Hochschule Niederrhein
University of Applied Sciences

Ingenieurwissenschaften und Informatik
Faculty of Engineering and Computer Science

Bachelor Informatik: Modulhandbuch

Bachelor Informatik (PO2027)

Herausgeber Fachbereich Ingenieurwissenschaften und Informatik
Stand 2026-08-10

Inhaltsverzeichnis

Einleitung

Dieses Modulhandbuch beschreibt den Bachelorstudiengang Informatik. Es macht transparent, welche fachlichen, methodischen und überfachlichen Kompetenzen die Studierenden im Studienverlauf erwerben und wie diese Kompetenzen in Grundlagenmodulen, Lernpfaden, Wahlpflichtmodulen, Projekten, Praxisphase und Bachelorarbeit aufgebaut, angewendet und vertieft werden. Das Modulhandbuch dient Studierenden, Lehrenden und externen Interessierten als Orientierung über Aufbau, Ziele und Inhalte des Studiums.

Studiengangsprofil

Informatik besitzt eine hohe strategische Bedeutung für Region, Gesellschaft und Wirtschaft. Digitale Technologien prägen nahezu alle Lebens- und Arbeitsbereiche und sind eine zentrale Voraussetzung für Innovation, Wettbewerbsfähigkeit und gesellschaftliche Teilhabe. Informatikerinnen und Informatiker entwickeln Software, digitale Systeme und datenbasierte Anwendungen, gestalten Prozesse effizienter und erschließen neue Geschäftsmodelle, Dienstleistungen und Anwendungsfelder. Damit leisten sie einen wesentlichen Beitrag zur digitalen Transformation von Unternehmen und öffentlichen Einrichtungen sowie zur Stärkung regionaler und überregionaler Wirtschaftsstrukturen.

Die rasante Entwicklung künstlicher Intelligenz verändert das Berufsbild der Informatik grundlegend. KI-gestützte Werkzeuge übernehmen zunehmend Aufgaben bei Codegenerierung, Dokumentation, Qualitätssicherung, Fehleranalyse und Softwarewartung. Dadurch verliert die manuelle Erstellung einzelner Programmteile relativ an Bedeutung, während die Fähigkeit wichtiger wird, komplexe Softwaresysteme zu entwerfen, Anforderungen präzise zu formulieren, KI-generierte Ergebnisse kritisch zu prüfen und technische Entscheidungen in größere fachliche, wirtschaftliche und gesellschaftliche Zusammenhänge einzuordnen. Softwareentwicklung wird damit stärker zu einer integrierenden, steuernden und bewertenden Tätigkeit.

Der Bachelorstudiengang Informatik greift diesen Wandel auf, ohne bewährte Grundlagen aufzugeben. Algorithmen und Datenstrukturen, Programmierung, Software Engineering, Datenbanken, Rechnernetze, IT-Sicherheit, mathematische und theoretische Informatik bleiben unverzichtbare Bestandteile des Studiums. Sie werden jedoch durchgängig mit neuen Kompetenzanforderungen verknüpft: dem reflektierten Einsatz generativer KI, der Entwicklung und Integration KI-basierter Systeme, dem Umgang mit Daten und Modellen, der Bewertung automatisiert erzeugter Lösungen sowie Fragen der Sicherheit, Nachvollziehbarkeit, Qualität, Nachhaltigkeit und Verantwortung.

Zugleich kommt der Informatik eine Schlüsselrolle bei gesellschaftlichen und globalen Herausforderungen zu, etwa in Klimaschutz, Energieversorgung, Mobilität, Gesundheit, Bildung, Verwaltung und industrieller Transformation. Die zunehmende Vernetzung, Automatisierung und datenbasierte Entscheidungsfindung eröffnet große Gestaltungsmöglichkeiten, wirft aber auch Fragen nach Datenschutz, Cybersicherheit, Transparenz, Diskriminierung, Nachhaltigkeit und gesellschaftlicher Verantwortung auf. Deshalb verbindet der Studiengang fachliche Tiefe mit systematischem und abstraktem Problemlösen, Kreativität, Urteilsfähigkeit, interdisziplinärem Denken sowie Kommunikations- und Teamfähigkeit.

