene Softwareentwicklung genannt). Übliche modellbasierte Techniken beinhalten unter anderem den Entwurf von Modellierungssprachen anhand von statischer und dynamischer Semantik sowie Metamodellierung sowie die Anwendung der Modelle in Form von Modelltransformationen, oder auch zum Model Checking oder für das Reverse Engineering von Softwarearchitekturen. Den Trend zur modellbasierten und modellgetriebenen Softwareentwicklung kann man sowohl in der Forschung, als auch in der Praxis beobachten und stellt daher eine wichtige Grundlage für die Ausbildung eines Softwareentwicklers dar.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden sollen grundlegende Verfahren zur Konstruktion großer Softwaresysteme kennen und ihre Anwendung beherrschen. Sie sollen die Vor- und Nachteile von Spezifikationstechniken erfahren, die Notwendigkeit von Design erkennen und Modelle zur Verbesserung der Softwarequalität einsetzen können. Unter anderem wird auf das Paradigma des „Model Driven Development“ eingegangen, das einen wesentlichen Produktivitäts- und Qualitätsgewinn bei der Softwareentwicklung verspricht.	
Nichtkognitive Kompetenzen	
• Einsatz und Engagement • Lernkompetenz	

Methodische Umsetzung
Modellbasierte Softwareentwicklung: Vorlesung mit Beamer und praktische Rechnerübungen.
Prüfungsleistung (Dauer)
Mündliche Prüfung (ca. 40 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Jun.-Prof.Dr. Anthony Anjorin
Lernmaterialien, Literaturangaben
Modellbasierte Softwareentwicklung: • Völter, Stahl: Model-Driven Software Development: Technology, Engineering, Management (Wiley) • Ghezzi: Fundamentals of Software Engineering (Addison Wesley)
Sonstige Hinweise
keine

3.23 Pflichtmodul: Modellierung

Modulname	Modellierung / Modelling
Workload	240 h
Leistungspunkte	8 LP
Studiensemester	• Modellierung : 1
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Modellierung: Vorlesung (60h / 150h / DE / WS / 500) Modellierung: Übung (30h / 0h / DE / WS / 40)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Modellierung: keine, Veranstaltung im 1. Semester	
Inhalte	
Modellierung: Das Modellieren ist eine für das Fach Informatik typische Arbeitsmethode, die in allen Gebieten des Faches angewandt wird. Aufgaben, Probleme oder Strukturen werden untersucht und als Ganzes oder in Teilaspekten beschrieben, bevor sie durch den Entwurf von Software, Algorithmen, Daten und/oder Hardware gelöst bzw. implementiert werden. Mit der Modellierung eines Problems zeigt man, ob und wie es verstanden wurde. Damit ist sie Voraussetzung und Maßstab für die Lösung und sie liefert meist auch den Schlüssel für einen systematischen Entwurf. Als Ausdrucksmittel für die Modellierung steht ein breites Spektrum von Kalkülen und Notationen zur Verfügung. Sie sind spezifisch für unterschiedliche Arten von Problemen und Aufgaben. Deshalb werden in den verschiedenen Gebieten der Informatik unterschiedliche Modellierungsmethoden eingesetzt. In den entwurfsorientierten Gebieten (Softwaretech-nik, Hardware-Entwurf) ist die Bedeutung der Modellierung und die Vielfalt der Methoden natürlich besonders stark ausgeprägt.	
Lernergebnisse / Fachkompetenzen	
Studierende kennen wesentliche Techniken zur Modellierung informatischer Probleme. Sie können für ein gegebenes Problem eine geeignete Modellierungstechnik auswählen und das Problem mit dieses Technik beschreiben. Sie können grundlegende Techniken erweitern und verfeinern, um so neuartige Probleme zu modellieren.	
Nichtkognitive Kompetenzen	
• Lernkompetenz • Motivationale und volitionale Fähigkeiten	

Methodische Umsetzung
Modellierung: Die Vorlesung nutzt Tafelanschrieb und Folien sowie kleine Aufgaben für die Studierenden während der Vorlesung. Sie wird sowohl durch Tafelübung als auch durch Kleingruppentutorien begleitet. Studierende haben in den Kleingruppen Gelegenheit, Aufgaben in der Gruppe zu bearbeiten und Übungsblätter durch Tutoren benoten zu lassen.
Prüfungsleistung (Dauer)
Klausur (120 - 180 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 8 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. rer. nat. Johannes Blömer
Lernmaterialien, Literaturangaben
Modellierung: Uwe Kastens, Hans Kleine Büning, Modellierung; Angelika Steger, Diskrete Strukturen; Foliensatz der Vorlesung; Übungsblätter.
Sonstige Hinweise
keine

3.24 Wahlpflichtmodul: Parallelität und Kommunikation

Modulname	Parallelität und Kommunikation / Parallelism and Communication
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Parallelität und Kommunikation : 5

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Parallelität und Kommunikation: Vorlesung (45h / 105h / DE / WS / 60)
Parallelität und Kommunikation: Übung (30h / 0h / DE / WS / 30)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.

Empfohlene Kenntnisse

Parallelität und Kommunikation: Grundkenntnisse einiger grundlegender Algorithmen und Datenstrukturen und deren Analysen wird vorausgesetzt.

Inhalte

Parallelität und Kommunikation: Die Vorlesung beschäftigt sich mit effizienten Methoden, Kommunikation zwischen Mitglieder eines Netzwerks zu realisieren. Solche Netzwerke können z.B. LANs, WANs, Peer-to-Peer Systeme, das Internet oder Parallelrechner sein. In der Vorlesung stellen wir verteilte Algorithmen vor, Kommunikation durch Routing im Netzwerk, durch Simulation des Kommunikationsgraphen auf dem Netzwerk und mit Hilfe globaler Variablen zu realisieren. Zudem werden effiziente Methoden zur Verwaltung von globalem Speicher in Netzwerken vorgestellt. Diese Algorithmen werden bezüglich Korrektheit und Effizienz analysiert.

Lernergebnisse / Fachkompetenzen

Die Studierenden lernen die wichtigsten Techniken und Algorithmen im Bereich Netzwerkkommunikation kennen. Sie können entscheiden, in welcher Situation welcher Routing-Algorithmus geeignet ist. Sie können Routing-Algorithmen an neue Situationen anpassen.

Nichtkognitive Kompetenzen

- Haltung und Einstellung
- Selbststeuerungskompetenz

Methodische Umsetzung
Parallelität und Kommunikation: • Vorlesung mit Beamer und Tafelanschrieb • Übungen in Kleingruppen • erwartete Aktivitäten der Studierenden: Lösung von Übungsaufgaben, Mitarbeit in den Übungen
Prüfungsleistung (Dauer)
Klausur (90 - 120 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Friedhelm Meyer auf der Heide
Lernmaterialien, Literaturangaben
Parallelität und Kommunikation: Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes, Frank Thomson Leighton, M. Kaufmann Publishers, 1992, Skript, Foliensatz der Vorlesung, Übungsblätter
Sonstige Hinweise
keine

