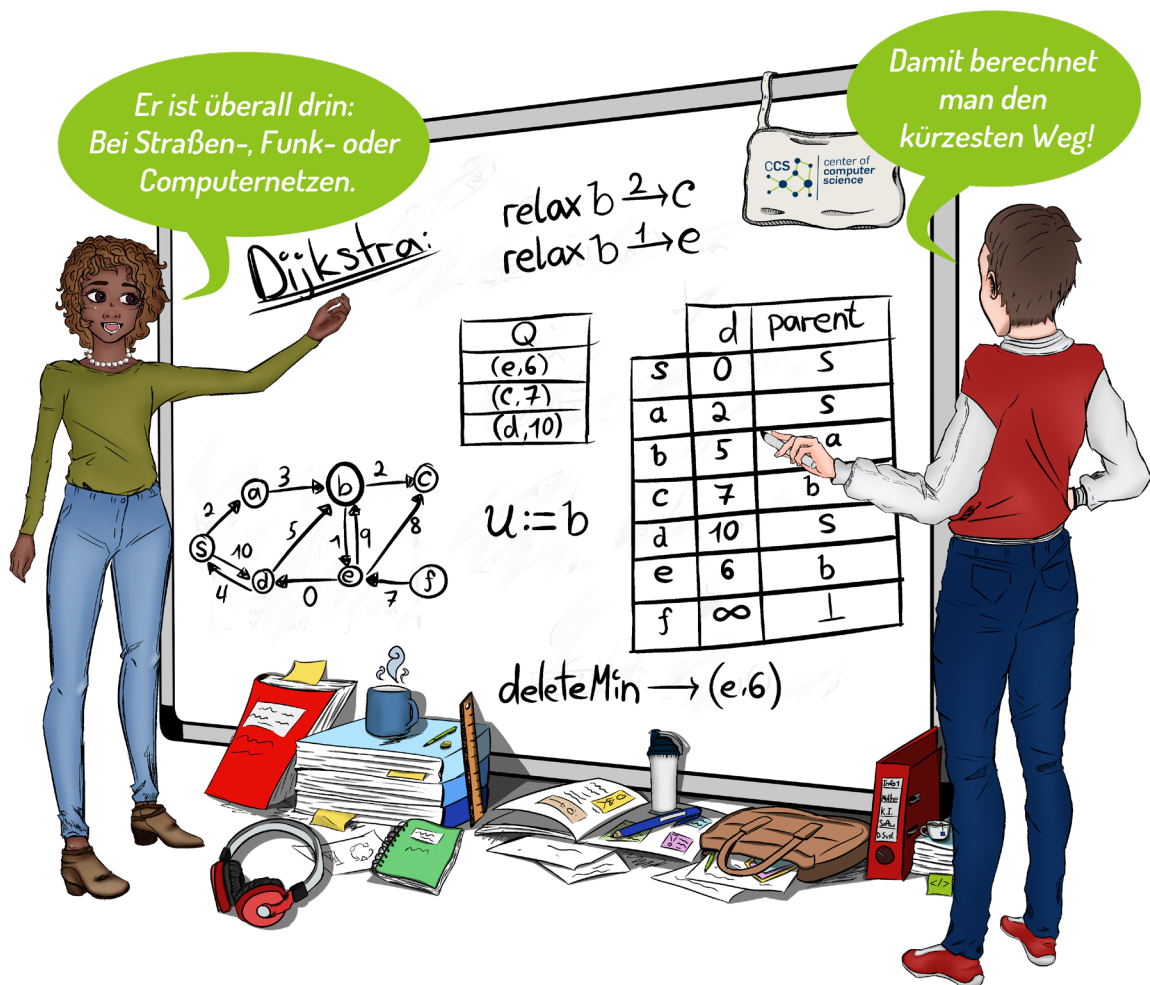


Modulhandbuch Bachelor of Science (B.Sc.)

Informatik [PO22]

Stand: Sommersemester 2025

<https://informatik.rub.de/studium/studiengaenge/inf/bsc/>



Studienplan Bachelor Informatik [PO 22] der Ruhr-Universität Bochum

Nr	Modul	Umfang (LP)	Empfohlenes Se- mester	Bewertung
Pflichtbereich				
1	Mathematik 1	9	1	benotet
2	Informatik 1	12	1	benotet
3	Technische Informatik 1	5	1	benotet
4	English for Computer Science 1	3+3	1+2	benotet
5	Mathematik 2	9	2	benotet
6	Informatik 2	8	2	benotet
7	Technische Informatik 2	5	2	benotet
8	Programmierung und Programmiersprachen	6	2	benotet
9	English for Computer Science 2	3+3	3+4	benotet
10	Mathematik 3	9	3	benotet
11	Informatik 3	8	3	benotet
12	Software Engineering	5	3	benotet
13	Technische Informatik 3	5	3	benotet
14	Einführung in die künstliche Intelligenz	5	4	benotet
15	Datenbanksysteme	7	4	benotet
16	Computernetze	5	4	benotet
17	Betriebssysteme	5	4	benotet
18	Software Engineering Praktikum	5	4	benotet
Wahlpflichtbereich				
19	Vertiefungspraktikum**	3	5	unbenotet
20	Vertiefungsseminar**	3	5	benotet
21	Vertiefungsmodule***	20	5-6	benotet
Wahlbereich				
22	Freie Wahlmodule****	a*	1-6	unbenotet
Praktische Ausbildung				
23	Praktische Ausbildung*****	b*	5-6	unbenotet
Abschlussarbeit				
24	Bachelorarbeit und Kolloquium	12 + 3	6	benotet

* $4 \leq a \leq 9, 10 \leq b \leq 15, a+b = 19$

** Informationen zu den im Semester wählbaren Programmierpraktika und Vertiefungsseminaren befinden sich im Vorlesungsverzeichnis.

*** Hier müssen Vertiefungsmodule aus dem Bereich Informatik im Umfang von mindestens 20 LP gewählt werden. Informationen zu den wählbaren Modulen befinden sich im jeweils aktuellen Modulhandbuch.

**** Hier können (nahezu) alle Veranstaltungen des Vorlesungsverzeichnisses der RUB, sowie Veranstaltungen im Rahmen der Universitätsallianz Ruhr gewählt werden.

***** Hier ist ein Industriepraktikum oder Forschungspraktikum zu absolvieren (siehe §5 (3)).

Angebote Vertiefungsmodulen

	Lehrveranstaltung	Lehreinheit	Umfang (CP)	Semester	Bewertung
Vertiefungsmodulen					
	Algorithmenparadigmen	Informatik	5	WS	benotet
	Artificial Neural Networks	Informatik	6	WS	benotet
	Einführung in die Kryptographie I	Informatik	5	WS	benotet
	Introduction to Data Science	Informatik	5	WS	benotet
	Natural Language Processing with Deep Learning	Informatik	5	WS	benotet
	Network Planing	ETIT	5	WS	benotet
	Private and Anonymous Communication	Informatik	5	WS	benotet
	Programming for Modern Machine Learning	Informatik	6	WS	benotet
	Proofs are programs	Informatik	5	WS	benotet
	Quantum Information and Computation	Informatik	5	WS	benotet
	Software Security 1	Informatik	5	WS	benotet
	System Performance Evaluation	Informatik	5	WS	benotet
	Cache Attacks	Informatik	5	SS	benotet
	Distributed Systems Algorithms (ehemals Distributed Systems)	Informatik	5	SS	benotet
	Einführung in die Kryptographie II	Informatik	5	SS	benotet
	Functional Programming	Informatik	5	SS (nicht im SS 25)	benotet
	Highlights of Theoretical Computer Science	Informatik	10	SS	benotet
	Programmanalyse	Infortmatik	5	SS	benotet
	Public Key Kryptanalyse 1	Informatik	5	SS (nicht im SS 25)	benotet
	Systemsicherheit	Informatik	5	SS	benotet
	Web Engineering	Baulng	5	SS	benotet
	Autonomous Vehicles and Artificial Intelligence	Informatik	5	Letztmalig SS 23	benotet
	Game Development	Informatik	6	Letztmalig SS 23	benotet
	Logik in der Informatik	Informatik	5	Letztmalig SS 23	benotet
	Proofs are programs	Informatik	5	Letztmalig SS 23	benotet
	Nebenläufige Programmierung	Informatik	5	Letztmalig SS 23	benotet
	Statistical Learning and Data Mining	Mathematik	5	Letztmalig SS 23	benotet
	Model Checking	Informatik	5	Letztmalig SS 23	benotet
	Information Theory	Informatik	5	Letztmalig SS 23	benotet
	Digitale Forensik	Informatik	5	Letztmalig WS 23/24	benotet

Angebote Vertiefungsseminare und Vertiefungspraktika

	Lehrveranstaltung	Lehreinheit	Umfang (CP)	Semester	Bewertung
Vertiefungsseminare					
	Seminar Security Engineering	Informatik	3	WS/SS	benotet
	Seminar Software and Internet Security	Informatik	3	WS/SS	benotet
	Seminar Randomisierte Algorithmen	Informatik	3	WS/SS (nicht im SS 25)	benotet
	Seminar Safety and Reliability in Artificial Intelligence	Informatik	3	WS/SS	benotet
	Seminar Ressourceneffiziente Systemsoftware	Informatik	3	WS/SS	benotet
	Seminar Automated Software Engineering	Informatik	3	WS/SS	benotet
	Seminar Algorithmen	Informatik	3	WS	benotet
	Perlen der theoretischen Informatik (ehemals Grenzen in der theoretischen Informatik)	Informatik	3	WS	benotet
	Seminar Distributed Systems	Informatik	3	WS	benotet
	Seminar Mobile Network Security	Informatik	3	WS	benotet
	Seminar Quantum Cryptography	Informatik	3	WS	benotet
	Seminar Modern Programming Languages	Informatik	3	WS	benotet
	Seminar Quantum Algorithms	Informatik	3	SS (nicht im SS 25)	benotet

Seminar Perlen der Logik (ehemals Satisfiability)	Informatik	3	SS	benotet
Seminar From Biological to Artificial Neural Networks	Informatik	3	SS	benotet
Seminar Networked Systemes	Informatik	3	SS	benotet
Current topics in microarchitectural security	Informatik	3	SS	benotet
Seminar on Applied Privacy and Anonymity	Informatik	3	SS	benotet
Seminar Reinforcement Learning	Informatik	3	SS	benotet
Seminar Machine Learning Applications	Informatik	3	Letztmalig WS 23/24	benotet
Seminar on Knowledge Graphs	Informatik	3	Letztmalig WS 22/23	benotet
Seminar on Current Topics for Systems Security and Privacy	Informatik	3	Letztmalig WS 23/24	benotet
Algorithms for Decision Making	Informatik	3	Letztmalig WS 23/24	benotet
Seminar Quantum Cryptography	Informatik	3	Letztmalig WS 23/24	benotet
Seminar Approximationsalgorithmen	Informatik	3	Letztmalig SS 24	benotet
Vertiefungspraktika				
Initial Research in Internet Security	Informatik	3	WS/SS	unbenotet
Initial Research in Software Security	Informatik	3	WS/SS	unbenotet
Advanced Python Programming	Informatik	3	WS	unbenotet
Unreal C++ Praktikum	Informatik	3	WS	unbenotet
Lab Course: Challenging Problems in Reinforcement Learning	Informatik	3	WS	unbenotet
Android App Evolution	Informatik	3	WS	unbenotet
Autonomous Driving Simulation Lab	Informatik	3	WS	unbenotet
Praktikum IDE Plugin Development	Informatik	3	WS	unbenotet
Praktikum Implementing Post-Quantum Standards and Challenges	Informatik	3	WS	unbenotet
Praktikum Systemsoftwaretechnik	Informatik	3	WS	unbenotet
Research in Microarchitectural Security	Informatik	3	WS	unbenotet
An Introduction to Python for Data Analysis (ehemals Introduction to Python)	Informatik	3	SS	unbenotet
Praktische Kryptanalyse von symmetrischen Chiffren	Informatik	3	SS (nicht SS 25)	unbenotet
Python programming and basic machine learning	Informatik	3	SS	unbenotet
Praktikum Rust	Informatik	3	Letztmalig WS 23/24	unbenotet
Praktikum Worst-Case Performance Evaluation	Informatik	3	Letztmalig WS 23/24	unbenotet
Creating Mystery Twister Crypto Challenges	Informatik	3	SS	unbenotet

Abkürzungen:

ETIT: Fakultät für Elektrotechnik und Informationstechnik
 Baulng: Fakultät für Bau- und Umweltingenieurwissenschaften

SS: Sommersemester
 WS: Wintersemester
 CP: Creditpoints

MODULHANDBUCH

Übersicht der Module

Informatik - Bachelor (1-Fach, PO 2022)

Pflichtbereich

Mathematik 1

Informatik 1

Technische Informatik 1

English for Computer Science 1

Mathematik 2

Mathematik 2

Informatik 2

Programmierung und Programmiersprachen

Technische Informatik 2

English for Computer Science 2

Mathematik 3

Mathematik 3

Informatik 3

Technische Informatik 3

Software Engineering

Betriebssysteme

Computernetze

Einführung in die künstliche Intelligenz

Datenbanksysteme

Software Engineering Praktikum

Wahlpflichtbereich

Introduction to Data Science

Algorithmenparadigmen

Cache Attacks

Distributed Systems Algorithms

Einführung in die Kryptographie 1

Einführung in die Kryptographie 2

Functional Programming

Game Development

Highlights of Theoretical Computer Science [B.Sc.]

Information Theory

Künstliche Neuronale Netze (kein Angebot im WS 24/25)
Model Checking (kein Angebot im SS 24)
Natural language processing with deep learning [B.Sc]
Nebenläufige Programmierung (letztmalig SS 23)
Network Planing (kein Angebot im WS 24/25)
Private and Anonymous Communication
Programmanalyse [B.Sc]
Programming for Modern Machine Learning
Proofs are programs [B.Sc.]
Public Key Kryptanalyse 1 [B.Sc] (nicht im SoSe 25)
Quantum Information and Computation [B.Sc.]
Requirements Engineering
Software Security 1 [B.Sc.]
Statistisches Lernen und Data Mining
System Performance Evaluation (keine Angebot im WS 24/25)
Systemsicherheit
Web-Engineering
Vertiefungspraktikum Informatik
Vertiefungsseminar Informatik

Wahlbereich

Freies Wahlmodul

Praktische Ausbildung

Praktische Ausbildung

Bachelorarbeit

Abschlussarbeit

Titel des Moduls: Mathematik 1 Mathematics 1					
Modul-Nr./Code	Credits 9 CP	Workload 270 h	Semester	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Mathematik 1 - Grundlagen			Kontaktzeit 105 h	Selbststudium 180 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen Erfolgreiches Bestehen der Modulklausur.		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Gregor Leander Lehrende: Prof. Dr. Gregor Leander					
Verwendung des Moduls Bachelor Informatik Bachelor IT-Sicherheit					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls -kennen Studierende grundlegende Begriffe und Schreibweisen der Mathematik - können Studierende die erlernten Techniken selbstständig anwenden und mathematische Sachverhalte darstellen - kennen Studierende die Grundlagen abstrakter mathematischer Strukturen und verschiedene Beispiele für Gruppen, Ringe und Körper - verstehen die Studierenden den abstrakten Vektorraumbegriff über beliebigen Körpern, können mit linearer Unabhängigkeit, Dimensionen und mit linearen Abbildungen umgehen - sind Studierende in der Lage, lineare Gleichungssysteme explizit zu lösen sowie Eigenwerte und Eigenvektoren zu berechnen					
Inhalt Dieses Modul gibt eine allgemeine Einführung in mathematische Grundlagen und behandelt wichtige Gebiete der Linearen Algebra. Folgende Themengebiete werden behandelt: <ul style="list-style-type: none"> • Grundlagen der Mathematik • Grundlegende mathematische Begriffe • Schreibweisen • Aussagenlogik • Mengenlehre • Relationen Algebraische Grundlagen • ganze Zahlen • Restklassen • Gruppen-, Ringe- und Körper-Axiome Lineare Algebra • Vektorräume • Basen • Dimension • Skalarprodukte • lineare Abbildungen • lineare Gleichungssysteme • Basiswechsel • Determinanten • Eigenwerttheorie 					

Lehrformen

Vorlesung und Übungen

Prüfungsformen

Klausurarbeit (120 Minuten)

Voraussetzungen für die Vergabe von Credits

Erfolgreiches Bestehen der Modulklausur.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

9/165: B.Sc. Informatik [PO 22]

9/150: B.Sc. IT-Sicherheit [PO 22]

9/149: B.Sc. IT-Sicherheit [PO 20]

Titel des Moduls: Informatik 1					
Modul-Nr./Code	Credits 12 CP	Workload 360 h	Semester 1	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Vorlesung und Übung: Informatik 1 (212004) Python-Praktikum (212400)			Kontaktzeit 6 Semesterwochenstunden + 10 Tage (80h)	Selbststudium 190 h	Gruppengröße 400 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen Keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Tobias Glasmachers Lehrende: Prof. Dr. Tobias Glasmachers					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik					
Vorkenntnisse Keine					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen die Teilnehmer die wichtigsten Konzepte imperativer und objektorientierter Programmierung, • können die Teilnehmer eigene Programme entwerfen und implementieren, • können die Teilnehmer mit Grundbegriffen der Informatik wie etwa Korrektheit, Laufzeit, Boole'scher Algebra, Invarianten und abstrakten Datentypen arbeiten, • können die Teilnehmer die einfache Datenstrukturen (Arrays, Dictionaries) gezielt einsetzen und kennen Standardalgorithmen darauf, insbesondere zum Sortieren von Arrays. 					
Inhalt <ul style="list-style-type: none"> • Imperative Programmierung (Variablen, Kontrollstrukturen, Funktionen und Rekursion, Fehlerbehandlung, Ereignisbehandlung) • einfache Datenstrukturen (Array und Dictionary) • Objektorientierung (Klassen, Sichtbarkeit, Schnittstellen, Vererbung) • Einführung in eine Reihe von Informatik-Konzepten (Invarianten, Laufzeitanalyse, Sortieralgorithmen, Repräsentation von Daten im Rechner, Boole'sche Algebra) 					
Lehrformen Vorlesung mit Übung plus zweiwöchiges Blockpraktikum Die Vorlesung nutzt das Flipped-Classroom Lehrformat. Sämtliches Vorlesungsmaterial steht online zur Verfügung.					
Prüfungsformen Schriftliche Modulabschlussprüfung (150 Minuten) Erfolgreiche Teilnahme am Praktikum					
Voraussetzungen für die Vergabe von Credits Teilnahme am Python-Praktikum und bestandene schriftliche Modulabschlussprüfung.					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 12/168: B.Sc. Angewandte Informatik					

Titel des Moduls: Technische Informatik 1 Technical Computer Science 1					
Modul-Nr./Code	Credits 5 CP	Workload 150 Stunden	Semester siehe Prüfungsordnung / see Examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Technische Informatik 1 - Rechnerarchitektur (212013)			Kontaktzeit 4 SWS	Selbststudium	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Yuval Yarom Lehrende: Prof. Yuval Yarom					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik B.Sc. IT-Sicherheit/ Informationstechnik					
Vorkenntnisse Keine					
Lernziele (learning outcomes) Upon successful completion of the course, students will be able <ul style="list-style-type: none"> • Identify the main components of a modern processor, describe their functionality, and demonstrate how they each contribute to program execution • Evaluate computer system based on their design, and assess their impact on program performance • Design and develop simple programs in assembly language 					
Inhalt The course introduces the structure and function of modern computers. It explores the various components that comprise the computer, including the execution pipeline, the memory subsystem, data storage, and external devices. A main focus of the course is exploring and analyzing program execution. This start from practical experience with assembly programming and develops to in-depth analysis of how the processor interprets and performs a program. In particular, the course identifies trade-offs made in processor design and their impact on performance.					
Lehrformen Contact teaching consists of two hours of lecture per week, supplemented by two hours of practical exercise sessions. Beyond contact hours, students are expected to read textbook chapters and to answer take-home exercises.					
Prüfungsformen Schriftliche Modulabschlussprüfung (120 Minuten). Bis zu 10% Bonus für Hausaufgaben.					
Voraussetzungen für die Vergabe von Credits Bestandene schriftliche Modulabschlussprüfung.					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)					

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/150: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

Titel des Moduls: English for Computer Science 1 English for Computer Science 1					
Modul-Nr./Code	Credits 6 CP	Workload 180 h	Semester 1	Turnus Jedes Winter- und Sommersemester	Dauer 2 Semester
Lehrveranstaltungen a) English for Computer Science I: Reading Skills Sprachkurs mit integrierten Übungen (2 SWS, 3 CP, im Wintersemester) (251213-16) b) English for Computer Science I: Listening Skills Sprachkurs mit integrierten Übungen (251211-15) (2 SWS, 3 CP, im Sommersemester)			Kontaktzeit 60 h	Selbststudium 120 h	Gruppengröße 25 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Einstufungstest des Zentrums für Fremdsprachenausbildung. Die Termine sind auf der folgenden Internet-Seite zu finden: http://www.zfa.ruhr-uni-bochum.de/lehre/einstufung/index.html.de Achtung: Diese Lehrveranstaltungen werden auf zwei verschiedenen Kompetenzniveaus angeboten: 1. Intermediate (B1/B2-B2) 2. Advanced (B2/C1-C2) Die Studierenden müssen sich für das ihrem Einstufungsergebnis entsprechende Kompetenzniveau anmelden. Bei einem Einstufungsergebnis unterhalb von B1/B2 muss zunächst ein Kurs der Kompetenzstufe B1 belegt werden, der kein Bestandteil von diesem Modul ist. Die Moduleile a) und b) können unabhängig voneinander besucht werden. Es empfiehlt sich jedoch, mit dem Besuch von English for Computer Science 1 anzufangen.		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Frau Melissa Oldfield-Mariano Lehrende: Dozent*innen des Zentrums für Fremdsprachenausbildung (ZFA) der Ruhr-Universität Bochum, Melissa Oldfield-Mariano					
Verwendung des Moduls Dieses Modul wurde exklusiv für diesen Bachelor-Studiengang konzipiert. Daher ist eine Öffnung dieser Veranstaltung für Studierende anderer Studiengänge nicht vorgesehen.					
Vorkenntnisse Englischniveau von ungefähr B2 (GeR). Die bis zum deutschen Abitur erlangten Englischkenntnisse reichen i.d.R. aus. Sollten Zweifel am eigenen Niveau bestehen, wird empfohlen, weitere Lehrveranstaltungen am Zentrum für Fremdsprachenausbildung (ZFA) zur Konsolidierung der Englischkompetenzen zu besuchen.					