Kurzporträt

Worum geht es? Informatik verbindet Programmierung, Software Engineering, Daten, Künstliche Intelligenz, mathematisch-formale Grundlagen, Systeme, Netze, IT-Sicherheit und projektorientierte Entwicklung. Studierende lernen, digitale Systeme zu analysieren, zu entwerfen, umzusetzen, zu bewerten und verantwortungsvoll weiterzuentwickeln.
Wofür braucht man es? Informatikkompetenzen werden benötigt, um robuste Software- und IT-Systeme zu entwickeln, Daten und KI-gestützte Werkzeuge produktiv einzusetzen, technische Risiken einzuschätzen und digitale Lösungen in Unternehmen, Verwaltung, Forschung und Gesellschaft wirksam zu gestalten.
Wieso ist es interessant? Informatik macht abstrakte Ideen praktisch wirksam. Sie verbindet kreatives Problemlösen mit präzisem Denken und eröffnet die Möglichkeit, digitale Werkzeuge, Anwendungen und Infrastrukturen aktiv mitzugestalten, statt sie nur zu nutzen.

Studiengangskonzept

Der Bachelorstudiengang Informatik umfasst 180 ECTS und wird in verschiedenen Studienvarianten angeboten. Neben der klassischen Vollzeitvariante mit einem Studienumfang von 30 ECTS pro Semester gibt es eine Teilzeitvariante mit reduzierter Arbeitsbelastung von 20 ECTS pro Semester und einer entsprechend verlängerten Regelstudienzeit von neun Semestern. Dadurch wird insbesondere Studierenden mit paralleler Ausbildung oder Berufstätigkeit ein planbarer und studierbarer Verlauf ermöglicht. Der Studiengang knüpft damit an die lange Tradition kooperativer Hochschulausbildung an, deren Organisationsform auch als Krefelder Modell bekannt ist.

Das strukturelle Grundkonzept des Studiengangs gliedert das Studium in mehrere Studienphasen. Zu Beginn durchlaufen die Studierenden die Studienphase A, die in die Informatik einführt und grundlegende Kompetenzen in Programmierung, Mathematik, Systemtechnik, Datennetzen, IT-Sicherheit und projektbasiertem Arbeiten aufbaut. Daran schließt sich der fachliche Kern des Studiengangs an: drei Pflicht-Modulbereiche in Form von Lernpfaden sowie ein Wahlpflichtbereich zur individuellen fachlichen Profilbildung. Das Studium schließt mit einem anwendungsbezogenen Projekt, der Praxisphase und der Bachelorarbeit ab; zugleich können Studierende ihr Studium in dieser finalen Phase durch einen Auslandsaufenthalt gezielt international ausrichten.

Die Neukonzeption des Studiengangs erfolgt im Kontext des von der Stiftung Innovation in der Hochschullehre bis mindestens Ende 2029 geförderten Projekts Dein Weg - Lernhindernisse überwinden in der Förderlinie Lehrarchitektur. Ziel des Projekts ist es, die Studienbedingungen in den Technikwissenschaften durch didaktische und strukturelle Maßnahmen systematisch zu verbessern. Gerade Grundlagen- und Bezugsfächer werden von Studierenden häufig als Hürde erlebt und können zu Verzögerungen oder Studienabbrüchen beitragen. Die Studiengangsentwicklung zielt daher darauf ab, Inhalte aus Grundlagen- und Bezugsfächern stärker mit den Kernfächern zu verknüpfen und eine rein disziplinorientierte Abfolge zugunsten eines kohärenteren Kompetenzaufbaus weiterzuentwickeln.

Flankierend dazu werden projektorientierte Lehr- und Lernformate durchgängig im Studienverlauf verankert. Studierende sollen von Beginn an disziplinübergreifend arbeiten und Zukunftskompetenzen wie Teamarbeit, Problemlösen, Kommunikation und Projektorganisation erwerben. Darüber hinaus soll die zeitliche Struktur von Lehrveranstaltungen stärker an den Lernbedarfen der Studierenden ausgerichtet werden, um Alternativen zu historisch gewachsenen, stark fragmentierten Lehrformaten zu etablieren.

Einführung und Grundlagen (Studienphase A). Im Zentrum der Studienphase A stehen Lehrangebote, die einerseits Informatik-Basiskenntnisse in Programmierung, Rechnerarchitektur, Rechnernetzen, Systemsoftware und IT-Sicherheit vermitteln und andererseits die Grundlage für projektbasiertes Arbeiten im weiteren Studienverlauf legen. Die Module Hello World 1 und Hello World 2 unterstützen dabei den Einstieg in informatische Arbeitsweisen, technische Selbstwirksamkeit, Teamarbeit, Dokumentation und Präsentation. Ergänzend werden mathematische Kompetenzen aufgebaut, die für das Informatikstudium wesentlich sind.