3.25 Pflichtmodul: Programmiersprachen

Modulname	Programmiersprachen / Programming Languages
Workload	120 h
Leistungspunkte	4 LP
Studiensemester	• Programmiersprachen : 1
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Programmiersprachen: Vorlesung (30h / 75h / DE / WS / 500) Programmiersprachen: Übung (15h / 0h / DE / WS / 40)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Programmiersprachen: Programmierung	
Inhalte	
Programmiersprachen: In Programmiersprachen werden Sprachkonstrukte, Spracheigenschaften und Programmierparadigmen im Vergleich und in Gegenüberstellung zu den im Modul Programmierung gelernten herausgearbeitet. Funktionale und logische Sprachkonstrukte und Programmierkonzepte werden auch praktisch an Beispielen in SML und Prolog erarbeitet.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden können Grundkonzepte von Programmier- und Anwendungssprachen erläutern. Sie können typische Eigenschaften nicht-imperativer Sprachen erklären. Sie sind in der Lage einfache Grammatiken, Typenspezifikationen und funktionale Programme zu entwickeln. Sie sind in der Lage praktische Erfahrungen in der Programmentwicklung auf neue Aufgaben zu übertragen. Sie besitzen die Fähigkeit neue Programmier- und Anwendungssprachen selbstständig zu erlernen.	
Nichtkognitive Kompetenzen	
• Gruppenarbeit • Lernkompetenz • Lernmotivation	
Methodische Umsetzung	
Programmiersprachen:	
Prüfungsleistung (Dauer)	
Klausur (60 - 90 Minuten)	

Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 4 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Stefan Böttcher
Lernmaterialien, Literaturangaben
Programmiersprachen: keine
Sonstige Hinweise
keine

3.26 Wahlpflichtmodul: Programmiersprachen und Übersetzer

Modulname	Programmiersprachen und Übersetzer / Programming Languages and Compilers
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Programmiersprachen und Übersetzer : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Programmiersprachen und Übersetzer: Vorlesung (45h / 105h / EN / WS / 120) Programmiersprachen und Übersetzer: Übung (30h / 0h / EN / WS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Programmiersprachen und Übersetzer: Aus der Vorlesung “Modellierung”: Endliche Automaten, kontext-freie Grammatiken. Aus der Vorlesung “Grundlagen der Programmiersprachen”: Spracheigenschaften, kontext-freie Grammatiken, Gültigkeitsbereiche, Datentypen.	
Inhalte	
Programmiersprachen und Übersetzer: Sprachen spielen in der Softwaretechnik vielfältige und wichtige Rollen: Als Programmiersprachen sind sie Ausdrucksmittel für die Programmentwicklung. Als Spezifikationssprachen dienen sie zur Formulierung von Aufgabenbeschreibungen im allgemeinen oder sind für bestimmte Anwendungsgebiete speziell zugeschnitten. Der Entwurf und die Implementierung solcher Sprachen durch Übersetzer und deren Herstellung durch Generatoren sind die zentralen Themen dieser Veranstaltung.	
Lernergebnisse / Fachkompetenzen	
Die Teilnehmer sind in der Lage • grundlegende Kalküle zur präzisen Beschreibung von Spracheigenschaften anzuwenden, • grundlegende Methoden zur Implementierung von Sprachen einzusetzen. • generierende Werkzeuge zur Sprachimplementierung auszuwählen und zu benutzen.	

Nichtkognitive Kompetenzen

- Gruppenarbeit
- Lernmotivation

Methodische Umsetzung

Programmiersprachen und Übersetzer:
- Vorlesung
- Diskussion
- Lesen
- "Check your Knowledge", Überprüfung des Lernstands
- Übungen
- Rechner-Übungen
- Sprachimplementierungs-Projekt "SetLan"
- Hausaufgaben

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Dr. Peter Pfahler

Lernmaterialien, Literaturangaben

Programmiersprachen und Übersetzer:

- Vorlesungsfolien
- Elektronischer Seminarapparat (koaLA)
- Ebooks
- Handbücher zu den Übersetzer-Werkzeugen

Sonstige Hinweise

keine

3.27 Pflichtmodul: Programmierung

Modulname	Programmierung / Programming
Workload	240 h
Leistungspunkte	8 LP
Studiensemester	• Programmierung : 1

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Programmierung: Vorlesung (60h / 150h / DE / WS / 500)
Programmierung: Übung (30h / 0h / DE / WS / 40)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

keine

Empfohlene Kenntnisse

Programmierung: Es sind keine Vorkenntnise erforderlich

Inhalte

Programmierung: Softwareentwicklung is ein zentrales Arbeitsgebiet der Informatik. Software-Entwickler müssen Aufgaben analysieren und modellieren, Software-Strukturen entwerfen und diese in einer Programmiersprache implementieren können. Dieser Modul vermittelt einführende und wissenschaftlich fundierte Kenntnisse und Fähigkeiten in der Programmierung. Zusammen mit den Modulen Modellierung, Datenbanksysteme, und Softwaretechnik werden damit die wissenschaftlichen Grundlagen für das Arbeitsgebiet Software-Entwicklung gelegt und praktisch eingeübt. Dieses Modul soll die Teilnehmer befähigen,
• eine für die Software-Entwicklung relevante Programmiersprache anzuwenden (zur Zeit Python, in geringerem Umfang auch Java)
• Grundbegriffe der objektorientierten Programmiermethodik einzusetzen,
• Algorithmen in Programmen zu implementieren.
Im Informatikstudium bildet dieses Modul zusammen mit den Pflichtmodulen Modellierung, Datenbanksysteme und Softwaretechnik den Kern der Grundausbildung in Gebiet Softwaretechnik.

Lernergebnisse / Fachkompetenzen

Die Studierenden lernen ...
Faktenwissen, unter anderem - die wesentlichen Konstrukte einer Programmiersprache (derzeit Python, in geringem Umfang auch Java), - die Grundkonzepte von Komposition und Abstraktion in der Programmierung zu verstehen
methodisches Wissen - die gelernten Sprachkonstrukte sinnvoll und mit Verständnis anzuwenden, - Software zu testen sowie Fehlerursachen zu finden und zu beseitigen, - objektorientierte Grundkonzepte zu verstehen und anzuwenden, - Software aus objektorientierten Bibliotheken wiederzuverwenden

Transferkompetenz - praktische Erfahrungen in der Programmentwicklung auf neue Aufgaben zu übertragen
normativ-bewertenden Kompetenzen - den Aufwand und die Durchführbarkeit von Programmieraufgabe zu beurteilen

Nichtkognitive Kompetenzen

- Gruppenarbeit
- Kooperationskompetenz
- Lernmotivation

Methodische Umsetzung

Programmierung: Sprachkonstrukte und Programmiertechniken werden an typischen Beispielen eingeführt und erläutert und anschließend in den Übungen praktisch erprobt. Objektorientierte Methoden und Abstraktion werden überwiegend an der Benutzung von Bibliotheken erklärt. In Übungenstunden in Kleingruppen werden praktische Programmieraufgaben unter Anleitung an Rechnern bearbeitet.

Prüfungsleistung (Dauer)

Klausur (120 - 180 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 8 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Stefan Böttcher

Lernmaterialien, Literaturangaben

Programmierung: keine

Sonstige Hinweise
keine

3.28 Pflichtmodul: Rechnerarchitektur

Modulname	Rechnerarchitektur / Computer Architecture
Workload	150 h
Leistungspunkte	5 LP
Studiensemester	• Rechnerarchitektur : 3

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Rechnerarchitektur: Vorlesung (30h / 90h / DE / WS / 300)
Rechnerarchitektur: Übung (30h / 0h / DE / WS / 25)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

keine

Empfohlene Kenntnisse

Rechnerarchitektur: Kenntnisse aus der Lehrveranstaltung Digitaltechnik sind hilfreich.