Lernziele (learning outcomes)

Nach dem erfolgreichen Abschluss des Moduls haben die Studierenden das volle Kompetenzniveau B2 in der *Fachsprache* erreicht, d.h. in diesem Zusammenhang:

- Die Studierenden verstehen englischsprachige Fachtexte in den von ihnen bereits erlernten Wissenschaftsbereichen
- Die Studierenden verstehen englischsprachige Vorlesungen, Vorträge und Diskussionen soweit, dass sie ihnen problemlos folgen können
- Die Studierenden beherrschen das wichtigste Fachvokabular (passiv und aktiv) und können dieses konform ihrer Fachkenntnisse innerhalb verschiedener Aufgaben/ Anforderungen im Studium einsetzen und anwenden
- Die Studierenden sind auf das Rezipieren von und Interagieren in fachlichen Veranstaltungen auf Englisch grundlegend vorbereitet
- Die Studierenden können anlassbezogen und zielführend in englischer Sprache im eigenen Fachkontext kommunizieren und agieren
- Demnach sind die Studierenden beispielsweise in der Lage, sich zu fachlichen Themen auszutauschen und an Diskussionen teilzunehmen
- Sie können Funktion und Form verschiedener Textsorten erarbeiten und dieses Wissen in selbstproduzierten Texten kompetent anwenden
- Sie weisen Kompetenzen im Bereich der sprachlichen Mediation auf und können sowohl Global- als auch Detailinformationen aus Hör- und Lesetexten anderen klar, präzise und prägnant vermitteln

Inhalt

In den Lehrveranstaltungen ‚**Reading Skills**‘ und auch ‚**Listening Skills**‘ erfolgt eine Einführung in die Wissenschaftssprache der Informatik. Anhand von zunehmend komplexer werdenden fachlichen Inhalten werden alle sprachlichen Teilkompetenzen auf dem Niveau B2 handlungsorientiert trainiert.

In der Lehrveranstaltung ‚**Reading Skills**‘ bauen die Studierenden schwerpunktmäßig ihre Kompetenzen im Bereich Leseverstehen und in der schriftlichen Produktion aus. Mit besonderem Fokus auf die Fachsprache lesen die Studierenden authentische Texte (Lehrwerksausschnitte, Artikel, Berichte etc.) und verfassen eigene studienbezogene Texte (z.B. Beschreibungen, Kommentare, Erklärungen, Anleitungen). In diesem Zusammenhang erfolgt ebenso eine Analyse von Funktionen von Texten und Formaten, Textbausteine werden verinnerlicht und zentrale Formulierungen automatisiert. Die Studierenden lernen auch, gelesene Fachtexte adäquat und korrekt schriftlich oder mündlich wiederzugeben (Mediation).

In der Lehrveranstaltung ‚**Listening Skills**‘ bauen die Studierenden schwerpunktmäßig ihre Kompetenzen im Bereich Hör-Sehverstehen und in der mündlichen Produktion aus. Anhand von authentischen Hör(Seh)texten (Vorträgen, Vorlesungen, Diskussionen u.a.) analysieren sie Unterschiede zwischen verschiedenen Hörtexten und trainieren Strategien zum Global- und Detailverstehen. Die Studierenden lernen, sowohl unkomplizierte wissenschaftliche Texte (studienniveauekonform) zu verstehen und diese wiederzugeben (Mediation) als auch das Argumentieren zu technischen bzw. fachlichen Fragestellungen. Ebenso spielt die Beschreibung mathematischer Formeln und Prozesse eine Rolle.

Die beiden Lehrveranstaltungen haben verschiedene Kompetenzschwerpunkte, die sich gegenseitig ergänzen. Mit steigendem Studien- und Wissensniveau verstärkt sich ebenso die Progression in der englischen Sprache. Beide führen zu einem steten Auf- und Ausbau des fachlichen Vokabulars und haben zum Ziel, den Studierenden mehr Zutrauen beim Umgang mit wissenschaftlichen Texten und der eigenen sprachlichen mündlichen und schriftlichen Produktion zu geben.

Vorbereitende (im Sinne vom Flipped Classroom), vertiefende bzw. weiterführende Aufgaben in Moodle sind integrativer Bestandteil der Lehrveranstaltungen.

Lehrformen

Gruppenarbeiten, Sprachmittlung, Flipped Classroom, Blended Learning

Prüfungsformen**Semesterbegleitende Leistungsüberprüfung:****Zu a) English for Computer Science I: Reading Skills**

eine Sammlung von Kursleistungen v.a. in den Teilkompetenzen Leseverstehen und Schreiben (z.B. Leseverstehentests, Erarbeitung schriftlicher Texte, Sprachmittlungsaufgaben (reading-into-writing))

Zu b) English for Computer Science I: Listening Skills

eine Sammlung von Kursleistungen v.a. in den Teilkompetenzen Hörverstehen und Sprechen (z.B. Hörverstehentests, Aufnahmen eigener Sprachproduktion, Gruppenprojekte)

Die Modulnote setzt sich aus dem Durchschnitt der Abschlussnoten der Moduleile a) und b) im Verhältnis von 50% : 50% zusammen. Beide Moduleile müssen mit mindestens einer 4,0 bestanden sein. Für die Berechnung der Abschlussnoten und der Modulnote wird die Skala des ZFA verwendet.

Ein nicht bestandener Moduleil muss wiederholt werden, d.h. die Lehrveranstaltung muss erneut belegt und erfolgreich abgeschlossen werden.

Voraussetzungen für die Vergabe von Credits

Zu a) und b)

- Anwesenheit von 75 Prozent
- erfolgreicher Abschluss aller semesterbegleitenden Leistungen

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

6/158: B.Sc. Informatik [PO 22]

6/165: B.Sc. Informatik [PO 20]

Titel des Moduls: Mathematik 2 Mathematics 2					
Modul-Nr./Code	Credits 9 CP	Workload 270 h	Semester 2	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Mathematik 2 - Algorithmische Mathematik			Kontaktzeit 105 h	Selbststudium 180 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Christian Stump Lehrende: Prof. Dr. Christian Stump					
Verwendung des Moduls B.Sc. Informatik B.Sc. IT-Sicherheit					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen Studierende grundlegende Begriffe, Beweismethoden und Algorithmen aus der elementaren Zahlentheorie • können Studierende die Beweistechniken selbstständig anwenden und mathematische Sachverhalte darstellen • kennen Studierende erste Sätze und Methoden aus der Kombinatorik und insbesondere aus der Graphentheorie und verstehen deren strukturelle Eigenschaften • kennen Studierende erste fundamentale Algorithmen aus der Zahlentheorie und der Kombinatorik, können diese formalisieren, selbstständig implementieren sowie deren Laufzeiten analysieren 					
Inhalt Diese Lehrveranstaltung behandelt die folgenden Themen: - Euklidischer Algorithmus, Gruppen-, Ring-, Körperaxiome, Symmetriegruppen, Polynomarithmetik, formale Potenzreihen, modulare Arithmetik, Lemma von Bezout, Kleiner Satz von Fermat, diskreter Logarithmus, RSA-Verschlüsselungsverfahren, Primzahltests, Chinesischer Restesatz, p-adische Brüche, Newton-Verfahren, Asymptotische Notation durch Landausymbole, Binomialkoeffizienten, Rekursionsgleichungen, Erzeugendefunktionen, Prinzip der Inklusion-Exklusion, Vier-FarbenProblem, Dijkstra-Algorithmus, Satz von Cayley, Hamiltonkreise, Google PageRank Algorithmus, Satz von Perron-Frobenius. Konkrete Algorithmen werden in Computeralgebra-Systemen implementiert.					
Lehrformen Vorlesung mit Übungen					
Prüfungsformen Schriftliche Modulabschlussprüfung über 180 Minuten					
Voraussetzungen für die Vergabe von Credits Bestandene Modulabschlussprüfung und erfolgreiche Teilnahme an den praktischen Übungen am Rechner					

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

9/165: B.Sc. Informatik [PO 22]

9/158: B.Sc. Informatik [PO 20]

9/150: B.Sc. IT-Sicherheit [PO 22]

9/149: B.Sc. IT-Sicherheit [PO 20]

Titel des Moduls: Mathematik 2 Mathematics 2					
Modul-Nr./Code	Credits 9 CP	Workload 270 h	Semester 2	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Logik in der Informatik (211061) und Wahrscheinlichkeit in der Informatik (212028)			Kontaktzeit 90 h	Selbststudium 180 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Thomas Zeume, Prof. Alexander May Lehrende: Prof. Dr. Thomas Zeume, Prof. Christoph Thäle (SS 25)					
Verwendung des Moduls B.Sc. Informatik, B.Sc. IT-Sicherheit					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen die Studierenden lernen, wie sich Problemstellungen durch geeignete logische Systeme modellieren lassen, • beherrschen die Studierenden Syntax und Semantik verschiedener logischer Systeme und können diese nutzen, • kennen die Studierenden einige klassische logische Kalküle und können diese durchführen, • haben die Studierenden ein grundlegendes Verständnis für die Logik-Programmierung entwickelt und können insbesondere einfache Sachverhalte durch Prolog-Programme ausdrücken. • kennen die Studierende grundlegende Begrifflichkeiten der Wahrscheinlichkeitstheorie wie Zufallsvariablen, sowohl im Diskreten als auch im Kontinuierlichen, und können diese sicher anwenden. • beherrschen Studierende die Bestimmung von Momenten, wie dem Erwartungswert und der Varianz, im Diskreten mit Hilfe der Technik der z-Transformation. • erlernen Studierende verschiedene für die Informatik bedeutende Verteilungen wie die Binomial-, Geometrische, Poisson, Uniforme, Exponential- und Normal-Verteilung und können diese sicher anwenden. • sind Studierende in der Lage, die vorgestellten Techniken mit Hilfe der Computer-Algebra Sage in Python zu implementieren. 					
Inhalt Logische Methoden spielen in vielen modernen Anwendungen der Informatik eine wichtige Rolle und der vielfältige Einsatz von Zufallsbits ist grundlegend für die Entwicklung effizienter Algorithmen. Aus Datenbanken werden relevante Informationen mit Hilfe auf Logik basierender Anfragesprachen extrahiert; die formale Verifikation von Software und Hardware basiert auf logischen Spezifikationsprachen und Algorithmen für diese; und Methoden für das automatisierte Schlussfolgern in der künstlichen Intelligenz haben ihre Grundlage in der formalen Logik. In diesem Modul werden die formalen Grundlagen von modernen Logiken behandelt, mit einem Fokus auf ihrer Anwendung in der Informatik. Neben der klassischen Aussagenlogik und Prädikatenlogik betrachten wir auch Modallogik. Für jede dieser Logiken formalisieren wir Syntax und Semantik, lernen wie sich informatische					

Szenarien in ihnen modellieren lassen, und betrachten Algorithmen und Kalküle für Unerfüllbarkeit und Folgerungsbeziehung.

Für viele zentrale Probleme der Informatik sind Lösungen überhaupt nur mit Hilfe von Zufallsbits bekannt. Oft beschleunigen Zufallsbits Algorithmen, oder erlauben eine besonders elegante Analyse der Korrektheit bzw. der Laufzeit. Das Modul verschafft einen Überblick über die vielfältigen Einsatzmöglichkeiten von Zufallsbits im Algorithmen-Design. Die Studierenden erlernen Techniken zum Einsatz von Zufallsbits in Algorithmen (sogenannte probabilistische Algorithmen), sowohl bei der Korrektheits- als auch bei der Laufzeit-Analyse, und implementieren diese in einem Computeralgebra-System.

Lehrformen

Hörsaalvorlesung mit Medienunterstützung, Tutorien als seminaristischer Unterricht, zusätzlich Selbststudium mit ergänzend bereitgestellten Materialien und Aufgaben

Prüfungsformen

Die Modul setzt sich aus 2 schriftlichen Teilprüfungen zusammen:

Teilprüfung 1: Klausur zu Logik über 90 Minuten

Teilprüfung 2: Klausur zu Wahrscheinlichkeit in der Informatik über 90 Minuten

Voraussetzungen für die Vergabe von Credits

Um das Modul abzuschließen, müssen beide Teilprüfungen bestanden werden.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

9/150: B.Sc. IT-Sicherheit/Informationstechnik

9/158: B.Sc. Informatik

Titel des Moduls: Informatik 2 Computer Science 2					
Modul-Nr./Code	Credits 8 CP	Workload 240 h	Semester 2	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Informatik 2 - Algorithmen und Datenstrukturen (211002)			Kontaktzeit 90 h	Selbststudium 150 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Maike Buchin Lehrende: Prof. Maike Buchin					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik B.Sc. IT-Sicherheit/ Informationstechnik					
Vorkenntnisse Inhalte der Module Informatik 1 und Mathematik 1 bzw. Höhere Mathematik 1, insbesondere Programmieren und lineare Algebra					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls: <ul style="list-style-type: none"> • können Studierende Algorithmen formal beschreiben und deren Korrektheit beweisen • können Studierende die Laufzeit und den Speicherbedarf von Algorithmen und Datenstrukturen analysieren und bewerten • kennen Studierende grundlegende Datenstrukturen • kennen Studierende grundlegende Schemata zum Entwurf von Algorithmen sind Studierende in der Lage, Algorithmen und Datenstrukturen für spezifische Probleme zu entwickeln • haben die Studierenden die Grundlagen der Programmiersprache Python kennengelernt 					
Inhalt Die Vorlesung gibt einen systematischen Überblick über den Entwurf und die Analyse von Algorithmen und Datenstrukturen. Dazu werden zunächst grundlegende Methoden der Analyse (insbesondere Korrektheit, Laufzeit und Speicherbedarf) von Algorithmen vorgestellt. Anschließend werden einige Algorithmen zum Sortieren und Suchen analysiert. Ebenfalls werden verschiedene grundlegende Datenstrukturen (Listen, Felder, Suchbäume und Heaps) vorgestellt. Schließlich werden Graphen betrachtet, und zwar ihre Darstellung und diverse Algorithmen auf Graphen (Durchläufe, kürzeste Wege, minimale Spannbäume). In den Übungen lernen die Studierenden sowohl die theoretische Analyse von Algorithmen und Datenstrukturen als auch deren praktische Umsetzung in eine moderne Programmiersprache (z.B. Python).					
Lehrformen Hörsaalvorlesung mit Medienunterstützung und theoretische sowie praktische Übungen am Rechner					
Prüfungsformen Schriftliche Modulabschlussprüfung über 150 Minuten					

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

8/158: B.Sc. Informatik [PO 22]

8/165: B.Sc. Informatik [PO 20]

8/168: B.Sc. Angewandte Informatik [PO 22]

8/170: B.Sc. Angewandte Informatik [PO 20]

8/150: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

8/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

Titel des Moduls: Programmierung und Programmiersprachen Programming and programming languages					
Modul-Nr./Code	Credits 6 CP	Workload 180 h	Semester 2	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Programmierung und Programmiersprachen: Vorlesung und Übung (211053)			Kontaktzeit 60 h	Selbststudium 120 h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Nils Jansen Lehrende: Prof. Dr. Nils Jansen					
Verwendung des Moduls B. Sc. Informatik B.Sc. Angewandte Informatik					
Vorkenntnisse Inhalte des Moduls Informatik 1					
Lernziele (learning outcomes) Ziel dieses Kurses ist es, ein umfassendes Verständnis der wichtigsten Programmierparadigmen zu vermitteln: Imperative und objektorientierte Programmierung, funktionale Programmierung und logische Programmierung. Der Kurs stellt den Studierenden praktische und moderne Programmierübungen bereit, die ihnen bei der Lösung realer Programmierprobleme helfen. Der Schwerpunkt des Kurses liegt auf der Entwicklung, Analyse und Verifizierung von Code, der solche Probleme effizient löst. Darüber hinaus erlernen die Studierenden im Rahmen der Programmierübungen den Umgang mit modernen Softwareentwicklungstools wie IDEs und DevOps-Paketen. Nach erfolgreichem Abschluss des Kurses sind die Studierenden dazu in der Lage <ul style="list-style-type: none"> • reale Programmierprobleme zur objektorientierten Programmierung zu abstrahieren • die Unterschiede und individuellen Vorteile der Programmierparadigmen der Objektorientierung, der funktionalen Programmierung und der logischen Programmierung verstehen • in JAVA zu programmieren • einfache Programme in Haskell und Prolog zu schreiben • Moderne IDEs und DevOps-Pakete zu nutzen 					
Inhalt <ul style="list-style-type: none"> • Grundlagen der Programmierung in JAVA • Objektorientierte Programmierung in JAVA • GUI-Programmierung in JAVA • Entwurfsmuster in JAVA • Grundlagen der funktionalen Programmierung in Haskell • Grundlagen der Logikprogrammierung in Prolog 					
Lehrformen Vorlesungen, digitale Übungen, und Übungsstunden					
Prüfungsformen Schriftliche Modulabschlussprüfung (120 Minuten)					
Voraussetzungen für die Vergabe von Credits Bestandene schriftliche Modulabschlussprüfung					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 6/158: B.Sc. Informatik					