Alle Lehrveranstaltungen dieser Studienphase werden in jedem Semester angeboten. Dadurch wird ein Studienstart sowohl im Winter- als auch im Sommersemester ermöglicht. Zugleich unterstützt die doppelte Angebotsstruktur Studierende bei einem verzögerten oder erschwerten Studienbeginn, etwa bei später Einschreibung oder Herausforderungen beim Übergang von Schule zu Hochschule, weil ein Neustart bereits im folgenden Semester möglich ist. Die Lehrveranstaltungen können dadurch konsequent auf Studienanfänger:innen zugeschnitten werden. Neben der fachlichen Einführung wird die Enkulturation, also das Ankommen in Fachkultur, Arbeitsweisen und Studienorganisation, als explizites Element der Studienphase A verankert.

Lernpfade (Studienphase B). Die Studienphase B bildet den fachlich-inhaltlichen Kern des Studiums. Sie umfasst die Pflichtmodule, die in thematisch abgegrenzten Lernpfaden organisiert sind. Jeder Lernpfad orientiert sich an einer Teilmenge der Studiengangsziele und bündelt inhaltlich zusammenhängende Lehrveranstaltungen in einer Form, die für alle Studienvarianten gleich ist. Dadurch wird eine enge inhaltliche Verzahnung innerhalb eines Lernpfads ermöglicht: Grundlagen können im Curriculum konsequent in zeitlicher Nähe zu ihrer Anwendung positioniert werden.

In Analogie zur Modularisierung aus der Softwaretechnik strukturieren Lernpfade das Curriculum in klar definierte Teilbereiche mit eindeutigen Themen und definierten Schnittstellen. Alle Lernpfade folgen einer einheitlichen Struktur. Sie bestehen jeweils aus fünf Modulen, die sich über drei Semester verteilen: Zu Beginn stehen zwei Module, die fachliche Grundlagen aus komplementären Perspektiven legen. Es folgen zwei Module zur Vertiefung und Systematisierung der Inhalte. Den Abschluss bildet ein Capstone-Modul, das die Inhalte des Lernpfads in einer größeren, zusammenhängenden Aufgabenstellung integriert.

Die Capstone-Module verknüpfen zentrale Schlüsselkompetenzen wie Projektorganisation, Teamarbeit, Kommunikation und Dokumentation unmittelbar mit den Fachinhalten. Diese Kompetenzen werden nicht als separate Add-ons in eigenständigen Modulen behandelt, sondern innerhalb fachlicher Kontexte systematisch mitentwickelt. Die Lernpfadstruktur ist zudem so angelegt, dass Lernpfade flexibel im Studienverlauf angeordnet werden können. Dadurch lassen sich unterschiedliche Studienvarianten realisieren, ohne dass Lehrveranstaltungen gedoppelt werden müssen.

Zudem ermöglichen Wahlpflichtmodule eine individuelle Profilbildung. Sie vertiefen unter anderem Themen aus eingebetteten Systemen, Robotik, Cyber-Physical Systems, IT-Sicherheit, fortgeschrittener Programmierung, Informations- und Prozessmanagement sowie aktuellen Anwendungsfeldern der Informatik.

Projekt, Praxisphase und Abschluss (Studienphase C). Das anwendungsbezogene Informatik-Projekt, Praxisphase und Bachelorarbeit führen die erworbenen Kompetenzen schließlich in größeren anwendungsbezogenen, berufspraktischen und eigenständigen Aufgaben zusammen.

Qualifikationsziele

Die Qualifikationsziele beschreiben, welche fachlichen, methodischen, sozialen und kommunikativen Kompetenzen die Studierenden im Studienverlauf aufbauen und in den Modulen anwenden, vertiefen und integrieren.