Inhalte

Rechnerarchitektur: Die Vorlesung gibt eine Einführung in den Aufbau und Entwurf moderner Rechensysteme. Insbesondere wird vermittelt, wie durch ein effizientes Zusammenspiel von Hardware und Software kostengünstige und leistungsstarke Rechner entwickelt werden können. Die vorgestellten Techniken und Methoden werden in den Übungen an Beispielen vertieft.

Lernergebnisse / Fachkompetenzen

Die Studierenden sind nach dem Besuch der Lehrveranstaltung in der Lage,
• den Aufbau eines modernen Rechners sowie das Zusammenspiel von Hardware und Software zu beschreiben,
• die zugrunde liegenden allgemeinen Entwurfsprinzipien und -strategien zu erklären und anzuwenden,
• Rechnersysteme im Hinblick auf Leistung und Kosten zu analysieren und zu bewerten, sowie
• selbständig einfache Assemblerprogramme zu schreiben.

Nichtkognitive Kompetenzen

• Gruppenarbeit
• Lernkompetenz

Methodische Umsetzung

Rechnerarchitektur:
• Vorlesung mit Beamer und Tafelanschrieb
• Präsenzübungen in kleinen Gruppen mit Übungsblättern zu den theoretischen Grundlagen, Präsentation der Lösungen durch Übungsteilnehmer

• Rechnerübungen zur Assemblerprogrammierung
Prüfungsleistung (Dauer)
Klausur (60 - 90 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
keine
Voraussetzungen für die Teilnahme an der Prüfung
keine
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 5 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Marco Platzner
Lernmaterialien, Literaturangaben
Rechnerarchitektur: • Vorlesungsfolien und Übungsblätter • D. A. Patterson, J. L. Hennessy: Computer Organization & Design –The Hardware / Software Interface (3rd Edition); Morgan Kaufmann, 2007; ISBN: 978-0-12-370606-5, ISBN-10: 0-12-370606-8 • Aktuelle Hinweise auf ergänzende Literatur und Lehrmaterialien auf der Webseite und in den Vorlesungsfolien
Sonstige Hinweise
keine

3.29 Wahlpflichtmodul: Rechnernetze

Modulname	Rechnernetze / Computer Networks
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Rechnernetze : 5

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Rechnernetze: Vorlesung (45h / 105h / DE / WS / 100)
Rechnernetze: Übung (30h / 0h / DE / WS / 30)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.

Empfohlene Kenntnisse

Rechnernetze: Vorlesung Systemsoftware und systemnahe Programmierung oder vergleichbar.

Inhalte

Rechnernetze: Die Vorlesung Rechnernetze behandelt konzeptionelle und technologische Grundlagen von Rechnernetzen/Internet; thematisch werden dabei die Ebenen 1–4 des ISO/OSI-Modells abgedeckt. Zusätzlich werden Ansätze und Werkzeuge zur quantitativen Untersuchung von Kommunikationsprotokollen behandelt. Die Vorlesung wird durch eine Tafelübung begleitet.
• Die Veranstaltung lässt sich sehr gut mit der Veranstaltung Verteilte Systeme ergänzen.
• In einigen Semestern (wenn sowohl Rechnernetze als auch Verteilte Systeme angeboten werden) findet die Veranstaltung halbsemestrig statt; in der zweiten Semesterhälfte die Veranstaltung Verteilte Systeme.

Lernergebnisse / Fachkompetenzen

Absolventen der Lehrveranstaltung
• können die wesentlichen Aufgaben bei Konstruktion und Bau eines Rechnernetzes benennen und wesentliche Architekturansätze beschreiben;
• können unterschiedliche Lösungen für ein Problem aufzählen, deren Vor- und Nachteile herausfinden und sich, gemäß der Anforderungen, für eine Lösung entscheiden;
• Schwachstellen existierender Lösungen identifizieren und neue Kommunikationsprotokolle entwickeln und deren Leistungsfähigkeit bewerten.
ibung der zugehörigen Lehrveranstaltung

Nichtkognitive Kompetenzen
• Einsatz und Engagement • Lernkompetenz
Methodische Umsetzung
Rechnernetze: Folienbasierte Vorlesung mit Tafelanschrieb, durch Übung begleitet. Übungen dabei sowohl konzeptionell/analytisch als auch mit praktischen Aufgaben.
Prüfungsleistung (Dauer)
Klausur (90 - 120 Minuten) Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Studienleistung: schriftliche Übungsaufgaben Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 12 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. rer. nat. Holger Karl
Lernmaterialien, Literaturangaben
Rechnernetze: Folien, Standardlehrbücher (insbes. Tanenbaum, Rechnernetze), Übungsblätter.
Sonstige Hinweise
keine

3.30 Pflichtmodul: Schlüsselqualifikation

Modulname	Schlüsselqualifikation
Workload	150 h
Leistungspunkte	5 LP
Studiensemester	• Mentoring : beliebig • Proseminar : beliebig

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Mentoring: Treffen in Kleingruppen (15h / 15 h / DE / WS oder SS / 20)
Proseminar: Präsenzzeit Seminar (15h / 105 h / DE / WS oder SS / 15)

Wahlmöglichkeiten innerhalb des Moduls

Für das Proseminar können alle Proseminare aus dem Angebot des Bachelorstudiengangs Informatik gewählt werden.

Teilnahmevoraussetzungen

keine

Empfohlene Kenntnisse

Mentoring: keine

Proseminar: Je nach gewähltem Thema.

Inhalte

Mentoring: Im Mentoring werden Studierende einzelnen Lehrenden und deren Mitarbeiterinnen bzw. Mitarbeitern in Mentorengruppen (15 – 20 Studierende) zugeordnet. Es finden während des gesamten Bachelorstudiums je nach Bedarf etwa zweimal im Semester Treffen statt. Ziel ist es, durch Beratung – individuell oder in Kleingruppen – Probleme des Studiums und des Fa-ches zu bearbeiten. Dabei sollen Engagement, Motivation und Selbstständigkeit als Aspekte von Selbstkompetenz gestärkt werden. Das Mentoring zielt auf Vermeidung unnötig langer Studiendauern und auf Reduktion der Abbrecherquote.

Proseminar: Im Proseminar soll beispielhaft die Einarbeitung in ein wissenschaftliches Thema erlernt und abstraktes Denken gestärkt werden. Die Inhalte sollen schriftlich und mündlich präsentiert werden. Dazu soll Basiswissen in Bezug auf Literaturrecherche, Rhetorik und aktuelle Präsentationstechniken sowie in Bezug auf Kritikfähigkeit und Feedbackmethoden erworben und angewendet werden.

Lernergebnisse / Fachkompetenzen

Im Proseminar werden wesentliche Techniken des Erwerbs und der Präsentation wissenschaftlicher Ergebnisse und Erkenntnisse vermittelt. Im Mentoring erhalten Studierenden wichtige Kenntnisse zur erfolgreichen Organisation des Studiums.

Nichtkognitive Kompetenzen

- Einsatz und Engagement
- Haltung und Einstellung
- Lernkompetenz
- Lernmotivation
- Medienkompetenz
- Motivationale und volitionale Fähigkeiten
- Schreib- und Lesekompetenz (wissenschaftlich)
- Selbststeuerungskompetenz

Methodische Umsetzung

Mentoring: Es finden während des gesamten Bachelorstudiums je nach Bedarf etwa zweimal im Semester Treffen statt, in Kleingruppen oder individuell.

Proseminar: Referate mit schriftlicher Ausarbeitung und Vortrag.