Titel des Moduls: Technische Informatik 2					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Digitaltechnik für ITS und Informatik (211014, ab SoSe 23) Digitaltechnik (141304, bis SoSe 22)			Kontaktzeit 60 h	Selbststudium 90	Gruppengröße Studierende
Unterrichtssprache Deutsch, Vorlesungs- und Übungsfolien auf Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Tim Güneysu Lehrende: Prof. Dr. Tim Güneysu					
Verwendung des Moduls B.Sc. Informatik B.Sc. IT-Sicherheit					
Vorkenntnisse Inhalte des Moduls Mathematik 1 - Grundlagen. Vorausgesetzt wird ein generelles Interesse an technischen Systemen, die Fähigkeit zu strukturieren, algorithmischem Denken sowie die Fähigkeit zum Erfassen von komplexen Abhängigkeiten und Interaktionsmustern.					
Lernziele (learning outcomes) Nach erfolgreichem Abschluss des Moduls haben die Studierenden umfassende Kenntnisse in Boolescher Algebra, Struktur und Funktionsweise grundlegender digitaler Schaltungen, Kostenoptimierung digitaler Funktionsgruppen, Techniken zur takt-synchronen Verarbeitung von Daten, Kodierung und Verarbeitung von Daten, Struktur und Funktionsweise solcher Grundfunktionalitäten, die insbesondere in Mikroprozessorarchitekturen zentrale Bestandteile sind, erworben. Die Studierenden sind in der Lage, grundlegende Schaltungskonzepte digitaler Logik- und Funktionsblöcke zu verstehen, ihr Zusammenspiel zu analysieren, die Funktionalität zu bewerten und einfache Blöcke selbst zu entwickeln. Weiterhin werden die Bewertung und Entwicklung von mehrstufigen kombinatorischen Logikblöcken sowie von Finite State Machines (FSMs) behandelt. Die Studierenden erlernen die Hardwarebeschreibungssprache Verilog, und zu jedem Thema der Vorlesung werden Verilog-Beispiele gegeben. Die Vorlesung befasst sich ausschließlich mit (takt-)synchronen Schaltungen.					
Inhalt Der Kurs gibt einen systematischen Überblick über die folgenden Themen: Boolesche Algebra, Realisierung boolescher Funktionen, Minimierung boolescher Funktionen, Multiplexer, Kodierer, Dekodierer, fehlererkennende und fehlerkorrigierende Codes, Addierer, Subtrahierer, Multiplizierer, Hardwarebeschreibungssprache Verilog, Speicherelemente (Flipflops), sequentielle Schaltungen, Zähler, Schieberegister, RAM, Finite State Machines (FSMs), Timing-Analyse sequentieller Schaltungen, und kurzer Überblick über FPGAs.					
Lehrformen Die Vorlesung wird als seminaristischer Unterricht abgehalten, die Übungen entweder am Rechner oder mit Stift und Papier.					
Prüfungsformen Schriftliche Modulabschlussprüfung über 120 Minuten					

Voraussetzungen für die Vergabe von Credits

Bestandene Modulabschlussprüfung und erfolgreiche Teilnahme an Übungen

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/150: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

Titel des Moduls: English for Computer Science 2**English for Computer Science 2**

Modul-Nr./Code	Credits	Workload	Semester	Turnus	Dauer
	6 CP	180 h	3		2 Semester
Lehrveranstaltungen a) English for Computer Science II: Presenting in English Sprachkurs mit integrierten Übungen (251217-21) (2 SWS, Wintersemester) b) English for Computer Science II: Writing in English Sprachkurs mit integrierten Übungen (251216-19) (2 SWS, Sommersemester)			Kontaktzeit 60 h	Selbststudium	Gruppengröße 25 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Bestandenes Modul <i>English for Computer Science I</i> Achtung: Diese Lehrveranstaltungen werden auf zwei verschiedenen Kompetenzniveaus angeboten: 1. Intermediate (B1/B2-B2) 2. Advanced (B2/C1-C2) Die Zuteilung zu den zwei Kompetenzniveaus im 2. Modul sollte mit der Zuteilung im 1. Modul übereinstimmen. Der Aufstieg in die höhere Kompetenzgruppe wäre nach einer sehr guten Leistung (Note) im ersten Modul möglich. Die Moduleile a) und b) können unabhängig voneinander besucht werden. Es empfiehlt sich jedoch, mit dem Besuch von <i>Presenting in English</i> anzufangen.		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Frau Melissa Oldfield-Mariano Lehrende: Dozent*innen des Zentrums für Fremdsprachenausbildung (ZFA) der Ruhr-Universität Bochum, Melissa Oldfield-Mariano					
Verwendung des Moduls Dieses Modul wurde exklusiv für diesen Bachelor-Studiengang konzipiert. Daher ist eine Öffnung dieser Veranstaltung für Studierende anderer Studiengänge nicht vorgesehen.					
Vorkenntnisse Das Modul English for Computer Science I sollte erfolgreich abgeschlossen und das Niveau B2 voll erreicht sein.					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls - können die Studierenden Fachbegriffe gezielt und effektiv in Präsentationen und schriftlichen Texten anwenden - sind die Studierenden in der Lage, ihr Wissen über ein Fachthema bzw. eigene Erkenntnisse sicher zu vermitteln und spontan dazu Fragen zu beantworten - sind die Studierenden in der Lage, in einer fachlichen Diskussion persönliche Standpunkte und Meinungen zu äußern und zu erfragen, Stellung zu beziehen, Hypothesen aufzustellen, Argumente und Gegenargumente zu formulieren sowie Vor- und Nachteile aufzuzeigen					

Zu a) English for Computer Science II: Presenting in English

Darüber hinaus beherrschen die Studierenden alle erforderlichen Kompetenzen, um eine (unkomplizierte) fachliche Präsentation im Englischen halten zu können, d.h. in diesem Zusammenhang:

- Sie können aus verschiedenen Quellen Informationen für ihre Präsentation zusammentragen
- Sie können diese Informationen fachkonform und den Konventionen der englischen Sprache angemessen aufbereiten
- Sie sind in der Lage, die Vortragssprache (sprachliche Mittel) korrekt einzusetzen und die Präsentation publikumsadäquat vorzutragen
- Sie können auf Fragen zur Präsentation spontan eingehen
- Sie können mit fachrelevanten Vorträgen sicher umgehen

Zu b) English for Computer Science II: Writing in English

Die Studierenden beherrschen außerdem alle erforderlichen Kompetenzen, um (unkomplizierte) Fachtexte im Englischen schreiben zu können, d.h. in diesem Zusammenhang:

- Sie können aus verschiedenen Quellen Informationen für ihren Text zusammentragen
- Sie können diese Informationen fachkonform und den Konventionen der englischen Sprache angemessen aufbereiten
- Sie können ihre eigenen Texte überprüfen und über Verfasstes reflektieren.
- Sie können anderen Studierenden Feedback geben (mündlich und schriftlich) und fremde Texte evaluieren (Peer-Review)

Sie können mit fachrelevanten Textsorten sicher umgehen

Inhalt

Zu a) Presenting in English

Der Fokus dieser Veranstaltung liegt auf der mündlichen Kompetenz, hier insbesondere im Schwerpunktbereich des Präsentierens. Im Vordergrund steht eine Erweiterung des Wortschatzes und eine Verbesserung der fachlichen Ausdrucksweise im mündlichen Kontext. Authentische Vorträge und Vorlesungen aus dem Fach Informatik liefern sprachlichen Input und dienen als Grundlage für Analysen und als Beispiele für die Erstellung der eigenen Präsentation.

Innerhalb einer Projektarbeit erhalten die Studierenden die Gelegenheit, die fach-konformen lexikalischen und strukturellen Aspekte kennenzulernen und anzuwenden. Dabei wird das Erstellen und Durchführen von Präsentationen für unterschiedliche Adressaten eine Rolle spielen. Durch die regelmäßige Praxis und die intensive Arbeit mit Präsentationen gewinnen die Kursteilnehmenden eine größere Vertrautheit und Sicherheit im Umgang mit den fachrelevanten Vorträgen, entwickeln Präsentationsstrategien und erwerben Werkzeuge/Kompetenzen für eine effektive mündliche Kommunikation in der englischen Sprache in akademischen und beruflichen Kontexten.

Zu b) Writing in English

Der Fokus dieser Veranstaltung liegt auf der schriftlichen Kompetenz sowie einer Erweiterung des Wortschatzes und einer Verbesserung der Ausdrucksweise im fachlichen Kontext. Authentische Texte aus dem Fach Informatik liefern sprachlichen Input und dienen als Grundlage für Textanalysen und als Beispiele für die eigene schriftliche Produktion. Dadurch erhalten die Kursteilnehmenden die Gelegenheit, die fachkonformen lexikalischen und strukturellen Aspekte kennenzulernen und anzuwenden. Auch das Schreiben in verschiedenen Formaten und für unterschiedliche Adressaten wird eine Rolle spielen.

Zu den Aufgaben wird das wöchentliche Verfassen von Texten von einem vorgegebenen Umfang gehören, zu

denen die Studierenden Feedback erhalten und die sie überarbeiten werden. Durch die regelmäßige Praxis und Anwendung der geschriebenen Sprache gewinnen die Kursteilnehmenden eine größere Vertrautheit und Sicherheit im Umgang mit den fachrelevanten Textsorten, entwickeln Schreibstrategien und erwerben Werkzeuge/ Kompetenzen für einen effektiven schriftlichen Umgang mit der englischen Sprache in akademischen und beruflichen Kontexten.

Neben einer Steigerung der individuellen schriftlichen Fähigkeiten, arbeiten die Studierenden an Techniken des gegenseitigen Feedbacks (Peer-Review), erhöhen durch gezielte Übungen ihre schriftliche Ausdrucksfähigkeit im Englischen und werden an studiennahe Aufgabenformate herangeführt. In einem Portfolio werden die unterschiedlichen Schreibprodukte gesammelt, die den Lernprozess dokumentieren und dabei individuelle Schwerpunkte sowie den Lernerfolg widerspiegeln.

Vorbereitende (im Sinne vom Flipped Classroom), vertiefende bzw. weiterführende Aufgaben in Moodle sind integrativer Bestandteil beider Lehrveranstaltungen.

Lehrformen

Gruppenarbeiten, Projektarbeit, Sprachmittlung, Flipped Classroom, Blended Learning

Prüfungsformen

Semesterbegleitende Leistungsüberprüfung:

Zu a) Presenting in English: wöchentliche Aufgaben zur Vorbereitung und abschließendes Abhalten einer Präsentation im vorgegebenen Format

Zu b) Writing in English: eine Sammlung von schriftlichen Aufgaben (Texte) unterschiedlicher Formate

Die Modulnote setzt sich aus dem Durchschnitt der Abschlussnoten der Modulteile a) und b) im Verhältnis von 50% : 50% zusammen. Beide Modulteile müssen mit mindestens einer 4,0 bestanden sein. Für die Berechnung der Abschlussnoten und der Modulnote wird die Skala des ZFA verwendet.

Ein nicht bestandener Modulteil muss wiederholt werden, d.h. die Lehrveranstaltung muss erneut belegt und erfolgreich abgeschlossen werden.

Voraussetzungen für die Vergabe von Credits

Zu a) und b)

- Anwesenheit von 75 Prozent

Erfolgreicher Abschluss aller semesterbegleitenden Leistungen (siehe Prüfungsformen)

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

6/158: B.Sc. Informatik [PO 22]

6/165: B.Sc. Informatik [PO 20]

Titel des Moduls: Mathematik 3 Mathematics 3					
Modul-Nr./Code	Credits 9 CP	Workload 270 h	Semester 3	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Logik in der Informatik (212013) und Probabilistische Algorithmen (212028)			Kontaktzeit 90 h	Selbststudium 180 h	Gruppengröße 100 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Thomas Zeume Lehrende: Prof. Dr. Thomas Zeume, Prof. Dr. Alex May					
Verwendung des Moduls B.Sc. Informatik					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen die Studierenden lernen wie sich Problemstellungen durch geeignete logische Systeme modellieren lassen, • beherrschen die Studierenden Syntax und Semantik verschiedener logischer Systeme und können diese nutzen, • kennen die Studierenden einige klassische logische Kalküle und und können diese durchführen, • haben die Studierenden ein grundlegendes Verständnis für die Logik-Programmierung entwickelt und können insbesondere einfache Sachverhalte durch Prolog-Programme ausdrücken. • kennen die Studierende grundlegende Begrifflichkeiten wie diskrete Zufallsvariablen, Momente und Verteilungen, • beherrschen Studierende die Abschätzung von Laufzeiten von Algorithmen mit Hilfe von Chernoff-Schranken, • können sie zufällige Walks auf Graphen modellieren und mit Hilfe von Markovketten analysieren, • erlernen Studierende den Entropie-Begriff und können Daten optimal verlustfrei komprimieren, • beherrschen Studenten die Analyse und Anwendung universeller Hashfunktionen, • sind Studierende in der Lage, die vorgestellten Techniken mit Hilfe der Computer-Algebra Sage zu implementieren. 					
Inhalt Logische Methoden spielen in vielen modernen Anwendungen der Informatik eine wichtige Rolle und der vielfältige Einsatz von Zufallsbits ist grundlegend für die Entwicklung effizienter Algorithmen. Aus Datenbanken werden relevante Informationen mit Hilfe auf Logik basierender Anfragesprachen extrahiert; die formale Verifikation von Software und Hardware basiert auf logischen Spezifikationsprachen und Algorithmen für diese; und Methoden für das automatisierte Schlussfolgern in der künstlichen Intelligenz haben ihre Grundlage in der formalen Logik. In diesem Modul werden die formalen Grundlagen von modernen Logiken behandelt, mit einem Fokus auf ihrer Anwendung in der Informatik. Neben der klassischen Aussagenlogik und Prädikatenlogik betrachten wir auch Modallogik. Für jede dieser Logiken formalisieren wir Syntax und Semantik, lernen wie sich informatische Szenarien in ihnen modellieren lassen, und betrachten Algorithmen und Kalküle für Unerfüllbarkeit und					

Folgerungsbeziehung.

Für viele zentrale Probleme der Informatik sind Lösungen überhaupt nur mit Hilfe von Zufallsbits bekannt. Oft beschleunigen Zufallsbits Algorithmen, oder erlauben eine besonders elegante Analyse der Korrektheit bzw. der Laufzeit. Das Modul verschafft einen Überblick über die vielfältigen Einsatzmöglichkeiten von Zufallsbits im Algorithmen-Design. Die Studierenden erlernen Techniken zum Einsatz von Zufallsbits in Algorithmen (sogenannte probabilistische Algorithmen), sowohl bei der Korrektheits- als auch bei der Laufzeit-Analyse, und implementieren diese in einem Computeralgebra-System.

Lehrformen

Hörsaalvorlesung mit Medienunterstützung, Tutorien als seminaristischer Unterricht, zusätzlich Selbststudium mit ergänzend bereitgestellten Materialien und Aufgaben

Prüfungsformen

Schriftliche Modulabschlussprüfung über 180 Minuten

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

9/158: B.Sc. Informatik [PO 22]

9/170: B.Sc. Informatik [PO 20]

Titel des Moduls: Mathematik 3 Mathematics 3					
Modul-Nr./Code	Credits 9 CP	Workload 270 h	Semester 3	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Mathematik 3 (212009)			Kontaktzeit 90 h	Selbststudium 180 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Dr. Christof Beierle Lehrende: Dr. Christof Beierle					
Verwendung des Moduls B.Sc. Informatik					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen Studierende grundlegende Begriffe und Techniken der reellen Analysis in einer Variablen, sowie Grundzüge der mehrdimensionalen Analysis • verstehen die Studierenden insbesondere den Konvergenzbegriff, der Begriff der Differenzierbarkeit, sowie den Begriff des Riemann-Integrals und Anwendungen dieser • können Studierende die erlernten Techniken selbstständig anwenden. 					
Inhalt Dieses Modul vermittelt die Grundlagen in der reellen Analysis in einer Variablen, sowie eine Einführung in die mehrdimensionale Differenzialrechnung. Behandelte Themen sind: <ul style="list-style-type: none"> • Axiomatische Einführung der reellen Zahlen • Folgen und Reihen, Konvergenzbegriff, Vollständigkeit der reellen Zahlen • b-adische Darstellung reeller Zahlen • Grenzwerte und Stetigkeit reellwertiger Funktionen • Komplexe Zahlen • Exponentialfunktion und Winkelfunktionen • Umkehrfunktionen • Differenzierbarkeit, Ableitungen, Ableitungsregeln • Extremwerte • Stammfunktionen und Riemann-Integrale • Hauptsatz der Differential- und Integralrechnung • Skalarwertige Funktionen in mehreren Veränderlichen, partielle und totale Ableitungen, Richtungsableitung und Gradient • - Gradientenabstiegsverfahren 					
Lehrformen Vorlesung mit Übung					
Prüfungsformen Schriftliche Modulabschlussprüfung über 120 Minuten					

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

9/158: B.Sc. Informatik [PO 22]

9/170: B.Sc. Informatik [PO 20]

Titel des Moduls: Informatik 3 Computer Science 3					
Modul-Nr./Code	Credits 8 CP	Workload 240 h	Semester 3	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Informatik 3 - Theoretische Informatik (212002)			Kontaktzeit 90 h	Selbststudium 150 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Thomas Zeume Lehrende: Prof. Dr. Thomas Zeume					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik B.Sc. IT-Sicherheit/ Informationstechnik					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • beherrschen die Studierenden den professionellen Umgang mit Berechnungsmodellen und ihren Beziehungen zu Sprachklassen. Dazu gehört die intellektuelle und methodische Fähigkeit, den Nachweis der Zugehörigkeit bzw. Nichtzugehörigkeit zu einer solchen Klasse zu führen. • ist durch Einüben von Beweistechniken wie wechselseitige Simulation oder berechenbare Reduktionen bei den Studierenden die Einsicht gereift, dass an der Oberfläche verschieden aussehende Konzepte im Kern identisch sein können. Zudem erlaubt dies den Studierenden, neue Anwendungsprobleme selbstständig zu klassifizieren. • haben die Studierenden mit der Turingmaschine ein einfach handhabbares Rechnermodell erlernt, das ihnen fortan als Abstraktion für alle möglichen Rechner dient. • haben die Studierenden fundamentale Einsichten erlangt, welche Probleme mithilfe von Rechnern effizient entschieden, zum Teil entschieden oder prinzipiell nicht entschieden werden können. Dadurch erlangen Sie ein tieferes Verständnis der Komplexität von Berechnungsproblemen. 					
Inhalt Die Lehrveranstaltung gibt einen systematischen Überblick über die folgenden Themengebiete: <ul style="list-style-type: none"> • Endliche Automaten und reguläre Ausdrücke • Kellerautomaten und kontextfreie Grammatiken • Turingmaschinen und Entscheidbarkeit • Nichtdeterminismus und NP-Vollständigkeitstheorie 					
Lehrformen Hörsaalvorlesung mit Medienunterstützung und Übungen, bei denen die vorgestellten Konzepte und Techniken praktisch umgesetzt werden, teilweise mit Rechnerübungen.					
Prüfungsformen Schriftliche Modulabschlussprüfung (180 Minuten)					

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

8/165: B.Sc. Informatik [PO 22]

8/158: B.Sc. Informatik [PO 22]

8/168: B.Sc. Angewandte Informatik [PO 22]

8/170: B.Sc. Angewandte Informatik [PO 20]

8/150: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

8/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

Titel des Moduls: Technische Informatik 3					
Modul-Nr./Code	Credits 5 CP	Workload 150 Stunden	Semester 3	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Technische Informatik 3 - Hardwareprogrammierung (212003)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr.-Ing. Tim Güneysu Lehrende: Prof. Dr.-Ing. Tim Güneysu					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik					
Vorkenntnisse Inhalte der Module Informatik 1 - Programmierung, Informatik 2 - Algorithmen und Datenstrukturen, Technische Informatik 1 - Rechnerarchitektur und Technische Informatik 2 - Digitaltechnik					
Lernziele (learning outcomes) Die Studierenden sollen Kenntnisse über technische Herausforderungen bei der anwendungsbezogenen Entwicklung von eingebetteten Systemen sowie des Internet of Things erlernen. Im Vordergrund der Veranstaltung stehen die maschinennahe Programmierung sowie die problemgerechte Integration von Aktorik und Sensorik. Nach dem erfolgreichen Abschluss des Moduls - kennen Studierende das Entwicklungs- und Programmiermodell sowie spezifische Eigenschaften von Mikrocontrollern als zentrale Grundlage eines eingebetteten Systems - haben Studierende die Fähigkeit zur maschinennahen Programmierung eines ausgewählten Mikrocontrollers zur Realisierung grundlegender Steuerprozesse sowie für die umgebende Peripherie 43 - erlernen Studierende Kommunikations- und Interaktionskonzepte (z.B. serielle Kommunikation via UART, SPI, PCIe etc.) mit externen Komponenten - können Studierende nebenläufige Prozesse strukturieren, in Systemen integrieren sowie damit verbundene potenzielle Probleme verstehen - sind Studierende in der Lage, komplexe anwendungsnahe Anforderungen mittels ausgewählter Hardwarekomponenten in ein eingebettetes System zu realisieren					
Inhalt Die Entwicklung von komplexen eingebetteten Systemen in Form einer zentralen Steuereinheit mit unterschiedlicher Sensorik und Aktorik spielen in vielen Anwendungen eine zentrale Rolle. In der Veranstaltung „Technische Informatik 3“ werden die vielfältigen Aufgabentypen und Realisierungsmöglichkeiten eines eingebetteten Systems sowie dessen anwendungsnaher Entwurf und Implementierung behandelt. Ein besonderes Gewicht in der Veranstaltung wird dabei auf ein anwendungsnahes Beispiel aus der Robotik gelegt, bei der viele der genannten Eigenschaften, Programme und Peripherie entwickelt und gesamtheitlich integriert werden müssen, um die Fähigkeiten eines einfachen Robotersystems auf vorhandener Hardware nachzubilden.					
Lehrformen Vorlesung (als Folien und Tafelvortrag) und Übungen, bei denen die vorgestellten Konzepte und Techniken praktisch an unterschiedlichen Hardware-Architekturen umgesetzt werden. Die Übungen beinhalten Elemente der Gruppen- und Projektarbeit					
Prüfungsformen Schriftliche Modulabschlussprüfung über 150 Minuten					
Voraussetzungen für die Vergabe von Credits Bestandene Modulabschlussprüfung und erfolgreiche Teilnahme an den Übungen					

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

Titel des Moduls: Software Engineering Software Engineering					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester 3	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen 212000: Software Engineering			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 350 Studierende
Unterrichtssprache English			Teilnahmevoraussetzungen Programming and Programming Languages, Informatik 1		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Thorsten Berger Lehrende: Prof. Dr. Thorsten Berger					
Verwendung des Moduls					
Vorkenntnisse Imperative and Object-Oriented Programming in a statically typed programming language Theoretical Computer Science					
Lernziele (learning outcomes)					
Knowledge and understanding					
<ul style="list-style-type: none"> • Explain in detail the core activities for engineering software, including requirements engineering, software architectures and design, implementation, quality assurance (esp. testing), software process • explain underlying representation of programs (AST) and the compilation process • explain programming paradigms, such as imperative programming, generic programming, object-oriented programming, AI engineering 					
Competence and skills					
<ul style="list-style-type: none"> • elicit and define software requirements in different formalisms (e.g., textual requirements, state machine diagrams, or other behavior diagrams) • define a software architecture upon quality requirements, using architectural patterns or styles • implement software architectures, frameworks, and software modules/components • define a software engineering process based on process models, including plan-based and agile models • perform quality assurance, including designing test cases according to test-case design methods for black-box and white-box testing • create behavioral and structural models in the context of software engineering 					
Judgement and approach					
<ul style="list-style-type: none"> • identify use-cases and the potential of different SE methods and technologies for a given domain/problem • select and justify SE methods and technologies for a given domain/problem 					
Inhalt					
<p>You can write code, but can you also write software? The course provides an introduction into Software Engineering methods and tools. Covering the overall phases of software engineering, namely planning, requirements engineering, architectural design, implementation, quality assurance, and evolution and maintenance, the curriculum will walk students through modern software engineering technology that aims at building the modern software systems and products. The course attempts to be close to programming technology, while covering multiple paradigms and solutions.</p> <p>We cover:</p> <ul style="list-style-type: none"> • Introduction to software engineering, motivated by software and project failures, laws of software engineering, and optionally a guest lecture from industry 					

- Introduction into compiler construction and the underlying representation of programs (e.g., abstract syntax trees, control-flow diagrams)
- Paradigms (e.g., imperative programming, generic programming, object-oriented programming, AI engineering) and software engineering principles (e.g., modularity, cohesion/coupling)
- UML behavioral and structural diagrams for modeling software
- Requirements engineering
- Software architecture and architecture implementation
- Quality assurance, including black-box and white-box testing, mutation testing, combinatorial interaction testing
- Advanced topics, such as software product lines, model-driven software engineering or security engineering (e.g., threat modeling)

Lehrformen

The teaching of this course consists of different forms: lectures, interactive quizzes, group work, group supervision, and practical assignments.

Prüfungsformen

Written exam at the end of the course (120 minutes)

Voraussetzungen für die Vergabe von Credits

Group project and final exam passed

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

5/150: B.Sc. IT-Sicherheit/Informationstechnik [PO22]

5/149: B.Sc. IT-Sicherheit/Informationstechnik [PO20]

Titel des Moduls: Betriebssysteme Operating Systems					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester 4	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Betriebssysteme (211005)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 350 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr.-Ing. Timo Hönig Lehrende: Prof. Dr.-Ing. Timo Hönig					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik B.Sc. IT-Sicherheit/Informationstechnik					
Vorkenntnisse Grundkenntnisse der Informatik (Inhalte der Module Informatik 1 - Programmierung und Technische Informatik 1 - Rechnerarchitektur)					
Lernziele (learning outcomes) Nach dem erfolgreichen Absolvieren des Moduls <ul style="list-style-type: none"> • erlangen die Studierenden ein solides Grundverständnis von modernen Betriebssystemen, ihrer Funktion und ihrer Implementierung • sind die Studierenden in der Lage, verschiedene Aspekte eines Betriebssystems wie Prozess- und Speichermanagement zu verstehen und zu nutzen, sie können dabei verschiedene Designentscheidungen eigenständig analysieren und bewerten • sind die Studierenden in der Lage, bestimmte Aspekte eines Betriebssystems selbst zu designen und diese argumentativ zu verteidigen 					
Inhalt In diesem Modul werden die wichtigsten Grundlagen zu Betriebssystemen vorgestellt. Dazu gehören zum Beispiel: <ul style="list-style-type: none"> • Betriebssystemkonzepte • Prozesse und Threads, Interprozesskommunikation • Scheduling-Mechanismen • Speicherverwaltung, Speicherabstraktionen, Paging • Dateisysteme • Eingabe- und Ausgabeverwaltung • Algorithmen zur Vermeidung von Deadlocks • Grundlagen der Sicherheit von Betriebssystemen <p>In den letzten Wochen der Veranstaltung, abhängig vom verfügbaren Zeitfenster, werden spezielle Themen wie beispielsweise Multimedia-Betriebssysteme, Multiprozessorsysteme und Entwurf von Betriebssystemen, behandelt.</p> <p>Um den Bezug zu modernen Betriebssystemen (aktuellen Versionen von Linux, Windows und macOS) herzustellen, werden die Themen an praktischen Beispielen illustriert. Dies ermöglicht es den Studierenden, die in der Vorlesung besprochenen Themen praktisch nachzuvollziehen.</p>					

Lehrformen

Die Vorlesung wird als seminaristischer Unterricht mit Medienunterstützung abgehalten. eLearning unterstützte Hausaufgaben mit praxisnahen, am Rechner zu implementierenden Übungen werden alle zwei Wochen vergeben und in der Übungsstunde besprochen.

Prüfungsformen

Schriftliche Modulabschlussprüfung (90 Minuten)

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/170: B.Sc. Informatik [PO 20]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

5/150: B.Sc. IT-Sicherheit/Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/Informationstechnik [PO 20]

Titel des Moduls: Computernetze Computer Networks					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester 2	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Computernetze (211006)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 400 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Dr.-Ing. Christian Mainka Lehrende: Dr.-Ing. Christian Mainka					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik B.Sc. IT-Sicherheit / Informationstechnik					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen Studierende die wichtigsten Standards, die das heutige Internet verwendet. • kennen Studierende grundlegende Angriffskonzepte auf Computernetzwerke • verstehen Studierende den Zusammenhang zwischen den einzelnen Schichten eines Computernetzwerks und der darin enthaltenen Protokolle • können Studierende die wichtigsten Netzwerktools für Analysezwecke anwenden 					
Inhalt Die Vorlesung gibt eine Einführung in grundlegenden Protokollen und Anwendungen von Computernetzen. Der Schwerpunkt der Vorlesung liegt auf Standardprotokollen und -Algorithmen, wie sie in modernen Computernetzwerken (zum Beispiel im Internet) eingesetzt werden. Anhand eines Schichtenmodells werden die wichtigsten Grundlagen nach dem Top-Down Ansatz vorgestellt und analysiert. Dazu gehören zum Beispiel auf der obersten Schicht DNS und HTTPS im Application Layer; TCP und UDP im Transport Layer; IPv4/IPv6 und Routing Algorithmen im Network Layer; sowie MAC und ARP im untersten Link Layer. Neben der reinen Funktionsweise dieser Standards werden Sicherheitsaspekte auf allen Schichten betrachtet. Ergänzend zur Vorlesung werden Übungsaufgaben über die eLearning Plattform Moodle gestellt und in der Übungsstunde besprochen. Weiterhin wird in jeder Übung ein "Tool der Woche" vorgestellt. Dabei handelt es sich jeweils um eine spezielle Software, die man als "Netzwerker" unbedingt kennen sollte (z.B. traceroute, nmap, ...). Alle besprochenen Tools sind frei verfügbar und werden den Studenten als eine Lernplattform (virtuelle Maschine) zur Verfügung gestellt. Als Primärliteratur wird "Computernetzwerke: Der Top-Down Ansatz" von Kurose und Ross (Pearson Verlag) verwendet.					
Lehrformen Moodle-Unterstützte Hausaufgaben mit praxisnahen, computerunterstützten Übungen. Tool-der-Woche: Vorstellung, Einarbeitung, und Verwendung von Netzwerkrelevanten Computer analysetools.					

Prüfungsformen

schriftliche Modulabschlussprüfung von 120 min

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

Titel des Moduls: Einführung in die künstliche Intelligenz Introduction to Artificial Intelligence					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Introduction to Artificial Intelligence (211045)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 250 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Sen Cheng Lehrende: Prof. Dr. Laurenz Wiskott, Prof. Dr. Tobias Glasmachers, Prof. Dr. Sen Cheng, Prof. Dr. Gregor Schöner, Prof. Dr. Nils Jansen Prof. Dr. Setareh Maghsudi Prof. Dr. Christian Straßer					
Verwendung des Moduls B.Sc. Informatik (Pflichtmodul) B.Sc. Angewandte Informatik (Pflichtmodul) B.Sc. IT-Sicherheit/Informationstechnik (Wahlpflichtmodul)					
Vorkenntnisse Basic knowledge of calculus and linear algebra.					
Lernziele (learning outcomes) After successful completion of this course, students will be able to <ul style="list-style-type: none"> • summarize a number of fundamental methods in artificial intelligence, • explain their mathematical basis and algorithmic nature, • apply them to simple problems, • decide which methods are suitable for which problems, and • communicate about the all that in English. 					
Inhalt This course gives an overview over representative methods in artificial intelligence: formal logic and reasoning, classical methods of AI, probabilistic reasoning, machine learning, deep neural networks, computational neuroscience, neural dynamics, perception, natural language processing, robotics.					
Lehrformen					
Prüfungsformen Written module final exam (digital, 90 minutes)					
Voraussetzungen für die Vergabe von Credits passed written exam					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 5/158: B.Sc. Informatik [PO 22]					

5/168: B.Sc. Angewandte Informatik [PO 22]

5/150: B.Sc. IT-Sicherheit/Informationstechnik [PO 22]

Titel des Moduls: Datenbanksysteme Database Systems					
Modul-Nr./Code	Credits 7 CP	Workload 210 h	Semester 4	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Datenbanksysteme (211008)			Kontaktzeit 60 h	Selbststudium 150 h	Gruppengröße 250 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Dr. Veelasha Moonsamy Lehrende: Dr. Veelasha Moonsamy					
Verwendung des Moduls B.Sc. Informatik [PO 22] B.Sc. Angewandte Informatik [PO 22]					
Vorkenntnisse Grundkenntnisse der Informatik (Inhalte der Module Informatik 1 - Programmierung, Technische Informatik 1 - Rechnerarchitektur)					
Lernziele (learning outcomes) Nach dem erfolgreichen Absolvieren des Moduls <ul style="list-style-type: none"> • erlangen die Studierenden ein Grundverständnis von modernen Datenbanksystemen, ihrer Funktion und ihrer Implementierung • haben die Studierenden Datenmodellierungstechniken erlernt • haben die Studierenden die Semantik und die Syntax des Entity-Relationships Modells kennengelernt • kennen die Studenten das relational Datenbankmodell und die Relationale Algebra • kennen die Studierenden Anfragesprachen (z.B. SQL) und können diese nutzen • verstehen die Studierenden die Konzepte von Transaktion und Fehlerbehandlung • haben die Studierenden unterschiedliche Datenbankmanagementsysteme kennengelernt • sind die Studierenden in der Lage, neue Datenbanken zu modellieren und zu implementieren • haben die Studenten Kenntnisse über die Prozesse hinter einer Datenbankanfrage und wie diese optimiert werden kann 					
Inhalt Die Datenbanktechnologie ist eine Schlüsseltechnologie der praktischen und angewandten Informatik. Zentrales Thema dieser Veranstaltung sind die Modellierung, Aufbau und die Nutzung von Datenbanken. Folgende Themen werden behandelt: <ul style="list-style-type: none"> • Einführung in Datenbanksysteme • Entity-Relationship Modell und Verbesserungen • das relational Datenbankmodell • Relationale Algebra und Kalkül • Die Relationale Anfragesprache SQL • Datenbankprogrammierung • physische Datenorganisation • Anfragebearbeitung und Optimierung • Transaktionsverwaltung und Fehlerbehandlung 					
Lehrformen In der wöchentlichen Vorlesung werden die Lerninhalte theoretisch vermittelt. In der unterstützenden wöchentlichen Übung werden theoretische Fragestellungen sowie praktische Fragestellungen und Aufgaben am Computer bearbeitet. Die Aufgaben und Lösungen werden in der Übung gemeinsam erarbeitet und besprochen.					

Prüfungsformen

Schriftliche Modulabschlussprüfung (120 Minuten)

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

7/158: B.Sc. Informatik [PO 22]

7/168: B.Sc. Angewandte Informatik [PO 22]

Titel des Moduls: Software Engineering Praktikum Software Engineering Lab					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Software Engineering Lab (211500)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 5 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Thorsten Berger Lehrende: Prof. Dr. Thorsten Berger					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik					
Vorkenntnisse Programmiererfahrung in einer objektorientierten Programmiersprache (am besten Java), sicherer Umgang mit git					
Lernziele (learning outcomes) Nach dem Modulabschluss <ul style="list-style-type: none"> • können die Studierenden agil arbeiten nach SCRUM • wissen Studierende, wie man kleine Software-Projekte plant und können diese in Java-Android umsetzen • können Studierende ihre eigenen Ergebnisse in angemessener Form präsentieren 					
Inhalt Im Software Engineering Lab wird in kleinen Projektgruppen eine Android App mit AndroidStudio entwickelt. In der begleitenden Vorlesung werden die Grundlagen moderner Softwareentwicklung vermittelt und im Projekt praktisch umgesetzt. Die Projektgruppen arbeiten selbstorganisiert agil und werden durch den gesamten Entwicklungsprozess unterstützend angeleitet. Die Entwicklung beginnt mit der Backlog-Erstellung und endet mit einem kurzen Produkt-Pitch.					
Lehrformen Agiles Arbeiten in Projektgruppen					
Prüfungsformen Projektarbeit (semesterbegleitend) mit Zwischenmeetings, Abgaben und Abschlusspräsentation					
Voraussetzungen für die Vergabe von Credits Erfolgreich abgeschlossene Projektarbeit mit Abschlusspräsentation. Zum Bestehen des Moduls ist die Teilnahme am SCRUM-Workshop, an allen Supervisions- und Kundenterminen sowie an der Abschlusspräsentation (Prüfung) formal verpflichtend. Wenn Sie nicht teilnehmen können, informieren Sie bitte vorher Ihren Betreuer und Ihre Gruppenmitglieder und legen Sie die erforderlichen Unterlagen (z. B. ärztliches Attest bei Krankheit) vor. Außerdem müssen Sie, wie in der ersten Vorlesung erläutert, eine Gruppe bilden und diese rechtzeitig bei uns anmelden. Die Anwesenheit bei den Vorlesungen ist dringend erwünscht, aber keine Pflicht.					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)					

5/158: B.Sc. Informatik [PO 22]

5/168: B.Sc. Angewandte Informatik [PO 22]

Titel des Moduls: Introduction to Data Science Introduction to Data Science					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Introduction to Data Science (212039)			Kontaktzeit 60h (4 SWS)	Selbststudium 90 h	Gruppengröße 100 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Robert Schmidt Prof. Dr. Bilal Zafar Lehrende: Prof. Dr. Robert Schmidt Prof. Dr. Bilal Zafar					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik					
Vorkenntnisse Grundkenntnisse in Infinitesimalrechnung, linearer Algebra und Programmierung in Python					
Lernziele (learning outcomes) Am Ende dieses Kurses werden Sie mit folgenden Themen vertraut sein <ol style="list-style-type: none"> 1. Die wichtigsten modernen Methoden für datengestützte Vorhersagen 2. Methoden zur Verarbeitung, Erkundung und Visualisierung von Daten verschiedener Modalitäten wie Bild, Text und Tabellen 3. Aufbau von Proof-of-Concept-Code für die Lösung realer Data-Science-Probleme 4. Fragen rund um Vertrauen und mögliche Abhilfemaßnahmen bei Anwendungen von Data Science. 					
Inhalt Data Science ist ein sich schnell entwickelnder Bereich mit zahlreichen Anwendungsgebieten. In diesem Kurs lernen Sie grundlegende Werkzeuge von Data Science kennen. Sie werden auch mit fortgeschrittenen Methoden des Deep Learning und deren praktischen Anwendungen vertraut gemacht. Im ersten Teil des Kurses erhalten Sie eine Einführung in die grundlegenden statistischen Methoden, die Data Science zugrunde liegen. Sie erlernen auch Techniken zur Analyse und Visualisierung von Datensätzen unterschiedlicher Modalitäten wie Text, Bilder und Tabellen. Sie werden tief in die datengesteuerten Vorhersagemethoden des maschinellen Lernens und des Deep Learning eintauchen. Im letzten Teil des Kurses führen wir Sie in fortgeschrittene Themen ein, einschließlich der jüngsten Fortschritte bei der Modellierung großer Sprachen und der Nutzung datengesteuerter Entscheidungsfindung auf vertrauenswürdige Weise.					
Lehrformen Jede Sitzung des Kurses besteht aus einem Vorlesungsteil, in dem die theoretischen Konzepte vorgestellt werden, und einem praktischen Teil, in dem Sie praktische Erfahrungen mit Jupyter Notebooks sammeln können.					
Prüfungsformen Schriftliche Modulabschlussprüfung (120 Minuten)					
Voraussetzungen für die Vergabe von Credits Bestandene schriftliche Modulabschlussprüfung					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)					

5/168: B.Sc. Angewandte Informatik

5/158: B.Sc. Informatik

Titel des Moduls: Algorithmenparadigmen					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Algorithmenparadigmen (211043)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 40 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Maike Buchin Lehrende: Prof. Dr. Maike Buchin					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik					
Vorkenntnisse Inhalte aus den Modulen (Höhere) Mathematik 1 und 2 sowie aus den Modulen Informatik 1-3					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen Studierende eine Reihe von Algorithmenparadigmen • können Studierende basierend auf den Paradigmen effiziente Algorithmen für Probleme entwickeln • verstehen Studierende die Vor- und Nachteile unterschiedlicher Paradigmen 					
Inhalt In der Vorlesung betrachten wir unterschiedliche Algorithmenparadigmen, also Schemata zum Entwurf von effizienten Algorithmen. Dazu betrachten wir zunächst die bereits bekannten Paradigmen inkrementell, Teile-und-Herrsche und gierig und wenden diese auf verschiedene Probleme an. Darauf aufbauend lernen wir Dynamisches Programmieren kennen, sowie die Methoden Backtracking und Branch-and-Bound. Auch betrachten wir ein Paradigma speziell für geometrische Probleme: das Sweepline-Verfahren.					
Lehrformen Hörsaalvorlesung mit Medienunterstützung sowie Tutorien als seminaristischer Unterricht					
Prüfungsformen Schriftliche Modulabschlussprüfung (120 Minuten) oder mündliche Prüfung					
Voraussetzungen für die Vergabe von Credits Bestandene schriftliche oder mündliche Modulabschlussprüfung					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 5/170: B.Sc. Informatik [PO 22] 5/158: B.Sc. Informatik [PO 20] 5/168: B.Sc. Angewandte Informatik [PO 22]					