Prüfungsleistung (Dauer)

Seminarvortrag (45-60 Minuten) und schriftliche Ausarbeitung
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

keine

Voraussetzungen für die Teilnahme an der Prüfung

keine

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 10 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. rer. nat. Johannes Blömer

Lernmaterialien, Literaturangaben

Mentoring: keine

Proseminar: Je nach gewähltem Thema.

Sonstige Hinweise
keine

3.31 Pflichtmodul: Software Engineering

Modulname	Software Engineering / Software Engineering
Workload	150 h
Leistungspunkte	5 LP
Studiensemester	• Praktikum: Software Engineering : 2 • Software Engineering : 2
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Praktikum: Software Engineering: (0h / 30h / DE / SS / 3) Software Engineering: Vorlesung (30h / 75h / DE / WS / 300) Software Engineering: Übung (15h / 0h / DE / WS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Praktikum: Software Engineering: Programmierung, Modellierung Software Engineering: Programmierung, Modellierung	
Inhalte	
Praktikum: Software Engineering: Begleitend zur Vorlesung Software Engineering werden an einem durchgängigen Beispiel ausgehend von der Spezifikation von Anforderungen sowohl die Modellierung von Softwaresystemen als auch der Übergang von den Modellen zur Implementierung sowie zum modellbasierten Testen der Softwaresysteme bearbeitet. Software Engineering: In der Vorlesung werden die Grundlagen der systematischen und ingenieursmäßigen Softwareentwicklung vermittelt. Im Fokus steht dabei die modellbasierte Softwareentwicklung. Die Vorlesung führt in wesentliche Vorgehensmodelle für die Softwareentwicklung ein, sowohl klassische als auch agile. Es werden Methoden für die Softwareentwicklung und -qualitätssicherung vermittelt, die innerhalb der Vorgehensmodelle zum Einsatz kommen. Außerdem werden Modellierungssprachen und Softwarewerkzeuge vorgestellt, mit denen die statischen und dynamischen Aspekte von Softwaresystemen beschrieben werden können. Insbesondere wird die objektorientierte Modellierungssprache UML (Unified Modeling Language) eingeführt, die unterschiedliche Diagrammsprachen wie Klassendiagramme, Komponentendiagramme, Use-Case-Diagramme, Aktivitätendiagramme, Sequenzdiagramme und Zustandsdiagramme vereint. Modellierungswerkzeuge werden exemplarisch eingesetzt. Die Vorlesung wird abgerundet durch eine durchgängige Entwicklungsmethode von der Anforderungsspezifikation über den Architektur- und Softwareentwurf bis hin zur Implementierung und dem Testen der Software. Hierbei wird vor allem auf die Aspekte der systematischen Ableitung und Verfeinerung von Modellen, der Transformation von Modellen in Programmcode (Codegenerierung) sowie des modellbasierten Testens eingegangen. Es werden methodische Hinweise zur Erstellung der Ergeb-	

nisartefakte (u.a. Richtlinien, Architekturstile und Entwurfsmuster) und zur Prüfung ihrer Qualität sowie zum Einsatz der Modellierungssprachen im Softwareentwicklungsprozess gegeben. Darüber hinaus werden Techniken zur Definition und domänenspezifischen Anpassung von Modellierungssprachen (Metamodellierung, UML-Profile sowie Beispiele konkreter domänenspezifischer Sprachen (DSLs) wie SysML oder BPMN) betrachtet. Die Vorlesung wird durch Übungen begleitet, in der die Vorlesungsinhalte aufgegriffen, vertieft und an beispielhaften Entwicklungsaufgaben selbst angewendet werden.

Lernergebnisse / Fachkompetenzen

Die Studierenden sollen in der Lage sein, für ein gegebenes Problem schrittweise eine Softwarelösung zu entwickeln. Hierzu sollen sie ein modellbasiertes Vorgehen einsetzen können, wobei sie für die einzelnen Entwicklungsschritte unterschiedliche Diagrammarten der UML (Unified Modeling Language) verwenden. Zur Überprüfung der Qualität der entwickelten Softwarelösung sollen sie in der Lage sein, Techniken des modellbasierten Testens einzusetzen. Sie verstehen die Beziehungen und Übergänge zwischen verschieden Entwicklungsphasen eines Vorgehensmodells. Sie beherrschen verschiedene Diagrammsprachen der UML zur Modellierung der unterschiedlichen Aspekte einer Softwarelösung und können die Qualität von Zwischenergebnissen bewerten. Außerdem haben sie ein grundlegendes Verständnis der Techniken zur Entwicklung und Spezialisierung von Modellierungssprachen für spezielle Situationen und Domänen. Durch den Einsatz des Gelernten am durchgängigen Beispiel der Praktikumsaufgabe verstehen die Studierenden die Bedeutung der verschiedenen Phasen einer Softwareentwicklung und sind in der Lage, diese durchgängig an einem konkreten Softwaresystem einzusetzen.

Nichtkognitive Kompetenzen

- Einsatz und Engagement
- Haltung und Einstellung
- Kooperationskompetenz
- Lernkompetenz
- Lernmotivation
- Selbststeuerungskompetenz

Methodische Umsetzung

Praktikum: Software Engineering: Softwareentwicklungsaufgaben (Modellieren, Implementieren, Testen) in kleinen Teams.

Software Engineering: In der Vorlesung werden die Grundlagen, Begrifflichkeiten, Sprachen und Methoden des Software Engineering vermittelt, die dann in den begleitenden Übungen vertieft und in dem begleitenden Praktikumsanteil von den Studierenden an einem durchgängigen Beispiel selbst erprobt werden.

Prüfungsleistung (Dauer)

Klausur (60 - 90 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben Qualifizierte Teilnahme: Praktikumsarbeit mit anschließendem Gespräch Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
Bestehen der Studienleistung.
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 5 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Gregor Engels
Lernmaterialien, Literaturangaben
Praktikum: Software Engineering: Aufgabenstellung, Material der begleitenden Vorlesung Software Engineering, eigenständige Recherche zu weiterführender Literatur (in der Bibliothek, im Internet) Software Engineering: Folien, Tafelanschrieb, evtl. Vorlesungsaufzeichnung, Übungen, Praktikumsaufgabe (siehe Praktikum: Software Engineering)
Sonstige Hinweise
keine

3.32 Wahlpflichtmodul: Softwaremodellierung mit Formalen Methoden

Modulname	Softwaremodellierung mit Formalen Methoden / Softwaremodelling with Formal Methods
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Softwaremodellierung mit formalen Methoden : 6
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Softwaremodellierung mit formalen Methoden: Vorlesung (45h / 105h / DE / SS / 90) Softwaremodellierung mit formalen Methoden: Übung (30h / 0h / DE / SS / 90)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Softwaremodellierung mit formalen Methoden: Kurs Modellierung	
Inhalte	
Softwaremodellierung mit formalen Methoden: Formale Methoden sind Sprachen zur Modellierung/Spezifikation von Systemen. Ein Modell eines (Soft- oder Hardware) Systems beschreibt auf einer gewissen Abstraktionsebene die Funktionalität des Systems. Im Gegensatz zu (den meisten) Programmiersprachen besitzen formale Methoden eine genau festgelegte Semantik, d.h. eine mathematische Beschreibung der Bedeutung einer Spezifikation. Diese Festlegung der Semantik erlaubt es, das Systemmodell bereits vor der eigentlichen Implementierung formal zu analysieren und mögliche Fehler frühzeitig zu finden. In der Vorlesung werden verschiedene formale Methoden eingeführt, die für unterschiedliche Systemarten geeignet sind. Für jede dieser formalen Methoden werden Semantik und Analysetechniken vorgestellt und Modellierungsbeispiele zur Illustration des Einsatzbereiches besprochen. Am Anfang der Vorlesung wird es vorrangig um die Modellierung von Parallelität und Kommunikation gehen. Hier werden Petrinetze und die Prozessalgebra CCS vorgestellt. Danach werden Sprachen zur Beschreibung von zeitlichen Aspekten (Timed Automata) und zustandsbasierte Formalismen zur Spezifikationen von Daten und Operationen (Z und Object-Z) erläutert.	
Lernergebnisse / Fachkompetenzen	