Titel des Moduls: Cache Attacks Cache Attacks					
Modul-Nr./Code	Credits 5 CP	Workload 150h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Cache Attcks (211026)			Kontaktzeit 60h (4 SWS)	Selbststudium 90h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Yuval Yarom Lehrende: Prof. Dr. Yuval Yarom					
Verwendung des Moduls					
Vorkenntnisse - Rechnerarchitektur - Digital Design - Starke Programmiererfahrung - Vertrautheit mit C und Assembler wird empfohlen					
Lernziele (learning outcomes) <ul style="list-style-type: none"> • Entwurf und Implementierung von Techniken zum Reverse Engineering von Prozessor-Caches • Untersuchung von Cache-Designs und Software zur Identifizierung von Seitenkanallecks • Durchführung von Cache-Angriffen • Analyse kryptographischer Algorithmen und Nutzung von Seitenkanalinformationen zur Schlüsselwiederherstellung 					
Inhalt Der Kurs befasst sich mit Sicherheitsfragen bei der gemeinsamen Nutzung von CPU-Caches. Er behandelt den Aufbau der verschiedenen Caches in modernen Prozessoren, einschließlich Daten- und Anweisungscaches, Verzweigungsvorhersage-Caches und Adressübersetzungs-Caches. Es wird gezeigt, wie man diese Komponenten zurückentwickelt, wie man Informationen über ihren Zustand ausspäht und wie man geheime Daten aus den ausgespähten Informationen ableitet. Der Kurs deckt mehrere Bereiche der Informatik ab, darunter Computerarchitektur, Betriebssysteme, Compiler sowie Ansätze des maschinellen Lernens und der Kryptographie.					
Lehrformen Kombination aus traditionellen Vorlesungen, Selbststudium und praktischen Aufgabenstellungen					
Prüfungsformen Benotete Aufgaben + Abschlussprüfung von 120 Minuten					
Voraussetzungen für die Vergabe von Credits Bestehen der Durchschnittsnote; Bestehen der Abschlussprüfung					

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik

5/150: B.Sc. IT-Sicherheit/ Informationstechnik

Titel des Moduls: Distributed Systems Algorithms Distributed Systems Algorithms					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Distributed Systems Algorithms (211004)			Kontaktzeit 45 h (3 SWS)	Selbststudium 105h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr.-Ing. Steffen Bondorf Lehrende: Prof. Dr.-Ing. Steffen Bondorf					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik					
Vorkenntnisse Allgemeines Interesse an technischen Systemen					
Lernziele (learning outcomes) Die Studierenden sollen eine breite Kenntnis über die auftretenden Herausforderungen beim Entwurf und bei der Anwendung von verteilten Computersystemen erlangen. Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen die Studierenden verschiedene Systemmodelle und Architekturen, die zum Entwurf sowie zu der Klassifizierung von verteilten Systemen dienen. Sie können verschiedene Rollen von Teilsystemen differenzieren und sie formal beschreiben • können die Studierenden vielfältige Herausforderungen beim Aufbau eines verteilten Systems identifizieren und kennen die wichtigsten Standardtechniken zum Umgang mit diesen, inklusive deren Vor- und Nachteile • können die Studierenden die Funktionsweise eines verteilt implementierten Systems anhand dessen Beschreibung verstehen und die ausgeführte Aufgabe herleiten - können die Studierenden die Fähigkeit eines verteilten Systems zur Erfüllung seiner Aufgabe beurteilen, die Quellen potenzieller Probleme identifizieren und können Verbesserungen sowie deren Integration entwerfen • sind die Studierenden in der Lage, gegebene Alternativen zur verteilten Implementierung eines Systems für eine bestimmte Aufgabe zu bewerten und begründet in eine Rangfolge zu bringen 					
Inhalt Diese Lehrveranstaltung behandelt grundlegende Architekturen und Methoden, die die Funktionsfähigkeit leistungsfähiger verteilter Computersysteme ermöglichen. Ein solches verteilte System dient der Erfüllung einer bestimmten Aufgabe unter Verwendung von mehreren unabhängigen Teilsystemen und soll dem Benutzer dabei jedoch wie ein einzelnes Computersystem erscheinen. Um dies zu erreichen, müssen die verschiedenen Teilsysteme über gemeinsames Wissen verfügen. Es treten durch die Verteilung im Vergleich zu einzelnen Systemen eine Reihe von Herausforderungen auf, die den Inhalt der Vorlesung bilden: Teilsysteme müssen sich gegenseitig auffinden können, sie müssen in der Lage sein, Nachrichten auszutauschen, Daten müssen trotz der so entstehenden Replikation über Teilsysteme hinweg konsistent gehalten werden, Fehler in Teilsystemen müssen tolerierbar sein und die Ressourcen des Gesamtsystems sollen möglichst effizient genutzt werden, sodass die gegebene Aufgabe performant erfüllt wird. All diese Komponenten und Aspekte finden sich in modernen, Internetbasierten Systemen wieder. Sie garantieren die Funktionsfähigkeit von Diensten wie das World Wide Web, E-Mail oder File-Sharing.					

Lehrformen

Die Vorlesung wird als seminaristischer Unterricht abgehalten, die praktischen Übungen am Rechner werden zudem weitere Lehrformen wie Gruppen- und Projektarbeit beinhalten

Prüfungsformen

Schriftliche Modulabschlussprüfung über 90 Minuten

Voraussetzungen für die Vergabe von Credits

Bestandene Modulabschlussprüfung und erfolgreiche Teilnahme an Übungen

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/168: B.Sc. Angewandte Informatik

5/158: B.Sc. Informatik

Titel des Moduls: Einführung in die Kryptographie 1 Introduction to Cryptography 1					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Einführung in die Kryptographie 1 (212010)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 300 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr.-Ing. Christof Paar Lehrende: Prof. Dr.-Ing. Christof Paar					
Verwendung des Moduls B.Sc. IT-Sicherheit B.Sc. Informatik B.Sc. Angewandte Informatik M.Sc. IT-Sicherheit/ Netze und Systeme					
Vorkenntnisse Fähigkeit zum abstrakten und logischen Denken.					
Lernziele (learning outcomes) Nach erfolgreichem Abschluss des Moduls verfügen die Studierenden über Kenntnisse der symmetrischen und asymmetrischen kryptographischer Verfahren. Sie können entscheiden, unter welchen Bedingungen man in der Praxis bestimmte Chiffren einsetzt und wie die Sicherheitsparameter zu wählen sind. Sie sind mit den mathematischen Grundlagen, auf denen aktuelle Kryptoverfahren beruhen, vertraut. Durch die Beschreibung ausgewählter praxisrelevanter Chiffren, wie z. B. des AES- oder RSA-Algorithmus sowie des Diffie-Hellman-Schlüsselaustauschs, erreichen die Studierenden zudem ein algorithmisches und technisches Verständnis für moderne Kryptographie. Sie sind in der Lage argumentativ eine bestimmte Lösung zu verteidigen. Die Vorlesungen werden zusätzlich auch als Videos in Deutsch und Englisch angeboten. Durch das zweisprachige E-Learning-Angebot können die Studierenden Sprachkompetenzen in der Wissenschaftssprache Englisch erwerben.					
Inhalt Das Modul bietet eine Einführung in die moderne angewandte Kryptographie und Grundlagen der IT-Sicherheit. Der Fokus liegt auf kryptographischen Verfahren, der hierfür notwendigen Mathematik sowie dem Zusammenspiel von Kryptographie und IT-Sicherheit. Es werden viele für die Anwendung relevante symmetrische und asymmetrische kryptographische Verfahren vorgestellt und an praxisrelevanten Beispielen erläutert. Die Vorlesung lässt sich in zwei Teile gliedern: Die Funktionsweise der symmetrischen Kryptographie, einschließlich der Beschreibung historischer symmetrischer Chiffren (Caesar Chiffre, affine Chiffre, One Time Pad) und aktueller symmetrischer Verfahren (Advanced Encryption Standard, Data Encryption Standard, Stromchiffren), wird im ersten Teil behandelt. Der zweite Teil besteht aus einer Einführung in die asymmetrische Kryptographie und zwei ihrer wichtigsten Stellvertreter (RSA und der Diffie-Hellman-Schlüsselaustausch). Hierfür werden relevante Grundlagen der Zahlentheorie eingeführt, um ein grundlegendes Verständnis der Verfahren sicherzustellen (u. a. Ringe ganzer Zahlen, Gruppen, Körper, diskrete Logarithmen, euklidischer Algorithmus, eulersche Phi-Funktion).					

Nichtsdestotrotz liegt der Schwerpunkt auf der algorithmischen Einführung asymmetrischer Krypto-Verfahren.

Lehrformen

Vorlesung mit Übung, die Veranstaltung wird digital angeboten

Prüfungsformen

Klausurarbeit (120 Minuten)

Voraussetzungen für die Vergabe von Credits

Erfolgreiches Bestehen der Modulklausur.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/149: B.Sc. IT-Sicherheit [PO 20]

5/150: B.Sc. IT-Sicherheit [PO 22]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

5/96 : M.Sc. IT-Sicherheit/ Netze und Systeme [PO20]

5/99 : M.Sc. IT-Sicherheit/ Netze und Systeme [PO22]

Titel des Moduls: Einführung in die Kryptographie 2
Introduction to Cryptography 2

Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Einführung in die Kryptographie 2 (211009)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 300 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen Keine		

Modulbeauftragte/r und hauptamtlich Lehrende
 Modulbeauftragte/r: Prof. Dr.-Ing. Christof Paar
 Lehrende: Prof. Dr.-Ing. Christof Paar

Verwendung des Moduls

B.Sc. IT-Sicherheit
 B.Sc. Informatik
 B.Sc. Angewandte Informatik
 M.Sc. IT-Sicherheit/ Netze und Systeme

Vorkenntnisse

Inhalte der Vorlesung "Einführung in die Kryptographie 1"

Lernziele (learning outcomes)

Nach erfolgreichem Abschluss des Moduls verfügen die Studierenden über Kenntnisse der grundlegenden Anwendungen asymmetrischer und hybrider Verfahren. Sie können entscheiden, wann welche kryptographischen Lösungen eingesetzt werden können, um gewisse Sicherheitsziele zu erreichen. Die Studierenden erreichen durch Beschreibungen ausgewählter praxisrelevanter Algorithmen, z. B. basierend auf elliptischen Kurven, Hash-Funktionen, digitalen Signaturverfahren und Post-Quantum-Kryptographie, ein algorithmisches und technisches Verständnis für den Einsatz in der Praxis. Sie wissen, wie Sicherheitsparameter für kryptographische Lösungen zu wählen sind und sind in der Lage argumentativ eine bestimmte Lösung zu verteidigen. Die Vorlesung bereitet zudem auf Veranstaltungen vor, in denen Kryptographie eingesetzt wird, beispielsweise in der Netz- und Software-Sicherheit, sowie auf Vorlesungen, in denen Kryptographie mit formalen Methoden behandelt wird. Die Vorlesungen werden zusätzlich auch als Videos in Deutsch und Englisch angeboten. Durch das zweisprachige E-Learning-Angebot können die Studierenden Sprachkompetenzen in der Wissenschaftssprache Englisch erwerben.

Inhalt

Das Modul bietet eine vertiefende Einführung in die moderne angewandte Kryptographie und Grundlagen der IT-Sicherheit. Der Fokus liegt auf asymmetrischen kryptographischen Verfahren, Hashfunktionen und der Konstruktion von Schemata wie digitalen Signaturen und Protokollen zur Schlüsselvereinbarung.

Die Vorlesung lässt sich in drei Teile gliedern: Im ersten Teil werden Verfahren basierend auf elliptischen Kurven, ein für die Praxis sehr wichtiger Vertreter der asymmetrischen Kryptographie, vorgestellt. Ebenso werden die Grundlagen der sogenannten Post-Quanten-Kryptographie erläutert und Hash-basierte Verfahren eingeführt.

Im zweiten Teil der Vorlesung werden Hash-Funktionen, die streng genommen symmetrische Verfahren sind, behandelt. Als wichtiger Praxisvertreter wird der Algorithmus SHA-3 im Detail besprochen.

Im letzten Teil werden Sicherheitsdienste (u. a. Authentisierung, Integrität und Nichtzurückweisbarkeit) und hierfür

notwendige Schemata behandelt, die auf symmetrischen und asymmetrischen Krypto-Verfahren beruhen. Konkret werden Schemata für digitale Signaturen, Message Authentication Codes (MACs) sowie die Grundlagen der Schlüsselvereinbarung, digitale Zertifikate und PKI eingeführt.

Lehrformen

Vorlesung mit Übungen, Die Veranstaltung wird online durchgeführt

Prüfungsformen

Klausurarbeit (120 Minuten)

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/150: B.Sc. IT-Sicherheit/Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/Informationstechnik [PO 20]

5/165: B.Sc. Informatik [PO 22]

5/158: B.Sc. Informatik [PO 20]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

5/96 : M.Sc. IT-Sicherheit / Netze und Systeme [PO20]

5/99 : M.Sc. IT-Sicherheit / Netze und Systeme [PO22]

Titel des Moduls: Functional Programming Functional Programming					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Functional Programming (211060)			Kontaktzeit 4 SWS (60 h)	Selbststudium 90 h	Gruppengröße 50 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen kein		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Dr. Clara Schneidewind Lehrende: Dr. Clara Schneidewind Dr. Roberto Blanco Dr. Catalin Hritcu					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik M.Sc. IT-Sicherheit / Informationstechnik					
Vorkenntnisse Es sind keine besonderen Vorkenntnisse erforderlich. Der Kurs ist für Studierende aller Niveaus zugänglich und dürfte sowohl für Studierende interessant sein, die bereits über Programmierkenntnisse verfügen und ein eleganteres Programmierparadigma erlernen möchten, als auch für Studierende ohne vorherige Programmiererfahrung.					
Lernziele (learning outcomes) Nach erfolgreichem Abschluss dieses Kurses können die Studierenden: <ul style="list-style-type: none"> • Programme in funktionalen Hochsprachen, insbesondere in OCaml, entwickeln • die Verwendung von Rekursion zur Definition von Datenstrukturen (Listen, Maps, Bäume, etc.) und rein funktionalen Algorithmen verstehen und anwenden • den Aufbau und die Vorteile von Typensystemen verstehen und sie zur Unterstützung des Programmierentwurfs und der Implementierung einzusetzen • fortgeschrittene Funktionen der funktionalen Programmierung analysieren, einschließlich Typ-Polymorphismus und Funktionen höherer Ordnung • informelle Überlegungen zur Korrektheit und Effizienz funktionaler Programme anstellen und formale Alternativen zur Argumentation kennen • Anwendung von Typabstraktion und Modularisierung zur Strukturierung von Programmen in Sammlungen von Bibliotheken und Verwendung dieser Bibliotheken, um darauf komplexere Programme aufzubauen • die grundlegenden Prinzipien des Entwurfs von Programmiersprachen verstehen, insbesondere in Bezug auf die funktionale Programmierung • einfache Programmiersprachen entwerfen und entwickeln, einschließlich ihrer formalen Definition und der anschließenden Implementierung als Interpreter 					
Inhalt Dieser Kurs bietet eine fundierte und dennoch praxisnahe Einführung in die Prinzipien und Praxis der Programmierung nach dem funktionalen Paradigma. Diese zunehmend populäre Disziplin basiert auf der Definition und Ausführung von Funktionen zur Durchführung von Berechnungen. In der reinen funktionalen Programmierung, die im Mittelpunkt des Kurses steht, stellen diese Funktionen in sich abgeschlossene Berechnungen dar, die andere Teile des Programms nicht beeinflussen. Infolgedessen sind funktionale Programme elegant, effizient, leicht zu verstehen und zu ändern und schließen viele häufige Fehlerquellen aus.					

die bei der imperativen Programmierung auftreten. Wir untersuchen die Prinzipien, die das Design funktionaler Sprachen leiten, wie sie funktionieren und warum. Im Mittelpunkt dieser Untersuchung steht der Begriff der starken Typisierung und der Entwurf von Typsystemen, die es uns ermöglichen, ausdrucksstarke und gut funktionierende Programme zu schreiben.

Während des gesamten Kurses werden die theoretischen Grundlagen der Programmiersprache OCaml sorgfältig von den ersten Prinzipien an erklärt und durch interaktive Übungen sofort in die Praxis umgesetzt. Diese decken alles ab, was die Studenten benötigen, um komplexe funktionale Programme zu entwickeln, einschließlich erweiterter Fallstudien wie eine kleine Programmiersprache, die auf OCaml selbst basiert. Die im Kurs erlernten Techniken ermöglichen es den Studierenden, sich zu geschickten Programmierern zu entwickeln und viele fortgeschrittene Funktionen auch in den gängigen Programmiersprachen zu nutzen, da diese zunehmend von den Fortschritten der funktionalen Programmierung beeinflusst werden. Darüber hinaus erwerben sie die notwendigen Grundlagen, um ihr Wissen über die Programmierung und deren Verbindung zur Informatik, Mathematik und Logik zu vertiefen. Da die funktionale Programmierung die primäre Alternative zur imperativen Programmierung darstellt und das direkteste Mittel zur Erstellung korrekter Programme ist, kann jeder, der ein besserer Programmierer werden möchte, von diesem Kurs profitieren.

Lehrformen

Vorlesung mit Übung

Prüfungsformen

Schriftliche Modulabschlussprüfung (120 Minuten)

Voraussetzungen für die Vergabe von Credits

Bestehen der Modulabschlussprüfung und erfolgreiche Teilnahme an den Übungen.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/170: B.Sc. Informatik [PO 20]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

5/91: M.Sc. IT-Sicherheit / Informationstechnik [PO 22]

5/84: M.Sc. IT-Sicherheit / Informationstechnik [PO 20]

Titel des Moduls: Game Development Game Development					
Modul-Nr./Code	Credits 6 CP	Workload 180 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Game Development (211001)			Kontaktzeit 30 h	Selbststudium 150 h	Gruppengröße Studierende
Unterrichtssprache Deutsch, Kursmaterial auf Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Tobias Glasmachers Lehrende: M. Sc. Daniel Vonk					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik					
Vorkenntnisse Grundlegende Programmierkenntnisse; Zugang zu einem Computer mit Internet, auf dem die Unity-Engine (https://unity3d.com/de/get-unity/download)					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • verstehen die Studierenden Grundlagen der objektorientierten Programmierung mit C# im Rahmen der Unity-Engine, • haben die Studierenden ein umfassendes Wissen über den Bereich der Spieleentwicklung erworben und kennen moderne Tools sowie aktuelle Methoden der 2D- und 3D-Entwicklung, • können die Studierenden praxisnahe Problemstellungen der Softwareentwicklung analysieren und eigenständig lösen, • können die Studierenden Projekte im Bereich der Spieleentwicklung definieren und fachgerecht umsetzen 					
Inhalt Die Veranstaltung bietet einen umfangreichen Einblick in viele Bereiche der Spieleentwicklung. Dazu gehören: <ul style="list-style-type: none"> • Grundlagenwissen (Spiele-Engines, moderne Softwaretools, Projektmanagement) • C#-Grundlagen (Syntax, Datentypen, Operatoren, Kontrollstrukturen) • Benutzerinteraktion (In-/Output mit Tastatur sowie Controller, User Interfaces) • Gameplay (Bewegen von Spielobjekten, Kamerasteuerung, Game Loop und Framerates) • Physik (Rigidbody, Collider, Trigger) • Assets (Import von Bildern, Audio und 3D-Modellen sowie Erstellung von Animationen) • Grafik (Texturen, Partikeleffekte, Beleuchtung, Post-Processing) • Leveldesign (Tilemaps, 3D-Umgebungen, Terrains) Studierende setzen das erlernte Wissen durch die Entwicklung einfacher Computerspiele in der Unity-Engine um. Die erworbenen Fähigkeiten lassen sich jedoch einfach auf andere Software-Frameworks übertragen. 					
Lehrformen Online-Videos und wöchentliche Hörsaalübungen					
Prüfungsformen Semesterbegleitende Projektarbeiten					

Voraussetzungen für die Vergabe von Credits

Bestandene Projektarbeiten; Studienleistung: Bearbeitung wöchentlicher Tests, die sich inhaltlich auf die einzelnen Kursabschnitte beziehen. Für die Zulassung zu den Projektarbeiten müssen die jeweiligen Tests erfolgreich absolviert worden sein.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

6/158: B.Sc. Informatik [PO 22]

6/165: B.Sc. Informatik [PO 20]

6/168: B.Sc. Angewandte Informatik [PO 22]

6/170: B.Sc. Angewandte Informatik [PO 20]

Titel des Moduls: Highlights of Theoretical Computer Science [B.Sc.]
Highlights of Theoretical Computer Science [B.Sc.]

Modul-Nr./Code	Credits 10 CP	Workload 300 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Highlights of Theoretical Computer Science (211057)			Kontaktzeit 90 h	Selbststudium 210	Gruppengröße 40 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Successful completion of an introductory course on theoretical computer science (covering formal languages, basics of complexity theory including NP-completeness and reductions, basics of computability theory). Interest and motivation to learn about theoretical concepts.		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Michael Walter Prof. Dr. Thomas Zeume Lehrende: Prof. Dr. Michael Walter Prof. Dr. Thomas Zeume Dr. Vladimir Lysikov					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik B.Sc. IT-Sicherheit / Informationstechnik					
Vorkenntnisse					
Lernziele (learning outcomes) You will know some of the most important results and insights of modern theoretical computer science. You will learn approaches and techniques that go well beyond a first course. You will be able to recognize when these can be used and how to adapt them to new situations. You will be able to independently acquire new knowledge in this area.					
Inhalt The insights and techniques of modern theoretical computer science have been key for advances in all areas of computer science. In this course, we will discuss some highlights and the techniques that underpin them. Possible topics that we might cover: <ul style="list-style-type: none"> • Computational models (is there life beyond Turing machines?) • Kolmogorov complexity (what is the shortest program that produces some output?) • Communication complexity (how many bits must Alice and Bob exchange to jointly solve a problem?) • Fine-grained complexity (are some easy problems easier than others? and why?) • Fast multiplication of numbers and matrices (can you beat the high-school method?) • Randomness (does it really help to compute faster?) • Circuit lower bounds (why is it so hard to prove that problems are hard?) • Convex optimization (how to maximize profit if all you can ask are yes/no questions) • Hardness of approximation (how easy is it to find near-optimal solutions?) • Cryptography and computation 					

If you enjoyed your first course in theoretical computer science in the Bachelor's and would like to deepen your knowledge by getting an overview of the fascinating theory of computing, then this course will be exactly right for you.

Lehrformen

Vorlesung mit Übung

Prüfungsformen

Final module examination. Format will depend on number of participants.

Voraussetzungen für die Vergabe von Credits

Bestandene Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

10/158: B.Sc. Informatik [PO 22]

10/170: B.Sc. Informatik [PO 20]

10/168: B.Sc. Angewandte Informatik [PO 22]

10/170: B.Sc. Angewandte Informatik [PO 20]

10/150: B.Sc. IT-Sicherheit / Informationstechnik [PO 22]

10/149: B.Sc. IT-Sicherheit / Informationstechnik [PO 20]

Titel des Moduls: Information Theory Information Theory					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Information Theory (211007)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 30 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Michael Walter Lehrende: Prof. Dr. Michael Walter					
Verwendung des Moduls B.Sc. Informatik (bis SS 23) B.Sc. IT-Sicherheit M.Sc. Informatik M.Sc. IT-Sicherheit/ Informationstechnik M.Sc. IT-Sicherheit/ Netze und Systeme (bis SS 23) M.Sc. Angewandte Informatik					
Vorkenntnisse Vertrautheit mit der diskreten Wahrscheinlichkeitsrechnung (wir werden Sie kurz an die wichtigsten Fakten erinnern). Einige Erfahrung mit präzisen mathematischen Aussagen und strengen Beweisen (da wir viele davon im Kurs sehen werden). Ein Teil der Hausaufgaben wird die Programmierung in Python erfordern.					
Lernziele (learning outcomes) Sie werden grundlegende Konzepte, Algorithmen und Ergebnisse der Informationstheorie kennenlernen. Nach erfolgreichem Abschluss dieses Kurses kennen Sie das mathematische Modell der Informationstheorie, wissen, wie man Algorithmen für eine Vielzahl von Informationsverarbeitungsaufgaben entwirft und analysiert, und wie man sie in Python implementiert. Sie haben sich selbstständig in ein Thema der Informationstheorie eingeleesen und dieses vor Ihren Kommilitonen präsentiert. Sie werden auf einen weiterführenden Kurs oder ein Forschungs- oder Abschlussprojekt in diesem Bereich vorbereitet. Eine genaue Auflistung der Lernziele finden Sie auf der Homepage des Kurses.					
Inhalt Dieser Kurs gibt eine Einführung in die Informationstheorie - die mathematische Theorie der Information. Seit ihren Anfängen hat die Informationstheorie einen tiefgreifenden Einfluss auf die Gesellschaft gehabt. Sie bildet die Grundlage für wichtige technologische Entwicklungen, von zuverlässigen Speichern bis hin zu Mobilfunkstandards, und ihr vielseitiges mathematisches Instrumentarium findet Anwendung in der Informatik, dem maschinellen Lernen, der Physik, der Elektrotechnik, der Mathematik und vielen anderen Disziplinen. Ausgehend von der Wahrscheinlichkeitstheorie werden wir erörtern, wie man Informationsquellen und Kommunikationskanäle mathematisch modelliert, wie man Informationen optimal komprimiert und wie man fehlerkorrigierende Codes entwirft, die uns eine zuverlässige Kommunikation über verrauschte Kommunikationskanäle ermöglichen. Wir werden auch sehen, wie die in der Informationstheorie verwendeten					

Techniken allgemeiner angewendet werden können, um Vorhersagen aus verrauschten Daten zu treffen.

Vorläufiger Lehrplan:

- Begrüßung, Einführung in die Informationstheorie
- Auffrischung der Wahrscheinlichkeitstheorie
- Numerische Zufallsvariablen, Konvexität und Konkavität, Entropie
- Symbol-Codes: Verlustfreie Komprimierung, Huffman-Algorithmus
- Block-Codes: Shannons Quellencodierungstheorem, sein Beweis und Variationen
- Strom-Codes: Lempel-Ziv-Algorithmus
- Strom-Codes: Arithmetische Kodierung
- Gemeinsame Entropien & Kommunikation über verrauschte Kanäle
- Shannons Theorem der verrauschten Kodierung
- Beweis des Theorems der verrauschten Kodierung (Noisy Coding Theorem)
- Beweis der Umkehrung, Shannons Theorie und Praxis
- Reed-Solomon-Codes
- Nachrichtenübermittlung für Dekodierung und Inferenz, Ausblick
- Studentische Präsentationen

Weitere Informationen finden Sie auf der Kurs-Homepage https://qi.rub.de/it_ss23.

Lehrformen

Vorlesung mit Übung

Prüfungsformen

Schriftliche (180 Minuten) oder mündliche (30 Minuten) Modulabschlussprüfung, abhängig von der Teilnehmerzahl. Wird zum Kursbeginn bekanntgegeben.

Voraussetzungen für die Vergabe von Credits

Passed Exam

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/170: B.Sc. Informatik [PO 20]

5/150: B.Sc. IT-Sicherheit [PO 22]

5/149: B.Sc. IT-Sicherheit [PO 20]

5/97: M.Sc. Informatik

5/91: M.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

5/84: M.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

5/99: M.Sc. IT-Sicherheit/ Netze und Systeme [PO 22]

5/105: M.Sc. Angewandte Informatik

Titel des Moduls: Künstliche Neuronale Netze (kein Angebot im WS 24/25)
Artificial Neural Networks (no offer in WS 24/25)

Modul-Nr./Code	Credits 6 CP	Workload 180 h	Semester siehe Prüfungsordnung	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Artificial Neural Networks (212006)			Kontaktzeit 60 h	Selbststudium 120 h	Gruppengröße 150 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Sen Cheng Lehrende: Prof. Dr. Sen Cheng					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik M.Sc. IT-Sicherheit/ Informationstechnik					
Vorkenntnisse Grundkenntnisse in der Infinitesimalrechnung, linearen Algebra, Statistik und Informatik. Erfahrung mit einer höheren Programmiersprache.					
Lernziele (learning outcomes) Die mathematischen Grundlagen, Möglichkeiten und Beschränkungen überwachter Lernverfahren für Regression und Klassifikation mit künstlichen neuronalen Netzen (KNN), sowie für deren Anwendung erforderliche praktische Kenntnisse werden vermittelt. Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • verstehen Studierende die theoretisch-mathematischen Grundlagen von KNN im Kontext des überwachten Lernens. • können Studierende selbstständig zwischen verschiedenen KNN unterscheiden und in einer Anwendungssituation das geeignete Verfahren auswählen. • können Studierende grundlegende Verfahren selbstständig in einer höheren Programmiersprache implementieren, sowie ihre eigene Implementierung und Standard-Implementierung anderer auf Daten anwenden. • können Studierende Ergebnis der KNN selbstständig interpretieren, insbesondere erkennen, wann sie unrealistisch sind. 					
Inhalt Verfahren: Struktur von Optimierungsproblemen, Regression, logistische Regression, biologische neuronale Netze, Modellselektion, universelle Approximationstheorem, Perzeptron, mehr-schichtiges Perzeptron, Backpropagation, tiefe neuronale Netze, rekurrente neuronale Netze, Long-Short Term Memory, Hopfield Netze, Boltzmann-Machine Software: python, numpy, matplotlib, scikit-learn, tensorflow					
Lehrformen Vorlesung, Hausaufgaben, angeleitete Übungen am Computer					
Prüfungsformen Schriftliche Modulabschlussprüfung (120 Minuten)					

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

6/168: B.Sc. Angewandte Informatik [PO 22]

6/170: B.Sc. Angewandte Informatik [PO 20]

6/158: B.Sc. Informatik [PO 22]

6/170: B.Sc. Informatik [PO 20]

6/91: M.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

6/84: M.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

Titel des Moduls: Model Checking (kein Angebot im SS 24)

Model Checking (no offer in SS 24)

Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Model Checking (211000)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Thomas Zeume Lehrende: Dr. Nils Vortmeier Marko Schmellenkamp					
Verwendung des Moduls B.Sc. Informatik M.Sc. Informatik M.Sc. Angewandte Informatik B.Sc. IT-Sicherheit/ Informationstechnik M.Sc. Mathematik					
Vorkenntnisse Grundlagenvorlesungen Mathematik, Einführung in die Theoretische Informatik (ggf. kann das nötige Wissen auch nachgeholt werden), Hilfreich: Logik in der Informatik, Datenstrukturen und elementare Programmierkenntnisse					
Lernziele (learning outcomes) Die Studierenden lernen wie sich verteilte Systeme durch Transitionssysteme modellieren und Eigenschaften in logischen Spezifikationssprachen wie LTL und CTL spezifizieren lassen. Sie sollen elementare Algorithmen zur Überprüfung von Eigenschaften in Transitionssystemen kennenlernen. Sie sollen ein Verständnis für die Möglichkeiten und Grenzen des Model Checking entwickeln, und in die Lage versetzt werden, sich eigenständig mit fortgeschrittenen Methoden des Model Checkings auseinanderzusetzen.					
Inhalt Wie kann die Korrektheit von Software und Hardware formal überprüft werden? Im Model Checking werden Software- und Hardware-Module durch Transitionssysteme formalisiert; gewünschte Eigenschaften mit Hilfe logischer Formalismen formal beschrieben; und mit Hilfe von Algorithmen automatisiert überprüft, ob ein Transitionssystem eine formal spezifizierte Eigenschaft besitzt. In dieser Veranstaltung werden die theoretischen Grundlagen des Model Checkings vermittelt, mit einem Fokus auf logik-basierten Spezifikationssprachen. Die Spezifikationssprachen LTL und CTL werden eingeführt, ihre Ausdrucksstärke untersucht, und die wichtigsten algorithmischen Ansätze für das Model Checking vorgestellt.					
Lehrformen Vorlesung mit Übungen					
Prüfungsformen Abschlussprüfung; mündliche Prüfung (20-30min) oder schriftliche Klausur (120min) in Abhängigkeit der Teilnehmerzahl					
Voraussetzungen für die Vergabe von Credits Erfolgreiches Bestehen der Modulprüfung.					

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/170: B.Sc. Informatik [PO 22]

5/158: B.Sc. Informatik [PO 20]

5/97: M.Sc. Informatik

5/105: M.Sc. Angewandte Informatik

5/150: B.Sc. IT-Sicherheit /Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

Titel des Moduls: Natural language processing with deep learning [B.Sc]

Natural language processing with deep learning [B.Sc]

Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen			Kontaktzeit 4 SWS (60 h)	Selbststudium 90 h	Gruppengröße 100 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Eigenes Laptop mit Python und einer geeigneten IDE für die Übungen mitbringen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Ivan Habernal Lehrende: Prof. Dr. Ivan Habernal					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik					
Vorkenntnisse Praktisch: Kenntnisse in Python (Projekt-Strukturen, OOP, Unit-Tests), Kenntnisse in Numpy sind von Vorteil. Theoretisch: Grundlagen der Analysis (Ableitungen) und der linearen Algebra (Vektoren), obwohl wir das wiederholen werden. Einige Grundkenntnisse der Linguistik (z. B. Syntax) auf Oberschulniveau sind ebenfalls von Vorteil.					
Lernziele (learning outcomes) Die Studierenden <ul style="list-style-type: none">• verstehen das Rückgrat des modernen NLP, wie z.B. Embeddings und Transformer-basierte Modelle• können die Fähigkeiten und Schwächen verschiedener Deep-Learning-Modelle im NLP kritisch beurteilen• verstehen verschiedene Tasks im NLP, deren Bewertung und Modellierungsannahmen• können verschiedene Modelle und Ansätze in Python implementieren und sammeln praktische Erfahrungen					
Inhalt Typische Aufgaben und Datensätze der natürlichen Sprachverarbeitung (NLP) und ihre Bewertung. Warum ist NLP schwierig? Auffrischung der mathematischen Grundlagen, Calculus, gradientenbasierte Optimierung, Backpropagation für beliebige Funktionen. Log-lineare Modelle und Textklassifikation. Tiefe neuronale Netze. Sprachmodelle und Word embeddings. Lernen statischer Word embeddings. Rekurrente neuronale Netze. Encoder-Decoder, Text Generation, Attention und autoregressive Modelle. Transformers. Self-Attention und BERT. Reine Decoder-Modelle und GPT. LLMs: Prompting und In-context Lernen					
Lehrformen Wir werden Vorlesungen und Übungen haben. Die Vorlesungen werden ziemlich interaktiv sein, da versucht wird, die Studierenden in Fragen und kritisches Denken einzubeziehen. Die Folien für jede Vorlesung werden im Voraus hochgeladen, damit die Studierenden sie für ihre Notizen verwenden können. Jede Vorlesung enthält Links zu relevanten Forschungsarbeiten für diejenigen, die sich eingehender mit dem Thema befassen möchten. Die Vorlesungen sind überwiegend theoretisch, d. h. wir behandeln die wichtigsten Konzepte, Ideen und mathematischen Beschreibungen, aber nicht, wie man es in einem bestimmten Framework programmiert. In den praktischen Kursen werden wir genau das Gegenteil tun. Die Studierenden werden mit aktuellen Mainstream-Frameworks für Deep Learning in NLP, wie Pytorch oder Huggingface, experimentieren. Wir werden ein breites Spektrum an Komplexitäten abdecken, von der Programmierung eines einfachen neuronalen Netzes von Grund auf bis hin zur Verwendung eines vortrainierten Sprachmodells.					
Prüfungsformen Schriftliche Modulabschlussprüfung über 90 Minuten					

Voraussetzungen für die Vergabe von Credits

Bestandene Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/168: B.Sc. Angewandte Informatik

5/158: B.Sc. Informatik

Titel des Moduls: Nebenläufige Programmierung (letztmalig SS 23)

Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Nebenläufige Programmierung (211012)			Kontaktzeit 45 h	Selbststudium 105 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Dr.-Ing. Doga Arinir (Lehrauftrag) Lehrende: Dr.-Ing. Doga Arinir					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik Master IT-Sicherheit/ Informationstechnik					
Vorkenntnisse Beherrschung einer Objektorientierten Programmiersprache (idealerweise Java)					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none">• haben die Studierenden grundlegende Fähigkeiten und Techniken erworben, um nebenläufige Programme sicher entwickeln zu können• kennen die Studierenden softwaretechnische Entwurfsmuster, welche bekannte Probleme bei nebenläufigen Programmen, wie zum Beispiel die Verklemmung, vermeiden lassen• können die Studierenden die Performanz von Programmen durch den Einsatz der nebenläufigen Programmierung verbessern• sind die Studierenden in der Lage, bestehende Programme zu analysieren und mögliche Fehler zu erkennen• können die Studierenden die Sprachmerkmale und Schnittstellen von JAVA für die nebenläufige Programmierung sicher anwenden					
Inhalt Moderne Hardware-Architekturen lassen sich nur durch den Einsatz nebenläufiger Programme richtig ausnutzen. Die nebenläufige Programmierung garantiert bei richtiger Anwendung eine optimale Auslastung der Hardware. Jedoch sind mit einem sorglosen Einsatz dieser Technik auch viele Risiken verbunden. Die Veranstaltung stellt Vorteile und auch Probleme nebenläufiger Programme dar und zeigt, wie sich die Performanz von Programmen verbessern lässt. 1. Nebenläufigkeit: Schnelleinstieg <ul style="list-style-type: none">• Anwendungen vs. Prozesse• Programme und ihre Ausführung• Vorteile und Probleme von nebenläufigen Programmen (Verbesserung der Performanz, Synchronisation, Realisierung kritischer Abschnitte, Monitore, Lebendigkeit, Verklemmungen) 2. Threads in Java					

3. UML-Modellierung von Nebenläufigkeit
4. Neues zur Nebenläufigkeit in Java 5 und Java 6
5. Realisierung von Nebenläufigkeit
6. Fortschritte Java-Konzepte für Nebenläufigkeit
6. Fortschritte Java-Konzepte für Nebenläufigkeit

Lehrformen

Online Vorlesung mit begleitendem eLearning Kurs

Prüfungsformen

Schriftliche Modulabschlussprüfung über 90 Minuten

Voraussetzungen für die Vergabe von Credits

Bestandene Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

5/91 Master IT-Sicherheit/ Informationstechnik

Titel des Moduls: Network Planing (kein Angebot im WS 24/25)					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Network Planing			Kontaktzeit 45 h	Selbststudium 105	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Aydin Sezgin Lehrende: Prof. Aydin Sezgin					
Verwendung des Moduls B.Sc. Informatik					
Vorkenntnisse					
Lernziele (learning outcomes) Die Studierenden beherrschen die Behandlung zentraler Aspekte der Linearen Optimierung. Dies sind: <ul style="list-style-type: none"> • die Modellierung von Problemen im Bereich der Informationstechnik (z.B. Leistungsallokation) sowie im Alltag (z.B. Rucksackproblem, Sudoku, Ernährung) als lineare Optimierungsprobleme • die Dualität sowie notwendige und hinreichende Bedingungen • Verfahren, die zur effizienten Bestimmung von Lösungen führen 					
Inhalt In vielen technischen (aber auch nichttechnischen) Bereichen werden Lösungen für Probleme gesucht, bei denen auch immer gewisse Vorgaben oder Nebenbedingungen erfüllt werden müssen. Die Optimierung dient hierbei als systematisches Werkzeug zur effizienten Lösungsbestimmung. Der Anwendungsfokus der Vorlesung ist in der Netzwerk-Planung wie Interferenz-Management, Frequenz- und Nutzerzuweisungen, Positionierung von Basisstationen sowie Routing. <ol style="list-style-type: none"> 1. Einleitung und Überblick <ul style="list-style-type: none"> ○ Motivation, Formulierung von linearen Problemen, Varianten, Beispiele, stückweise lineare Zielfunktionen ○ Graphische Darstellung und Lösung ○ Lineare Algebra: Überblick und Notation 2. Geometrie der linearen Optimierung <ul style="list-style-type: none"> ○ Konvexe Mengen, Polyhedra, Extrempunkte 3. Die Simplex-Methode <ul style="list-style-type: none"> ○ Optimalitätsbedingungen, Entwicklung, Implementierung 4. Dualitätstheorie <ul style="list-style-type: none"> ○ Motivation, Duales Problem, Dualitätstheorem 5. Spieltheorie 6. Sensitivitätsanalyse (Lokale) 7. Netzwerk-Fluss-Probleme <ul style="list-style-type: none"> ○ Formulierung, Probleme: Kürzester Pfad/Maximaler Fluss, Netzwerk-Simplex Algorithmus 8. Innere-Punkt-Methoden <ul style="list-style-type: none"> ○ Affiner Skalierungsalgorithmus 9. Ganzzahlige Optimierung <ul style="list-style-type: none"> ○ Formulierung ○ Methoden: Branch and bound, cutting plane 10. Anwendungen 					

Lehrformen

Vorlesung mit Übung

Prüfungsformen

Es gibt jeweils 4 Übungsblätter mit theoretischen Teilaufgaben mit insgesamt 15 Punkten und zusätzlich 4 Programmieraufgaben mit je 10 Punkten. Die Prüfung ist bestanden, wenn 30 Punkte in den theoretischen Aufgaben und 20 Punkte in den Programmieraufgaben erreicht werden.