Studierende sind in der Lage, Software- wie Hardwaresysteme formal zu modellieren. Sie können entscheiden, welche Formalismen für die Modellierung am geeignetsten sind. Studierenden können Sicherheitseigenschaften ihrer Modelle analysieren und dafür Werkzeuge einsetzen. Sie besitzen die Fähigkeit die Semantik von neuen Formalismen zu definieren und existierenden Analyseverfahren anzupassen.

Nichtkognitive Kompetenzen

- Gruppenarbeit
- Lernkompetenz

Methodische Umsetzung

Softwaremodellierung mit formalen Methoden: Eine Mischung aus Folien und Tafelanschrieb. Alle wichtigen Konzepte und Techniken werden in Übungen anhand von Beispielen weiter vertieft.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Heike Wehrheim

Lernmaterialien, Literaturangaben

Softwaremodellierung mit formalen Methoden: Ernst-Rüdiger Olderog, Henning Dierks: Real-time Systems (für Abschnitt Timed Automata) Vorlesungsfolien, Übungsaufgaben, evtl. Skript

Sonstige Hinweise
keine

3.33 Pflichtmodul: Softwaretechnikpraktikum

Modulname	Softwaretechnikpraktikum / Software Engineering Project
Workload	240 h
Leistungspunkte	8 LP
Studiensemester	• Softwaretechnikpraktikum : 3
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Softwaretechnikpraktikum: Vorlesung (15h / 150h / DE / WS / 150) Softwaretechnikpraktikum: Übung (30h / 0h / DE / WS / 10) Softwaretechnikpraktikum: (45h / 0 h / DE / WS / 150)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Softwaretechnikpraktikum: • Objektorientierte Programmierung mit Java • Modellbasierter Softwareentwurf mit UML	
Inhalte	
Softwaretechnikpraktikum: Das Softwaretechnikpraktikum ist eine praxisorientierte Lehrveranstaltung inklusive Vorlesungen zum Thema Projektmanagement. Eine komplexe Softwareentwicklungsaufgabe wird im Team von ca. zehn Studierenden unter Verwendung von UML und Java bearbeitet. Schwerpunkt des Praktikums ist die Vermittlung von Erfahrungen mit der gruppenbasierten Softwareentwicklung unter Benutzung marktüblicher Werkzeuge, Methoden und Prozesse.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden sollen Sprachen und Werkzeuge im Softwareentwicklungsprozess einsetzen sowie den organisatorischen Ablauf eines Softwareprojekts von der Anforderungsdefinition bis zur Abgabe kennen lernen. Darüber hinaus sollen die Studierenden den praktischen Nutzen von planerisch durchdachten Projekten, sowie die Probleme gruppenorientierter Softwareentwicklung und erste Ansätze zu ihrer Bewältigung kennen lernen.	
Nichtkognitive Kompetenzen	
• Einsatz und Engagement • Gruppenarbeit • Kooperationskompetenz • Selbststeuerungskompetenz	

Methodische Umsetzung

Softwaretechnikpraktikum: Durchführung eines Projekts mit regelmäßiger Abgabe von Arbeitsergebnissen (Modulteilprüfungen), protokollierten Gruppensitzungen und einer Abschlusspräsentation.

Prüfungsleistung (Dauer)

keine
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

Klausur (120-180 Minuten, 30% der Modulnote) und Softwareprojekte mit Dokumentation (70% der Modulnote).
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Studienleistung / qualifizierte Teilnahme

Studienleistung: Praktikumsarbeit
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 4 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. Eric Bodden

Lernmaterialien, Literaturangaben

Softwaretechnikpraktikum:
• Balzert, H. (2008). Lehrbuch der Softwaretechnik. Bd. 3: Softwaremanagement. Hrsg. von C. Ebert. 2. Aufl. Lehrbücher der Informatik. Heidelberg: Spektrum. ISBN: 978-3-8274-1161-7.
• Balzert, H. (2011). Lehrbuch der Softwaretechnik. Bd. 2: Entwurf, Implementierung, Installation und Betrieb. 3. Aufl. Lehrbücher der Informatik. Heidelberg: Spektrum. ISBN: 978-3-8274-1706-0.
• Sommerville, I. (2012). Software Engineering. 9. Aufl. Always Learning. München: Pearson. ISBN: 978-3-86894-099-2.

Sonstige Hinweise

Im Softwaretechnikpraktikum ist die erfolgreiche Bearbeitung von Projekten durch die Abgabe von Software und Dokumentation als phasenbezogene Prüfung nachzuweisen. Es wird eine Note für die Gesamtheit der bearbeiteten Projekte vergeben.

3.34 Pflichtmodul: Stochastik für Informatiker

Modulname	Stochastik für Informatiker / Stochastic for Computer Science
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Stochastik für Informatiker : 3
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Stochastik für Informatiker: Vorlesung (45h / 105h / DE / WS / 200) Stochastik für Informatiker: Übung (30h / 0h / DE / WS / 25)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Erfolgreicher Abschluss der Module Analysis für Informatiker und Lineare Algebra für Informatiker	
Empfohlene Kenntnisse	
Stochastik für Informatiker:	
Inhalte	
Stochastik für Informatiker: Deskriptive Statistik und Datenanalyse, Klassische Wahrscheinlichkeitsmodelle, Axiomatik, Standardverteilungen (u.a. Binomial), Satz von Bayes und Anwendungen, Bsp. für nicht-dis-krete Ws.räume, Zufallsgrößen und ihre Momente, Quantile, Gesetze der großen Zahlen, Zent-raler Grenzwertsatz, Schätzen (inkl.-Konfidenzintervalle) und Testen, Simulation und Zufalls-zahlen, Markovketten, mehrdimensionale Wahrscheinlichkeitsverteilungen Deskriptive Statistik und Datenanalyse • planen statistische Erhebungen (Befragung, Beobachtung oder Experiment), führen sie durch und werten sie aus • lesen und erstellen grafische Darstellungen für uni- und bivariate Daten (z.B. Kreuzta-belle) und bewerten deren Eignung für die jeweilige Fragestellung • bestimmen und verwenden uni- und bivariate Kennwerte (z.B. Mittelwerte, Streumaße, Korrelationen, Indexwerte) und interpretieren sie angemessen Zufallsmodellierung • modellieren mehrstufige Zufallsversuche durch endliche Ergebnismengen und nutzen geeignete Darstellungen (Baumdiagramm, Mehrfeldertafel) • rechnen und argumentieren mit Wahrscheinlichkeiten, bedingten Wahrscheinlichkeiten, Erwartungswerten und stochastischer Unabhängigkeit • erläutern inhaltlich das Bernoullische Gesetz der großen Zahlen und den zentralen Grenzwertsatz und deren Konsequenzen • verwenden diskrete und kontinuierliche Verteilungen und ihre Eigenschaften zur Mo-dellierung Stochastische Anwendungen • kennen Beispiele für die Anwendung von Stochastik in verschiedenen Wissenschaften (ökonomie, Physik, . . .) • schätzen in Zufallssituationen Parameter aus Daten	