Voraussetzungen für die Vergabe von Credits

Erfolgreiches Bestehen der Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

Titel des Moduls: Private and Anonymous Communication Private and Anonymous Communication					
Modul-Nr./Code	Credits 5 CP	Workload 150h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Private and Anonymous Communication (212028)			Kontaktzeit 4 SWS (60h)	Selbststudium 90 h	Gruppengröße 40 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Katharina Kohls Lehrende: Prof. Dr. Katharina Kohls					
Verwendung des Moduls B.Sc. IT-Sicherheit/Informationstechnik B.Sc. Informatik					
Vorkenntnisse In der Vorlesung werden Themen behandelt, in denen es um die Privacy in online Kommunikation geht. Dabei ist ein grundlegendes Vorwissen im Bereich der Computernetze und deren Sicherheit hilfreich, um die Privacy Konzepte der Vorlesung besser zu verstehen.					
Lernziele (learning outcomes) <ul style="list-style-type: none"> • Verständnis von Privacy Konzepten • Fähigkeit, diese Konzepte auf Kommunikationsinfrastrukturen anzuwenden, zum Beispiel bei Anwendungsfällen im Internet • Überblick über den aktuellen Stand der Technik hinsichtlich existierender Angriffe und Gegenmaßnahmen 					
Inhalt Bei der Nutzung des Internets hinterlassen wir Spuren, die wichtige und teilweise sensitive Informationen enthalten. Beispielsweise wird für eine Standardverbindung im Internet die IP-Adresse des Nutzers zusammen mit der Adresse des Endpunkts verwendet. Diese Informationen sind sensitiv, weil sie Aufschluss darüber geben, wo der Nutzer sich befindet und welchen Interessen online nachgegangen wird. Ein Angreifer kann solche Informationen sammeln, um daraus beispielsweise Nutzerprofile zu generieren. Im Rahmen der Vorlesung analysieren wir auf welche Weise Nutzer solche Spuren hinterlassen. Wir betrachten dazu die Qualität dieser Informationen, wie ein Angreifer darauf Zugriff bekommt, und welche Konsequenzen die unterschiedlichen Angriffe haben. Dazu schauen wir uns Angriffe auf dem aktuellen Stand der Technik an und evaluieren deren Angreifermodelle und die Konsequenzen der Angriffe. Gleichermaßen betrachten wir individuelle Gegenmaßnahmen und Systeme, die zusätzlichen Schutz für Nutzer bieten.					
Lehrformen Der Kurs besteht aus Vorlesungen, die das theoretische Wissen vermitteln, und praktischen Übungen in denen das Gelernte angewendet wird.					
Prüfungsformen Schriftliche Modulabschlussprüfung, 120 Minuten					
Voraussetzungen für die Vergabe von Credits Bestandene schriftliche Modulabschlussprüfung					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 5/150: B.Sc. IT-Sicherheit/Informationstechnik					

Titel des Moduls: Programmanalyse [B.Sc]					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Programmanalyse (211015)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Kevin Borgolte Lehrende: Prof. Kevin Borgolte					
Verwendung des Moduls B.Sc. IT-Sicherheit/ Informationstechnik B.Sc. Informatik					
Vorkenntnisse keine					
Lernziele (learning outcomes) Die Studierenden kennen verschiedene Konzepte, Techniken und Tools aus dem Bereich der Programmanalyse. Dies beinhaltet den Überblick über verschiedene Konzepte aus dem Bereich Reverse Engineering sowie Binäranalyse. Die Studierenden haben grundlegendes Verständnis von sowohl statischen als auch dynamischen Methoden zur Analyse eines gegebenen Programms. Sie sind in der Lage, verschiedene Aspekte der Programmanalyse zu beschreiben und auf neue Problemstellungen anzuwenden.					
Inhalt In der Vorlesung werden unter anderem die folgenden Themen und Techniken aus dem Bereich der Programmanalyse behandelt: <ul style="list-style-type: none"> • Statische und dynamische Analyse von Programmen • Analyse von Kontroll- und Datenfluss • Symbolische Ausführung • Taint Tracking • Binary Instrumentation • Program Slicing • Überblick zu existierenden Analysetools Daneben wird im ersten Teil der Vorlesung eine Einführung in x86/x64 Assembler gegeben sowie die grundlegenden Techniken aus dem Themenbereich Reverse Engineering vorgestellt. Begleitet wird die Vorlesung von Übungen, in denen die vorgestellten Konzepte und Techniken praktisch eingeübt werden.					
Lehrformen Vorlesung mit Übung					
Prüfungsformen mündliche oder schriftliche Modulabschlussprüfung (wird zu Beginn des Semester bekanntgegeben), Anmeldung: FlexNow					
Voraussetzungen für die Vergabe von Credits Bestandene Modulabschlussprüfung					

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/150: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

5/158: B.Sc. Informatik [PO 22]

5/170: B.Sc. Informatik [PO 22]

Titel des Moduls: Programming for Modern Machine Learning Programming for Modern Machine Learning					
Modul-Nr./Code	Credits 6 CP	Workload 180 h	Semester see examination regulations	Turnus Winter Semester	Dauer 1 Semester
Lehrveranstaltungen Programming for Modern Machine Learning (212040)			Kontaktzeit 60h (4 SWS)	Selbststudium 120 h	Gruppengröße 50 Studierende
Unterrichtssprache English			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Nils Jansen Prof. Dr. Bilal Zafar Lehrende: Prof. Dr. Nils Jansen (http://nilsjansen.org) Prof. Dr. Bilal Zafar (https://informatik.rub.de/zafar)					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik					
Vorkenntnisse Basic knowledge of statistics and programming.					
Lernziele (learning outcomes) At the end of this course, you will be able to: <ol style="list-style-type: none"> 1. Map a real world problem on a AI/ML paradigm like classification and reinforcement learning. 2. Select the right model type for your task, e.g., XGBoost, MLP, CNN, Transformers. 3. Select the right frameworks (scikit-learn, PyTorch, HuggingFace Transformers)and APIs (OpenAI API) for your tasks. 4. Put together common components like data loaders and training loops to construct your projects. 5. Use best practices like testing and linting to build maintainable code 					
Inhalt Past few years have seen confluence of two related trends: 1/ A rapid adoption of Artificial Intelligence (AI) and Machine Learning (ML) in a wide range of real-world applications across a variety of domains, e.g., healthcare, engineering. 2/ Development of specialized tooling and design patterns for AI/ML workloads. The goal of this course is to introduce the students to various AI/ML prediction paradigms, popular frameworks and design patterns. Specifically, we will build code bases involving (shallow) classification / regression models, CNNs and Transformers using frameworks like scikit-learn, PyTorch and Transformers. We will learn about using data loaders to manage large scale dataset and using GPUs to speed up deep learning workloads. We will also learn about best practices like testing and reproducibility.					
Lehrformen The course will be split into four blocks, each corresponding to a distinct paradigm of machine learning (traditional supervised learning, computer vision, language modeling and reinforcement learning). Each block will have corresponding in-person lectures and mini-projects.					
Prüfungsformen 50% e-Klausur (120 Minuten) + 50% Projektarbeit					
Voraussetzungen für die Vergabe von Credits Passing grade in the practical project and the final exam.					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)					

6/168: B.Sc. Angewandte Informatik

6/158: B.Sc. Informatik

Titel des Moduls: Proofs are programs [B.Sc.] Proofs are programs [B.Sc.]					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Proofs are Programms (211003)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Dr. Catalin Hritcu Lehrende: Dr. Catalin Hritcu Dr. Clara Schneidewind					
Verwendung des Moduls B.Sc. Informatik					
Vorkenntnisse The lecture is intended for a broad range of students, from motivated BSc students to MSc and PhD students. No specific prior knowledge in logic, programming, functional programming, or programming languages is assumed, though a degree of mathematical maturity is helpful.					
Lernziele (learning outcomes) After successful completion of this course, students will be able to <ul style="list-style-type: none"> • develop purely functional programs using recursive functions on numbers, lists, maps, and various kinds of trees, including the abstract syntax trees of programs; • use functional programming concepts such as type polymorphism and higher-order functions, which are increasingly becoming mainstream; • formally state and prove theorems in the Coq proof assistant; • apply different proof techniques in Coq (e.g. equational reasoning, contradiction, case analysis, induction on natural numbers, lists, and trees, induction on rule derivations, proof automation); • define new inductive types and relations in Coq and prove statements about them; • comprehend how the syntax and semantics of simple imperative programs can be formally defined in Coq and how to prove theorems about such programs and languages; • understand how the absence of information leaks can be formalized as a security property called noninterference and enforced using secure-multi execution or simple type systems; • optional: write simple proof terms and understand the connection between constructive logics and typed functional programming that is at the heart of Coq, in which propositions are types and proofs are programs. 					
Inhalt Complex proofs on paper are difficult to construct, check, and maintain. This holds not only for interesting proofs in mathematics, but also for complex formal proofs about interesting programs. For this reason, machine-checked proofs created with the help of interactive tools called proof assistants are gaining increased traction in academia and industry. Proof assistants have been used to prove the correctness and security of realistic compilers, operating systems, cryptographic libraries, or smart contracts, and also to construct machine-checked proofs for challenging theorems in mathematics. This course introduces the Coq proof assistant [3] and explains how to use it to prove properties about functional programs and inductive relations, how to formally define a simple imperative programming language, and how to securely enforce information-flow control for functional and imperative programs. The Coq proof assistant enables us to program formal proofs interactively and it machine-checks the correctness of the proofs along the way. The design of the Coq proof assistant itself exploits a beautiful connection between programs in typed functional					

programming languages and proofs in constructive logics, which is known as the Curry-Howard Correspondence [4]. This deep connection between programs and proofs should make this course interesting to not only to computer scientists, but also to mathematicians and other scientists. The goal is to demystify proofs as just programs in an elegant programming language, for which the course provides a gentle introduction. The course also shows that proofs are not only a way to convince a human reader, but they can actually be fully formalized in a proof assistant like Coq and automatically checked by a computer.

This hands-on course is based on the Logical Foundations [1] and Security Foundations [2] online textbooks, which are themselves formalized and machine-checked in the Coq proof assistant. The many exercises in each book chapter are to be solved weekly mostly in Coq, from easy exercises allowing the students to practice concepts from the lecture, building incrementally to slightly more interesting programs and proofs and also to various optional challenges. Finally, this course serves as the base for a more advanced course on “Foundations of Programming Languages, Verification, and Security”.

Lehrformen

This course consists of lectures and weekly exercises, in which the students will solve problems using the Coq proof assistant for which they can get help from a tutor.

Prüfungsformen

Written final exam (120 minutes).

Voraussetzungen für die Vergabe von Credits

Passing the final written exam.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik

Titel des Moduls: Public Key Kryptanalyse 1 [B.Sc] (nicht im SoSe 25) Public Key Cryptanalysis 1 [B.Sc]					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer Semester
Lehrveranstaltungen Public Key Kryptanalyse 1 (211055)			Kontaktzeit 45 h	Selbststudium 105 h	Gruppengröße Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Alex May Lehrende: Prof. Alex May					
Verwendung des Moduls					
Vorkenntnisse Vorausgesetzt werden elementare Kenntnisse der Lineare Algebra (Mathematik 1 für Informatiker) und ein Interesse an algorithmischen Techniken und Kryptographie, in Theorie und Praxis (umgesetzt mit Hilfe des Computeralgebra-Systems Sage).					
Lernziele (learning outcomes) Die Studierenden sollen breite Kenntnisse zu algorithmischen Techniken der asymmetrischen Kryptanalyse, insbesondere für codierungsbasierte Kryptographie, erlangen. Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen die Studierenden grundlegende Schlüsselfindungs-Algorithmen wie Brute-Force und Meet-in-the-Middle und können diese auf neue kryptographische Systeme anwenden, • beherrschen sie die Grundlagen linearer Codes und ihrer Dualcodes, insbesondere als kryptographische Anwendung das McEliece-Kryptosystem, • kennen Studierende Time-Memory Techniken wie Pollard Rho und Parallel Collision Search, und können sie auf neue Probleme anwenden, • haben Studierende einen Überblick über alle aktuellen Dekodieralgorithmen im Bereich des Information Set Decoding, die für die Sicherheits-Evaluierung moderner kodierungsbasierter Kryptosysteme relevant sind, • sind Studierende in der Lage, Techniken der Kryptanalyse mit Hilfe der Computer-Algebra Sage zu implementieren. 					
Inhalt Kryptanalyse dient dazu, kryptographische Systeme derart zu instantiiieren, dass sie einerseits ein vordefiniertes Sicherheitsniveau bieten, andererseits aber möglichst performant sind. Die Kryptanalyse bietet dazu einen ganzen Werkzeugkoffer an algorithmischen Techniken, um die Evaluation neuer kryptographischer Systeme zu realisieren. Dies beinhaltet sowohl klassische Algorithmen als auch Algorithmen für Quantenrechner, damit die verwendete Kryptographie selbst in einer Ära von Quantenrechnern sicher bleiben.					
Lehrformen Die Vorlesung wird als seminaristischer Unterricht abgehalten, die praktischen Übungen am Rechner mit der Computer-Algebra Sage werden zudem weitere Lehrformen wie Gruppen- und Projektarbeit beinhalten.					
Prüfungsformen Schriftliche Modulabschlussprüfung über 120 Minuten					
Voraussetzungen für die Vergabe von Credits					

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/149 B.Sc. IT-Sicherheit [PO20]

5/150 B.Sc. IT-Sicherheit [PO22]

Titel des Moduls: Quantum Information and Computation [B.Sc.] Quantum Information and Computation [B.Sc.]					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Quantum Information and Computation (212011)			Kontaktzeit 4 SWS (60 h)	Selbststudium 90 h	Gruppengröße 40 Studierende
Unterrichtssprache Deutsch oder Englisch (depends on audience)			Teilnahmevoraussetzungen Keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Michael Walter Lehrende: Prof. Dr. Michael Walter					
Verwendung des Moduls B.Sc. Informatik B.Sc. IT-Sicherheit/ Informationstechnik					
Vorkenntnisse Familiarity with linear algebra (in finite dimensions) and probability (with finitely many outcomes) at the level of a first Bachelors course; we will briefly remind you of the more difficult bits in class. In addition, some mathematical maturity, since we will discuss precise mathematical statements and rigorous proofs. No background in physics is required.					
Lernziele (learning outcomes) You will learn fundamental concepts, algorithms, and results in quantum information and computation. After successful completion of this course, you will know the theoretical model of quantum information and computation, how to generalize computer science concepts to the quantum setting, how to design and analyze quantum algorithms and protocols for a variety of computational problems, and how to prove complexity theoretic lower bounds. You will be prepared for an advanced course or a research or thesis project in this area.					
Inhalt This course will give an introduction to quantum information and quantum computation from the perspective of theoretical computer science. Topics to be covered will likely include: <ul style="list-style-type: none"> • Fundamentals of quantum computing: quantum bits, states and operations • The power of quantum entanglement: nonlocal games • Entanglement as a resource: superdense coding and teleportation • Quantum circuit model of computation • Quantum computing with oracles: Deutsch-Jozsa, Bernstein-Vazirani, Simon • Quantum Fourier transform and phase estimation • Shor's factoring algorithm • Grover's search algorithm and beyond: how to solve SAT on a quantum computer? • From no cloning to quantum money: a peek at quantum cryptography <p>The course should be of interest to students of computer science, mathematics, physics, and related disciplines. Students interested in a BSc or MSc project in quantum information, computing, cryptography, etc. are particularly encouraged to participate.</p>					
Lehrformen Lecture with Exercise					

Prüfungsformen

Final written module exam (180 minutes)

Voraussetzungen für die Vergabe von Credits

Passed written exam

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/150: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

Titel des Moduls: Requirements Engineering Requirements Engineering					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Requirements Engineering (212001)			Kontaktzeit	Selbststudium 90 h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr.- Alena Naiakshina Lehrende: Prof. Dr.- Alena Naiakshina					
Verwendung des Moduls B.Sc. Informatik					
Vorkenntnisse Grundkenntnisse der Informatik.					
Lernziele (learning outcomes) Die Veranstaltung vermittelt essentielles Wissen für die systematische Ermittlung, Dokumentation und Verwaltung von Anforderungen in Softwareprojekten. Die Studierenden erlangen ein fundiertes Verständnis von der Bedeutung präziser Anforderungen und deren Auswirkungen auf den gesamten Entwicklungsprozess. Die Vorlesung zielt darauf ab, die Studierenden in die Lage zu versetzen, effiziente Requirements Engineering-Praktiken in ihren zukünftigen Projekten anzuwenden und somit den Erfolg von Softwareentwicklungsprojekten zu fördern.					
Inhalt Effektives Requirements Engineering bildet das fundamentale Rückgrat erfolgreicher Softwareentwicklungsprojekte. Ein solides Verständnis dieses Fachgebiets ist entscheidend, um präzise Anforderungen zu ermitteln und zu verwalten, die den Bedürfnissen und Erwartungen der Stakeholder gerecht werden. Fehlende oder unklare Anforderungen können zu kostspieligen Verzögerungen, unzufriedenen Kunden und potenziell fehlerhafter Software führen. All dies unterstreicht die Bedeutung des Requirements Engineering in der Softwareentwicklung und die Rolle präziser Anforderungen für den Projekterfolg. Studierende lernen verschiedene Anforderungsermittlungsmethoden wie Interviews, Workshops und Umfragen kennen. Zudem werden bewährte Praktiken zur strukturierten Dokumentation und effektiven Verwaltung von Anforderungen vermittelt. Die Validierung von Anforderungen, um ihre Vollständigkeit, Konsistenz und Realisierbarkeit sicherzustellen, ist ein weiterer Schwerpunkt. Darüber hinaus wird die Anforderungsverfolgung und Werkzeuge zur Verwaltung von Änderungen präsentiert. Abschließend wird der Einfluss des Requirements Engineering auf Projektmanagement und Stakeholderkommunikation beleuchtet.					
Lehrformen Vorlesung mit Medienunterstützung und Übungen mit Gruppenarbeit					
Prüfungsformen Schriftliche Modulabschlussprüfung über 90 Minuten.					
Voraussetzungen für die Vergabe von Credits Bestandene schriftliche Modulabschlussprüfung					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 5/170: B.Sc. Informatik [PO 20]					

Titel des Moduls: Software Security 1 [B.Sc.] Software Security 1 [B.Sc.]					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen Software Security 1 (212026)			Kontaktzeit 4 SWS (60 h)	Selbststudium 90 h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Kevin Borgolte Lehrende: Prof. Kevin Borgolte					
Verwendung des Moduls B.Sc. IT-Sicherheit/ Informationstechnik B.Sc. Informatik					
Vorkenntnisse Prior knowledge about programming in Python, C, and assembler is recommended. The following courses (or equivalent) are required: System Security (211011) Operating Systems (211005)					
Lernziele (learning outcomes) At the end of this course, students will be able to: <ul style="list-style-type: none"> • understand user-space software vulnerability types and protection mechanisms • understand how to write code to reduce the risk of vulnerabilities and apply defensive programming techniques • identify new software vulnerabilities and evaluate their impact • demonstrate the existence of a vulnerability, for example, by developing proof of concept exploits 					
Inhalt The course covers the area of introductory software security, vulnerability discovery, and vulnerability verification, focusing on: <ul style="list-style-type: none"> • Assembly and Disassembly, Shellcode • Binary Reverse Engineering and Debugging • Memory and Type Safety/Errors • Stack-based Buffer Overflows • Heap Attacks • Information Leakage • Format String Vulnerabilities • Code Re-use Attacks • Types and Type Safety • Race Conditions 					
Lehrformen Vorlesung mit Übung					
Prüfungsformen Practical exam.					

Voraussetzungen für die Vergabe von Credits

Passed practical exam. The exam will take place on two days of 6 hours each, both dates are necessary to pass.

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/150: B.Sc. IT-Sicherheit/ Informationstechnik

5/158: B.Sc. Informatik

Titel des Moduls: Statistisches Lernen und Data Mining Statistical learning and data mining					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Statistisches Lernen und Data Mining (150331)			Kontaktzeit 60 h	Selbststudium 90	Gruppengröße 20 Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Johannes Lederer Lehrende: Prof. Dr. Johannes Lederer					
Verwendung des Moduls B.Sc. Informatik M.Sc. Computer Science					
Vorkenntnisse					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • kennen die Studierenden Standardmethoden der Datenanalyse • verstehen die Studierenden, wann welche Methoden passend sind • sind die Studierenden in der Lage, die Methoden anzuwenden • können die Studierenden die Ergebnisse interpretieren 					
Inhalt In dieser Lehrveranstaltung werden die grundlegenden Methoden der Datenanalyse eingeführt. Dabei werden verschiedene Datentypen berücksichtigt, insbesondere Regressionsdaten und Klassifikationsdaten. Immer werden auch die zu Grunde liegenden statistischen Modelle besprochen. Ebenfalls werden mögliche Anwendungen sowohl im Unterricht als auch in Computer-Übungen vorgestellt. Ziel ist es, den gesamten Verlauf einfacher Datenanalysen zu vermitteln: Datenaufbereitung, statistische Modellbildung, Auswahl einer Methode, Implementierung der Methode, Visualisierung der Resultate und Interpretation.					
Lehrformen Hörsaalvorlesung mit Medienunterstützung, insbesondere Datenanalysen mit dem Computer, Tutorien als seminaristischer Unterricht, zusätzlich Selbststudium mit ergänzend bereitgestellten Materialien und Aufgaben					
Prüfungsformen Schriftliche Modulabschlussprüfung (90 Minuten)					
Voraussetzungen für die Vergabe von Credits Bestandene Modulabschlussprüfung					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 5/158: B.Sc. Informatik [PO 22] 5/165: B.Sc. Informatik [PO 20] 5/97: M.Sc. Computer Science					

Titel des Moduls: System Performance Evaluation (keine Angebot im WS 24/25)
System Performance Evaluation (no offer in WS 24/25)

Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung / see examination regulations	Turnus Wintersemester	Dauer 1 Semester
Lehrveranstaltungen System Performance Evaluation (212033)			Kontaktzeit 45 h	Selbststudium 105 h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr.-Ing. Steffen Bondorf Lehrende: Prof. Dr.-Ing. Steffen Bondorf					
Verwendung des Moduls B.Sc. Informatik B.Sc. Angewandte Informatik					
Vorkenntnisse Empfohlen: Inhalte der Module Technische Informatik 1 und Computernetze					
Lernziele (learning outcomes) Die Studierenden sollen eine breite Kenntnis über folgende Themenfelder erlangen: <ul style="list-style-type: none"> • Systemmodellierung und -leistungsanalyse • Entwurf von Experimenten • Messungen • Simulation 					
Inhalt Die Veranstaltung „System Performance Evaluation“ vermittelt grundlegende Techniken für Modellierung von Computersystemen und der Quantifizierung ihrer Leistung. Im Vordergrund stehen der Entwurf von Experimenten, Simulation und Warteschlangen-theorie.					
Lehrformen Die Vorlesung wird als seminaristischer Unterricht abgehalten, die praktischen Übungen am Rechner können weitere Lehrformen wie Gruppen- und Projektarbeit beinhalten.					
Prüfungsformen 50% semesterbegleitendes Projekt + 50% Abschlussprüfung (geplant mündlich)					
Voraussetzungen für die Vergabe von Credits Bestandenes Projekt und bestandene Modulabschlussprüfung					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 5/158: B.Sc. Informatik [PO 22] 5/170: B.Sc. Informatik [PO 20] 5/168: B.Sc. Angewandte Informatik [PO 22]					

Titel des Moduls: Systemsicherheit System Security					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Systemsicherheit (211011)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen keine		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr. Ghassan Karame Lehrende: Prof. Dr. Ghassan Karame					
Verwendung des Moduls B.Sc. IT-Sicherheit/ Informationstechnik B.Sc. Informatik M.Sc. Angewandte Informatik M.Sc. IT-Sicherheit/ Netze und Systeme					
Vorkenntnisse Background in Cryptographic primitives (encryption methods, signatures, MACs, hash functions), principles of communication networks, is recommended.					
Lernziele (learning outcomes) At the end of this course, students will be able to <ul style="list-style-type: none"> • classify and describe vulnerabilities and protection mechanisms of popular systems and protocols, and • analyze / reason about basic protection mechanisms for modern OSs, software, and hardware systems. Students will also develop the ability to reason about the security of a given protocol and independently develop appropriate security defenses and security models. 					
Inhalt While clearly beneficial, the large-scale deployment of online services has resulted in the increase of security threats against existing services. As the size of the global network grows, the incentives of attackers to abuse the operation of online applications also increase and their advantage in mounting successful attacks becomes considerable. These cyber-attacks often target the resources, availability, and operation of online services. With an increasing number of services relying on online resources, integrating proper security measures therefore becomes integral to ensure the correct functioning of every online service. In this course, we discuss important theoretical and analytical aspects in system security. The focus of the course is to understand basic attack strategies on modern systems and platforms, with a focus on side-channel attacks, software-based attacks, malware analysis, as well as software-based defenses (e.g., address space randomization and non-executable memory) and hardware-based defenses (e.g., using TPMs and TEEs). Other topics of the course include analyzing the security of modern cryptocurrencies and ML platforms, and similar aspects in system security. An integral part of this course are exercises and homeworks, which aim to deepen the understanding of the material with practical examples.					
Lehrformen					

Prüfungsformen**Voraussetzungen für die Vergabe von Credits****Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)**

5/150: B.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

5/149: B.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/105: M.Sc. Angewandte Informatik

5/99: M.Sc. IT-Sicherheit/ Netze und Systeme [PO 22]

5/96: M.Sc. IT-Sicherheit/ Netze und Systeme [PO 20]

Titel des Moduls: Vertiefungspraktikum Informatik					
Modul-Nr./Code	Credits 3 CP	Workload 90 h	Semester 5	Turnus Wintersemester oder Sommersemester	Dauer 1 Semester
Lehrveranstaltungen In jedem Semester wird eine wechselnde Auswahl an Praktika bereitgestellt. Die zugeordneten Praktika können im Vorlesungsverzeichnis eingesehen werden.			Kontaktzeit Je nach Veranstaltungswahl	Selbststudium Abhängig von Praktikumswahl	Gruppengröße Studierende
Unterrichtssprache Abhängig von Praktikumswahl: Deutsch oder Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Studiendekan der Informatik Lehrende: siehe jeweiligen Praktikumseintrag					
Verwendung des Moduls B.Sc. Informatik					
Vorkenntnisse Programmieren, ggf. weitere Vorkenntnisse abhängig vom Praktikum					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • haben Studierende ihre Fähigkeiten im Programmieren in einem Forschungs- oder Anwendungsbereich vertieft und erweitert • können Studierende Softwarebibliotheken zur Lösung forschungsnaher Fragestellungen anwenden • je nach gewählten Praktikum können noch weitere Lernziele dazu kommen 					
Inhalt Es werden Praktika zu verschiedenen relevanten Themen angeboten wie z.B. Praktikum Systemsoftware-Technik, Android App Evolution, Python programming and basic machine learning. Weiterführende Informationen zu den jeweiligen Praktika finden Sie im Vorlesungsverzeichnis.					
Lehrformen Praktikum im Block oder als semesterbegleitende Veranstaltung.					
Prüfungsformen Praktikum					
Voraussetzungen für die Vergabe von Credits Erfolgreiche Teilnahme am Praktikum					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) Informatik PO20 : 3/165 Informatik PO22 : unbenotet					

Titel des Moduls: Vertiefungsseminar Informatik					
Modul-Nr./Code	Credits 3 CP	Workload 90 h	Semester 5	Turnus Jedes Semester	Dauer 1 Semester
Lehrveranstaltungen In jedem Semester wird eine wechselnde Auswahl an Seminaren bereitgestellt. Die zugeordneten Seminare können im Vorlesungsverzeichnis eingesehen werden.			Kontaktzeit 30 h	Selbststudium 60 h	Gruppengröße Studierende
Unterrichtssprache Je nach Seminarwahl: Deutsch oder Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Studiendekan Informatik Lehrende: siehe jeweilige Seminare					
Verwendung des Moduls <ul style="list-style-type: none"> • Bachelor Informatik (PO 22) • Bachelor Informatik (PO 20) 					
Vorkenntnisse Die Vertiefungsseminare beziehen sich in der Regel auf Inhalte aus bestimmten Pflicht- oder Vertiefungsmodulen, die im Vorfeld absolviert worden sein sollten.					
Lernziele (learning outcomes) Nach dem erfolgreichen Abschluss des Moduls <ul style="list-style-type: none"> • verfügen Studierende über vertiefte wissenschaftliche Kenntnisse in dem ausgewählten Seminarthema • haben Studierende das Halten eines wissenschaftlichen Vortrags praktisch eingeübt und können Forschungsergebnisse eigenständig in einem didaktisch wohl aufbereiteten Vortrag vermitteln • können die Teilnehmer konstruktives Feedback formulieren und entgegennehmen • können Studierende eine schriftliche Ausarbeitung zu ihrem Seminarvortrag verfassen 					
Inhalt Es werden Bachelorseminare zu mehreren relevanten Themen angeboten, wie beispielsweise zu maschinellem Lernen, Algorithmen, theoretischer Informatik oder zu Ingenieurinformatik. Von den angebotenen Themen wählen die Studierenden abhängig von den eigenen Interessen und den individuellen Vertiefungswünschen ein Thema aus. Dieses sollen die Studierenden selbstständig bearbeiten. Dazu gehören die Literaturrecherche, die Einarbeitung in das Thema und schließlich die Präsentation. Nähere Informationen sind zu den jeweiligen Seminaren im Vorlesungsverzeichnis zu entnehmen.					
Lehrformen Seminar					
Prüfungsformen Seminarvortrag					
Voraussetzungen für die Vergabe von Credits Der Seminarvortrag muss mindestens mit der Note „ausreichend“ bewertet sein. Um die Lernziele zu erreichen, besteht im Seminar Anwesenheitspflicht an mindestens 9 von 10 Terminen. Mehrfaches Fehlen muss durch ein ärztliches Attest entschuldigt werden. Die Anwesenheit beim ersten Termin ist obligatorisch, da zu diesem Termin die Themen verteilt werden. Das Seminar gilt als nicht bestanden, wenn an mehr als einem Termin unentschuldigt gefehlt wurde.					

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

3/158: B.Sc. Informatik [PO 22]

3/165: B.Sc. Informatik [PO 20]

Titel des Moduls: Web-Engineering Web-Engineering					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester Siehe Prüfungsordnung	Turnus Sommersemester	Dauer 1 Semester
Lehrveranstaltungen Web-Engineering (128968 + 128969)			Kontaktzeit 60 h	Selbststudium 90 h	Gruppengröße 200 Studierende
Unterrichtssprache Deutsch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Prof. Dr.-Ing. Markus König Lehrende: Prof. Dr.-Ing. Markus König Stephan Embers, M.Sc.					
Verwendung des Moduls B.Sc. Angewandte Informatik B.Sc. Informatik M.Sc. IT-Sicherheit/ Informationstechnik					
Vorkenntnisse Programmieren, Architektur von Web-basierten Systemen					
Lernziele (learning outcomes) Die Entwicklung von Web-Anwendungen und Web-Services ist zentraler Bestandteil der Digitalisierung. Ziel der Lehrveranstaltung ist die Vermittlung von Grundlagen und bewährten Verfahren in der Web-Entwicklung. Studierende lernen konzeptuelle technologische Bausteine kennen: Transportverfahren, Webseitendarstellung, dynamische Web-Anwendungen und Web-Services. Über das konzeptuelle Verständnis hinaus werden praktische Kompetenzen vermittelt. Dazu werden moderne Werkzeuge der Web-Entwicklung, sowohl server- als auch clientseitig, vorgestellt und in den Übungssitzungen praktisch vertieft. Während der Umsetzung einfacher Web-Anwendungen stehen auch analytische Fähigkeiten im Fokus: Studierende werden befähigt, verschiedene Verfahren in Hinblick auf Performanz und Wartbarkeit zu bewerten. Diese Fähigkeiten sind in der kritischen Planungsphase von Software-Projekten unerlässlich. Nach dem erfolgreichen Abschluss des Moduls: <ul style="list-style-type: none"> • kennen Studierende gängige Konzepte der Web-Entwicklung in den Aspekten Präsentation, Transport und Bereitstellung von Daten • beherrschen Studierende grundlegende Fähigkeiten in Webseitendarstellung, dynamischen Web-Anwendungen und modernen Services (Node.js) 					
Inhalt Im Rahmen des Modules werden den Studierenden aktuelle Techniken und Kenntnisse im Bereich der Web-Entwicklung aufgezeigt. Thematisch wird der Bereich der server- und clientseitigen Entwicklung abgedeckt. JavaScript stellt dabei eine zentrale Rolle dar. Folgende Lehrinhalte werden behandelt: <ul style="list-style-type: none"> • Einführung in clientseitige Web-Entwicklung: HTML, CSS, JavaScript, Web Components • Transportverfahren und deren Nutzung: Representational State Transfer (REST), Asynchronous JavaScript und XML (AJAX) • Serverseitige Entwicklung mit Node.js und weiterführende Technologien 					
Lehrformen Synchrone Onlinevorlesung, Tutorien als seminaristischer Unterricht, zusätzlich Selbststudium mit ergänzend					

bereitgestellten Materialien und Aufgaben.

Prüfungsformen

Schriftliche Modulabschlussprüfung (120 Minuten)

Voraussetzungen für die Vergabe von Credits

Bestandene schriftliche Modulabschlussprüfung

Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS)

5/168: B.Sc. Angewandte Informatik [PO 22]

5/170: B.Sc. Angewandte Informatik [PO 20]

5/158: B.Sc. Informatik [PO 22]

5/165: B.Sc. Informatik [PO 20]

5/91: M.Sc. IT-Sicherheit/ Informationstechnik [PO 22]

5/84: M.Sc. IT-Sicherheit/ Informationstechnik [PO 20]

Titel des Moduls: Freies Wahlmodul					
Modul-Nr./Code	Credits 5 CP	Workload 150 h	Semester	Turnus Jedes Semester	Dauer 1 Semester
Lehrveranstaltungen			Kontaktzeit abhängig von Veranstaltungswahl	Selbststudium Je nach Veranstaltungswahl	Gruppengröße Studierende
Unterrichtssprache Je nach Veranstaltungswahl			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Studienfachberatung Informatik Lehrende:					
Verwendung des Moduls					
Vorkenntnisse abhängig von Veranstaltungswahl					
Lernziele (learning outcomes) Die Teilnehmer erwerben so genannte Schlüsselfähigkeiten in den freien Wahlfächern					
Inhalt Studierende müssen Veranstaltungen im Gesamtumfang von 5 CP absolvieren. Je nach Veranstaltungswahl werden unterschiedliche Inhalte vermittelt. Studierende haben die Möglichkeit unter den Freien Wahlmodulen auch Fächer jenseits der Informatik zu absolvieren und ihre Soft Skills zu erweitern. Z.B. Die freien Wahlfächer erweitern die Soft Skills. Z.B. wird die englische Fachsprache verbessert, in die Grundlagen der Rechtswissenschaften eingeführt oder Grundkenntnisse der Betriebswirtschaft vermittelt. Die Studierenden haben die Möglichkeit eine Auswahl entsprechend der eigenen Interessen zu treffen.					
Lehrformen					
Prüfungsformen abhängig von Veranstaltungswahl					
Voraussetzungen für die Vergabe von Credits abhängig von Veranstaltungswahl					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) 5/180: B.Sc. Informatik [PO 22] - unbenotet					

Titel des Moduls: Praktische Ausbildung					
Modul-Nr./Code	Credits 15 CP	Workload 300 h - 450 h (10 -15 CP)	Semester 5	Turnus Jedes Winter und Sommersemester	Dauer 1 Semester
Lehrveranstaltungen <ul style="list-style-type: none"> • Industriepraktikum [10 - 15 CP] • Software-Projekt [10 CP] • Forschungsprojekt [10 CP] 			Kontaktzeit	Selbststudium 270 h	Gruppengröße Studierende
Unterrichtssprache Deutsch / Englisch			Teilnahmevoraussetzungen		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Studiendekan*in Informatik Lehrende:					
Verwendung des Moduls Bachelor Informatik					
Vorkenntnisse					
Lernziele (learning outcomes) Nach erfolgreichem Abschluss des Moduls <ul style="list-style-type: none"> • sind die Studierenden in der Lage, das erlernte Fachwissen anzuwenden • haben die Studierenden die Software-Entwicklungscompetenz maßgeblich ausgebaut, insbesondere im Hinblick auf größere Software-Systeme • haben die Studierenden zusätzliche Fachkompetenz gemäß der jeweiligen projektspezifischen Aufgabenstellung erworben • können die Studierenden eigene Lösungsstrategien erarbeiten • haben die Studierenden die Fähigkeit zur Arbeitsteilung und Zusammenarbeit im Team verbessert (Teamfähigkeit und Projektorganisation) • haben die Studierenden die Kompetenz im Hinblick auf die Dokumentation von der erstellten Software, aber auch im Hinblick auf die Dokumentation der eigenen Projektarbeit gefestigt • haben die Studierenden die Kompetenz im Hinblick auf die Präsentation von Projektergebnissen verbessert • haben die Studierenden Erfahrung im Bewerbungsprozess gesammelt und sind auf das Berufsleben gut vorbereitet 					
Inhalt Während der praktischen Ausbildung sollen verschiedene Arbeitsgebiete, die im Zusammenhang mit der späteren Tätigkeit einer Informatikerin bzw. eines Informatikers stehen, bearbeitet werden. Im Vordergrund soll die Entwicklung größerer Software- oder anderer IT-Systeme stehen.					
Lehrformen Projektarbeit					
Prüfungsformen Projektarbeit mit tabellarischer Dokumentation					
Voraussetzungen für die Vergabe von Credits Erfolgreich abgeschlossenes Projekt und positiv bewertete abgegebene Dokumentation					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) Dieses Modul wird mit einem Leistungsnachweis (erfolgreich bestanden/nicht bestanden) abgeschlossen und wird nicht benotet.					

Titel des Moduls: Abschlussarbeit					
Modul-Nr./Code	Credits 15 CP	Workload 450 h	Semester	Turnus Jedes Semester	Dauer Semester
Lehrveranstaltungen <ul style="list-style-type: none"> a) Bachelor-Thesis (12 CP) b) Colloquium (3 CP) 			Kontaktzeit 15 h	Selbststudium	Gruppengröße Studierende
Unterrichtssprache Englisch			Teilnahmevoraussetzungen Mindestens 135 absolvierte Leistungspunkte mit abgeschlossenen Modulen der ersten vier Semester		
Modulbeauftragte/r und hauptamtlich Lehrende Modulbeauftragte/r: Modulbeauftragte/r: Studiendekan*in Informatik Erstbetreuer*in: Jede/r am Studiengang beteiligte Hochschullehrer*in (s. Prüfungsordnung für die Regularien) Lehrende:					
Verwendung des Moduls <ul style="list-style-type: none"> Bachelor Informatik [PO 22] Bachelor Informatik [PO 20] 					
Vorkenntnisse					
Lernziele (learning outcomes) Die Bachelorarbeit soll zeigen, dass die oder der Studierende in der Lage ist, innerhalb einer vorgegebenen Frist eine anspruchsvolle Fragestellung der Informatik unter Anwendung der im Bachelorstudium erworbenen Methoden selbstständig zu bearbeiten. Darüber hinaus wird der Erwerb von Grundkenntnissen der wissenschaftlichen Arbeit einschließlich der Projektorganisation sowie die Präsentation der erarbeiteten Ergebnisse erwartet. Während der Bachelorarbeit werden die folgenden Kompetenzen erworben bzw. ausgebaut: <ul style="list-style-type: none"> Vertieftes Wissen im Bereich der bearbeiteten Aufgabenstellung Wissenschaftliches Arbeiten und Schreiben Projekt- und Zeitmanagement Präsentation wissenschaftlicher Ergebnisse Rhetorik und sprachliche Kompetenz Fächerübergreifendes Denken und Arbeiten 					
Inhalt a) Bearbeitung und Lösung einer wissenschaftlichen Aufgabe im Bereich der Informatik unter Anleitung. Die im Bachelorstudium erworbenen Kenntnisse, Kompetenzen und Methoden sollen angewendet werden. Die Ergebnisse der Arbeit sind schriftlich zu verfassen. b) Im Anschluss an die Bearbeitung der Bachelorarbeit werden die Ergebnisse in Form eines Kolloquium-Vortrags präsentiert.					
Lehrformen Projektarbeit					
Prüfungsformen Schriftliche Ausarbeitung der gestellten Aufgabe und Präsentation der Ergebnisse im Kolloquium					
Voraussetzungen für die Vergabe von Credits Positive Bewertung der Bachelorarbeit und des Kolloquiums					
Stellenwert der Note für die Endnote (bei einem Gesamtstudienumfang von 180 ECTS) <ul style="list-style-type: none"> Bachelor Informatik [PO 22] : 15/158 					

- Bachelor Informatik [PO 20] : 15/170