• führen Hypothesentests durch und reflektieren deren zentralen Schritte und bestimmen Konfidenzintervalle
• erläutern Unterschiede zwischen Bayes-Statistik und klassischen Testverfahren Neue Medien
• verwenden Tabellenkalkulation und statistische Software zur Darstellung und explorativen Analyse von Daten
• simulieren Zufallsversuche computergestützt

Lernergebnisse / Fachkompetenzen

• Kenntnis der Bedeutung der Stochastik in Gesellschaft und Wissenschaft.
• Sicherer Umgang mit den Begriffen der Stochastik in Wort und Schrift.
• Verständnis des mathematischen Sachverhaltes und den damit verbundenen Denkweisen.
• Verständnis der Beweise. Befähigung zur Lösung von Übungsaufgaben zur Stochastik. Fähigkeit des Erkennens von Verbindungen innerhalb der Stochastik beziehungsweise zwischen der Stochastik und anderen Bereichen der Mathematik.
• Durchführung von einfachen statistischen Analysen. Befähigung zum Umgang mit einem Software-Paket zur Stochastik.

Nichtkognitive Kompetenzen

• Lernkompetenz
• Selbststeuerungskompetenz

Methodische Umsetzung

Stochastik für Informatiker:

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 6 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Prof. Dr. Jürgen Klüners
Lernmaterialien, Literaturangaben
Stochastik für Informatiker: keine
Sonstige Hinweise
keine

3.35 Pflichtmodul: Studium Generale

Modulname	Studium Generale / General Studies
Workload	210 h
Leistungspunkte	7 LP
Studiensemester	• Studium Generale : beliebig • Studium Generale : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Studium Generale: Vorlesung (45h / 135h / DE / WS / 30) Studium Generale: Übung (30h / 0h / DE / WS / 30) Studium Generale: Vorlesung (90h / 225h / DE / WS / 30) Studium Generale: Übung (45h / 0h / DE / WS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
Beliebige Veranstaltungen außerhalb der Informatik können gewählt werden.	
Teilnahmevoraussetzungen	
keine	
Empfohlene Kenntnisse	
Studium Generale: Abhängig von den gewählten Veranstaltungen. Studium Generale: Abhängig von den gewählten Veranstaltungen.	
Inhalte	
Studium Generale: Abhängig von den gewählten Veranstaltungen. Studium Generale: Abhängig von den gewählten Veranstaltungen.	
Lernergebnisse / Fachkompetenzen	
Die Studierenden erweitern ihren wissenschaftlichen Horizont über die Grenzen der Informatik und des gewählten Nebenfaches hinaus. Je nach gewählter Veranstaltung haben sie Kompetenzen im Bereich Kommunikationsfähigkeit, Teamarbeit und Präsentationstechniken erworben.	
Nichtkognitive Kompetenzen	
• Einsatz und Engagement • Kooperationskompetenz • Medienkompetenz • Schreib- und Lesekompetenz (wissenschaftlich)	
Methodische Umsetzung	

Studium Generale: Abhängig von den gewählten Veranstaltungen.
Studium Generale: Abhängig von den gewählten Veranstaltungen.
Prüfungsleistung (Dauer)
Prüfung im Studium Generale Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.
Modulteilprüfungen
keine
Studienleistung / qualifizierte Teilnahme
Qualifizierte Teilnahme: Qualifizierte Teilnahme im Studium Generale Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.
Voraussetzungen für die Teilnahme an der Prüfung
keine
Voraussetzungen für die Vergabe von Credits
Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.
Gewichtung für die Gesamtnote
Das Modul wird mit 7 Credits gewichtet.
Modul wird in folgenden Studiengängen verwendet
–
Modulbeauftragte/r
Dozenten der Universität Paderborn
Lernmaterialien, Literaturangaben
Studium Generale: Abhängig von den gewählten Veranstaltungen.
Studium Generale: Abhängig von den gewählten Veranstaltungen.
Sonstige Hinweise
Eine beliebige Kombination von Veranstaltungen außerhalb der Informatik und im Umfang von maximal 7 LP muss gewählt werden. Nebenfach und Studium Generale haben einen Umfang von insgesamt 25 LP. Die angegebene Verteilung der LP auf Lehrveranstaltungen ist nur exemplarisch.

3.36 Pflichtmodul: Systemsoftware und systemnahe Programmierung

Modulname	Systemsoftware und systemnahe Programmierung / System software and system-level programming
Workload	270 h
Leistungspunkte	9 LP
Studiensemester	• Praktikum: Systemsoftware und systemnahe Programmierung : 4 • Systemsoftware und systemnahe Programmierung : 4

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Praktikum: Systemsoftware und systemnahe Programmierung : (0h / 30h / DE / SS / 3)
Systemsoftware und systemnahe Programmierung : Vorlesung (60h / 150h / EN / SS / 200)
Systemsoftware und systemnahe Programmierung : Übung (30h / 0h / EN / SS / 25)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

keine

Empfohlene Kenntnisse

Praktikum: Systemsoftware und systemnahe Programmierung : Grundlagen der Programmierung.

Systemsoftware und systemnahe Programmierung : Es ist dringend zu empfehlen, die Vorlesungen Programmierung und Modellierung erfolgreich abgeschlossen zu haben. Ebenso sollten Grundlagen der Rechnerarchitektur bekannt sein.

Inhalte

Praktikum: Systemsoftware und systemnahe Programmierung : Begleitend zur Vorlesung Systemsoftware und systemnahe Programmierung werden in diesem Programmierpraktikum Techniken der systemnnahen Programmierung praktisch erprobt und eingeübt. Studierende werden in konkreten Projekte das Problem analysieren, geeignete Programmiertechniken auswählen, praktisch realisieren und eine quantitative Leistungsbewertung durchführen.

Systemsoftware und systemnahe Programmierung : Einführung in grundlegende Probleme, Aufgaben, Herausforderungen und Herangehensweisen für systemnahe Software (z.B. Betriebssysteme, Protokollstacks). Es wird ein konzeptioneller Zugang gewählt (anstelle eines beispielorientierten Ansatzes); besonderer Wert wird auf praktisch orientierte Programmierübungen in kleinen Projekten gelegt, die den selbständigen Umgang mit der Materie vertiefen.

Lernergebnisse / Fachkompetenzen

Studierende können Aufgabenstellungen der Systemsoftware identifizieren, unterschiedliche Ansätze zu Problemlösungen benennen, klassifizieren und unterscheiden, deren Vor- und Nachteile evaluieren und

für ein Problem eine geeignete Lösung auswählen. Sie sind in der Lage, diese Verfahren in eigenen Anwendungen gezielt zum Einsatz zu bringen (bspw. Semaphoren zur Koordination nebenläufiger Aktivitäten).
Studierende können ggf. neue Lösungen konstruieren (bspw. Scheduling-Strategie) und deren Leistungsfähigkeit systematisch durch Einsatz geeigneter (mathematischer oder informatischer) Werkzeuge analysieren, deren Eignung evaluieren und mit Alternativen kontrastieren.

Nichtkognitive Kompetenzen

- Gruppenarbeit
- Lernkompetenz
- Selbststeuerungskompetenz

Methodische Umsetzung

Praktikum: Systemsoftware und systemnahe Programmierung : Programmieraufgaben in kleineren Teams.

Systemsoftware und systemnahe Programmierung : Die Vorlesung ist überwiegend folienorientiert, mit begleitendem Tafeleinsatz und Aufgaben für die Studierenden während der Vorlesung. Sie wird sowohl durch Tafelübung als auch durch Kleingruppentutorien begleitet. Studierende haben in den Kleingruppen Gelegenheit, Aufgaben in der Gruppe zu bearbeiten und Übungsblätter durch Tutoren benoten zu lassen.

Prüfungsleistung (Dauer)

Klausur (120 - 180 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Qualifizierte Teilnahme: Praktikumsarbeit mit anschließendem Gespräch
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 9 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r
Prof. Dr. rer. nat. Holger Karl
Lernmaterialien, Literaturangaben
Praktikum: Systemsoftware und systemnahe Programmierung : Aufgabenstellung, man pages, eigenständige Recherche zu unterstützender Kommunikation. Systemsoftware und systemnahe Programmierung : Standardlehrbücher (z.B. Stallings, Betriebssysteme); Foliensatz der VL; Übungsblätter.
Sonstige Hinweise
keine

3.37 Wahlpflichtmodul: Verteilte Algorithmen und Datenstrukturen

Modulname	Verteilte Algorithmen und Datenstrukturen / Distributed Algorithms and Data Structures
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Verteilte Algorithmen und Datenstrukturen : beliebig
Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)	
Verteilte Algorithmen und Datenstrukturen: Vorlesung (45h / 105h / DE / SS / 60) Verteilte Algorithmen und Datenstrukturen: Übung (30h / 0h / DE / SS / 30)	
Wahlmöglichkeiten innerhalb des Moduls	
keine	
Teilnahmevoraussetzungen	
Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.	
Empfohlene Kenntnisse	
Verteilte Algorithmen und Datenstrukturen: Datenstrukturen und Algorithmen	
Inhalte	
Verteilte Algorithmen und Datenstrukturen: Die Vorlesung gibt eine Einführung in die Grundlagen der verteilten Algorithmen und Datenstrukturen. Einen Schwerpunkt bilden dabei hochskalierbare Datenstrukturen, die auch für sehr große und dynamische verteilte Systeme anwendbar sind. Nach einer Einführung in die Netzwerktheorie werden zunächst grundlegende Designprinzipien für verteilte Algorithmen und Datenstrukturen vorgestellt wie z.B. das Konzept der selbst-stabilisierenden Systeme. Danach folgt eine kurze Einführung in verteilte Programmierung, damit die in der Vorlesung vorgestellten Datenstrukturen auch von den Studenten implementiert werden können. Anschließend werden zunächst prozessorientierte Datenstrukturen und dann informationsorientierte Datenstrukturen vorgestellt.	
Lernergebnisse / Fachkompetenzen	
• Kenntnis ausgewählter verteilter Algorithmen und Datenstrukturen • Kenntnis wesentlicher Konzepte im Bereich verteilter Algorithmen und Datenstrukturen • Fähigkeit, selbstständig adäquate Techniken und Verfahren im Bereich der verteilten Algorithmen und Datenstrukturen zu entwickeln • Fähigkeit, algorithmische Probleme gemäß ihrer Lösbarkeit und Komplexität einzuschätzen	

- Fähigkeit, grundlegende verteilte Datenstrukturen zu implementieren

Nichtkognitive Kompetenzen

- Gruppenarbeit
- Lernkompetenz
- Schreib- und Lesekompetenz (wissenschaftlich)
- Selbststeuerungskompetenz

Methodische Umsetzung

Verteilte Algorithmen und Datenstrukturen: Vorlesung mit Übungen, Hausaufgaben und Softwareprojekt

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. rer. nat. Christian Scheideler

Lernmaterialien, Literaturangaben

Verteilte Algorithmen und Datenstrukturen: Skript

Sonstige Hinweise

keine

3.38 Wahlpflichtmodul: Verteilte Systeme

Modulname	Verteilte Systeme / Distributed Systems
Workload	180 h
Leistungspunkte	6 LP
Studiensemester	• Verteilte Systeme : 5

Lehrveranstaltungen: Lehrform (Kontaktzeit / Selbststudium / Sprache / Termin / Gruppengröße)

Verteilte Systeme: Vorlesung (45h / 105h / DE / WS / 100)
Verteilte Systeme: Übung (30h / 0h / DE / WS / 30)

Wahlmöglichkeiten innerhalb des Moduls

keine

Teilnahmevoraussetzungen

Die Module Programmierung, Programmiersprachen, Software Engineering, Datenbanksysteme, Modellierung, Datenstrukturen und Algorithmen, Digitaltechnik, Analysis für Informatiker und Lineare Algebra für Informatiker müssen bestanden sein. Bei Studierenden des Nebenfachs Mathematik werden dabei die Module „Analysis 1“ und „Lineare Algebra 1“ statt der Module „Lineare Algebra für Informatiker“ und „Analysis für Informatiker“ berücksichtigt.

Empfohlene Kenntnisse

Verteilte Systeme: Vorlesung Systemsoftware und systemnahe Programmierung. Grundlegendes Verständnis von Algorithmen.

Inhalte

Verteilte Systeme: Diese Veranstaltung behandelt architekturelle, konzeptionelle und pragmatische Fragestellungen beim Entwurf, Einsatz und Betrieb von verteilten Systemen in der Informatik – Systeme, bei denen Daten oder Kontrollfunktionen nicht mehr an einem Ort konzentriert sind sondern die sich aus unabhängigen IT-Systemen zusammensetzen. Dabei wird der Systemaspekt betont; grundlegende algorithmische Fragestellungen werden ebenfalls behandelt. Zusätzlich werden Fragen der Leistungsbewertung und Verlässlichkeit behandelt.
Bemerkungen:
• Die Veranstaltung lässt sich sehr gut mit der Veranstaltung Rechnernetze ergänzen.
• In der Regel findet die Veranstaltung halbsemestrig in der zweiten Semesterhälfte statt; in der ersten Semesterhälfte die Veranstaltung Rechnernetze.

Lernergebnisse / Fachkompetenzen

Teilnehmer sind in der Lage,
• verteilte Systeme zur Erhöhung von Leistungsfähigkeit oder Fehlertoleranz zum Einsatz zu bringen und geeignet zu dimensionieren;
• sie können geeignete Systemansätze (Client-Server, P2P, ...) benennen und situationsgerecht auswählen und diese Auswahl architekturell begründen;

• sie haben algorithmische Problemstellungen für verteilte Systeme verstanden, können aus einer allgemeinen Problembeschreibung die zu lösenden algorithmische Aufgabe isolieren und eine begründete Wahl treffen.

Nichtkognitive Kompetenzen

• Einsatz und Engagement
• Lernkompetenz

Methodische Umsetzung

Verteilte Systeme: Folienbasierte Vorlesung mit Tafelanschrieb, durch Übung begleitet. Übungen dabei sowohl konzeptionell/analytisch als auch mit praktischen Aufgaben.

Prüfungsleistung (Dauer)

Klausur (90 - 120 Minuten)
Vom jeweiligen Lehrenden werden Art und Dauer der Prüfungsleistung spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben.

Modulteilprüfungen

keine

Studienleistung / qualifizierte Teilnahme

Studienleistung: schriftliche Übungsaufgaben
Vom jeweiligen Lehrenden wird spätestens in den ersten drei Wochen der Vorlesungszeit bekannt gegeben, wie die Studienleistung bzw. qualifizierte Teilnahme konkret zu erbringen ist.

Voraussetzungen für die Teilnahme an der Prüfung

Bestehen der Studienleistung.

Voraussetzungen für die Vergabe von Credits

Die Vergabe von Credits erfolgt, wenn die Modulabschlussprüfung bestanden ist.

Gewichtung für die Gesamtnote

Das Modul wird mit 12 Credits gewichtet.

Modul wird in folgenden Studiengängen verwendet

–

Modulbeauftragte/r

Prof. Dr. rer. nat. Holger Karl

Lernmaterialien, Literaturangaben

Verteilte Systeme: Folien, Standardlehrbücher (insbes. Colouris,Distributed Systems Concepts and Design; Tanenbaum, Verteilte Systeme), Übungsblätter.

Sonstige Hinweise

keine

Anhang A

Überblickstabellen

A.1 Studienrichtungen und Module

	Erster Studienabschnitt (S. 11)	Informatikgebiet Algorithmen und Komplexität (S. 13)	Informatikgebiet Computer Systeme (S. 14)	Informatikgebiet Daten und Wissen (S. 15)	Informatikgebiet Softwaretechnik (S. 16)	Informatikgebiet Vertiefung (S. 17)
Analysis für Informatiker (S. 20)	X	-	-	-	-	-
Angriffssicherer Softwareentwurf (S. 23)	-	-	-	-	X	X
Bachelor-Abschlussarbeit (S. 25)	-	-	-	-	-	-
Berechenbarkeit und Komplexität (S. 28)	X	-	-	-	-	-
Betriebssysteme (S. 30)	-	-	X	-	-	X
Computer Graphics Rendering (S. 32)	-	-	-	X	-	X
Data Mining (S. 35)	-	-	-	X	-	X
Databases and Information Systems (S. 37)	-	-	-	X	-	X
Datenbanksysteme (S. 40)	X	-	-	-	-	-
Datenstrukturen und Algorithmen (S. 43)	X	-	-	-	-	-
Digitaltechnik (S. 46)	X	-	-	-	-	-
Einführung in Kryptographie (S. 48)	-	X	-	-	-	X
Eingebettete Systeme (S. 50)	-	-	X	-	-	X
Gestaltung von Nutzungsschnittstellen (S. 53)	X	-	-	-	-	-
Grundlagen Wissensbasierter Systeme (S. 55)	-	-	-	X	-	X

Modellbasierte Softwareentwicklung (S. 72)	-	-	-	-	X	X
Modellierung (S. 74)	X	-	-	-	-	-
Parallelität und Kommunikation (S. 76)	-	X	-	-	-	X
Programmiersprachen (S. 78)	X	-	-	-	-	-
Programmiersprachen und Übersetzer (S. 80)	-	-	-	-	X	X
Programmierung (S. 83)	X	-	-	-	-	-
Rechnerarchitektur (S. 86)	X	-	-	-	-	-
Rechnernetze (S. 88)	-	-	X	-	-	X
Schlüsselqualifikation (S. 90)	X	-	-	-	-	-
Software Engineering (S. 93)	X	-	-	-	-	-
Softwaremodellierung mit Formalen Methoden (S. 96)	-	-	-	-	X	X
Softwaretechnikpraktikum (S. 99)	X	-	-	-	-	-
Stochastik für Informatiker (S. 102)	X	-	-	-	-	-
Studium Generale (S. 105)	-	-	-	-	-	-
Systemsoftware und systemnahe Programmierung (S. 107)	X	-	-	-	-	-
Verteilte Algorithmen und Datenstrukturen (S. 110)	-	X	-	-	-	X
Verteilte Systeme (S. 112)	-	-	X	-	-	X

A.2 Module und Lehrveranstaltungen

	Analysis für Informatiker (S. 20)	Angriffssicherer Softwareentwurf (S. 23)	Bachelor-Abschlussarbeit (S. 25)	Berechenbarkeit und Komplexität (S. 28)	Betriebssysteme (S. 30)	Computer Graphics Rendering (S. 32)	Data Mining (S. 35)	Databases and Information Systems (S. 37)	Datenbanksysteme (S. 40)	Datenstrukturen und Algorithmen (S. 43)	Digitaltechnik (S. 46)	Einführung in Kryptographie (S. 48)	Eingebettete Systeme (S. 50)	Gestaltung von Nutzungsschnittstellen (S. 53)	Grundlagen Wissensbasierter Systeme (S. 55)	Grundlegende Algorithmen (S. 57)	Interaktionsgestaltung (S. 60)	IT Sicherheit (S. 63)	Komplexitätstheorie (S. 65)	Lineare Algebra für Informatiker (S. 67)	Logik und Deduktion (S. 70)	Modellbasierte Softwareentwicklung (S. 72)	Modellierung (S. 74)	Parallelität und Kommunikation (S. 76)	Programmiersprachen (S. 78)	Programmiersprachen und Übersetzer (S. 80)	Programmierung (S. 83)	Rechnerarchitektur (S. 86)	Rechnernetze (S. 88)	Schlüsselqualifikation (S. 90)	Software Engineering (S. 93)	Softwaremodellierung mit Formalen Methoden (S. 96)	Softwaretechnikpraktikum (S. 99)	Stochastik für Informatiker (S. 102)	Studium Generale (S. 105)	Systemsoftware und systemnahe Programmierung (S. 107)	Verteilte Algorithmen und Datenstrukturen (S. 110)	Verteilte Systeme (S. 112)
Analysis für Informatiker (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Angriffssicherer Softwareentwurf (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Bachelorarbeit Abschlussarbeit (S. ??)	–	–	12	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Bachelorarbeit Arbeitsplan (S. ??)	–	–	3	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Berechenbarkeit und Komplexität (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Betriebssysteme (S. ??)	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Computer Graphics Rendering (S. ??)	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Data Mining (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Databases and Information Systems (S. ??)	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Datenbanksysteme (S. ??)	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Datenstrukturen und Algorithmen (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Digitaltechnik (S. ??)	–	–	–	–	–	–	–	–	–	–	5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Einführung in Kryptographie (S. ??)	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Eingebettete Systeme (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Gestaltung von Nutzungsschnittstellen (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Grundlagen Wissensbasierter Systeme (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Grundlegende Algorithmen (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Interaktionsgestaltung (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
IT Sicherheit (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Komplexitätstheorie (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Lineare Algebra für Informatiker (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Logik und Deduktion (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Modellbasierte Softwareentwicklung (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Modellierung (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	8	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Parallelität und Kommunikation (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–	–	–
Praktikum: Datenstrukturen und Algorithmen (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Praktikum: Software Engineering (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Praktikum: Systemsoftware und systemnahe Programmierung (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Programmiersprachen (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Programmiersprachen und Übersetzer (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–	–
Programmierung (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–	–	–
Proseminar (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	4	–	–	–	–	–	–	–	–	–
Rechnerarchitektur (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–	–
Rechnernetze (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–	–
Software Engineering (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–	–
Softwaremodellierung mit formalen Methoden (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–	–
Softwaretechnikpraktikum (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–	–
Stochastik für Informatiker (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6	–	–	–
Studium Generale (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Studium Generale (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Systemsoftware und systemnahe Programmierung (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Verteilte Algorithmen und Datenstrukturen (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Verteilte Systeme (S. ??)	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	6