Ziel	Qualifikationsziel
Z1	Grundlegende Konzepte, Prinzipien und Denkweisen der Informatik, insbesondere Abstraktion, Modellbildung, Dekomposition und algorithmisches Denken, zur Lösung praxisrelevanter Problemstellungen anwenden.
Z2	Mathematische, theoretische und technische Grundlagen der Informatik für die Modellierung, Entwicklung und Bewertung informatischer Systeme nutzen.
Z3	Algorithmen, Datenstrukturen und Programmierparadigmen problemangemessen auswählen, anwenden und bewerten.
Z4	Praxisrelevante, robuste, wartbare und erweiterbare Softwaresysteme sowie vernetzte Anwendungen unter Berücksichtigung von Anforderungen an Qualität und Sicherheit spezifizieren, entwerfen, implementieren, testen, dokumentieren, betreiben und weiterentwickeln.
Z5	Methoden des Software Engineering, der Qualitätssicherung, des Projektmanagements sowie moderner Entwicklungsprozesse einschließlich Automatisierung, Debugging, Continuous Integration und Deployment zielgerichtet anwenden.
Z6	Technologien, Werkzeuge, Architekturen, Softwarekomponenten und IT-Systeme nach fachlichen, technischen, wirtschaftlichen, sicherheitsbezogenen und qualitativen Kriterien auswählen, konfigurieren, einsetzen und eigene Lösungen begründet beurteilen.
Z7	Daten erfassen, aufbereiten und analysieren sowie grundlegende Methoden der Künstlichen Intelligenz und Data Science und KI-gestützte Werkzeuge verantwortungsvoll, sicher und produktiv einsetzen.
Z8	Fachliche Aufgaben selbständig und im Team bearbeiten, auch in interdisziplinären Kontexten zusammenarbeiten, Projektergebnisse adressatengerecht kommunizieren und technische Entscheidungen argumentativ vertreten.
Z9	Gesellschaftliche, ethische, rechtliche, ökologische und sicherheitsbezogene Auswirkungen informatischer Systeme reflektieren, die eigene Arbeitsweise weiterentwickeln und sich neue Methoden, Technologien, Werkzeuge sowie bestehende Systeme selbständig erschließen.

Ziele-Module-Matrix

Die Matrix zeigt, in welchen Modulen die übergreifenden Qualifikationsziele aufgebaut und vertieft werden. A bedeutet Anwendung und Vertiefung, B kennzeichnet Transfer und Beherrschung auf einem höheren Integrationsniveau. x bedeutet abhängig vom konkreten Wahlpflichtmodul.

Legende: Z1 = Konzepte und Denkweisen der Informatik anwenden; Z2 = mathematische, theoretische und technische Grundlagen nutzen; Z3 = Algorithmen, Datenstrukturen und Programmierparadigmen auswählen, anwenden und bewerten; Z4 = Softwaresysteme und vernetzte Anwendungen spezifizieren, entwerfen, implementieren, testen, dokumentieren, betreiben und weiterentwickeln; Z5 = Software Engineering, Qualitätssicherung, Projektmanagement und moderne Entwicklungsprozesse anwenden; Z6 = Technologien, Werkzeuge, Architekturen, Softwarekomponenten und IT-Systeme auswählen, konfigurieren, einsetzen und beurteilen; Z7 = Daten, KI, Data Science und KI-gestützte Werkzeuge verantwortungsvoll einsetzen; Z8 = selbständig und im Team arbeiten, kommunizieren und technische Entscheidungen vertreten; Z9 = gesellschaftliche, ethische, rechtliche, ökologische und sicherheitsbezogene Auswirkungen reflektieren und sich Neues selbständig erschließen. A = vertiefen/anwenden; B = beherrschen/transferieren; x = abhängig vom konkreten Wahlpflichtmodul.

Modul	Z1	Z2	Z3	Z4	Z5	Z6	Z7	Z8	Z9
BIN001: Hello World 1 - Ankommen, Ausprobieren, Umsetzen	A	A		A	A	A		A	A
BIN002: Hello World 2 - Ideen entwickeln, Lösungen gestalten	A			A	A	A		A	A
BIN003: Mathematik-Grundlagen der Informatik	A	A							A
BIN004: Einführung in die Programmierung	A	A	A	A		A			A
BIN005: Systemsoftware	A	A	A	A		A			A
BIN006: Systemhardware	A	A				A			A
BIN007: Datennetze	A	A		A		A			A
BIN008: Grundlagen der IT-Sicherheit	A	A		A		A			A
BIN101: Data Engineering	A	A	A	A	A	A	B		A
BIN102: Datenanalyse und Statistik	A	A					B		A
BIN103: Machine Learning	A	A	A	A		A	B		A
BIN104: Multimodale KI-Systeme	A	A	A	A		A	B		A
BIN105: Capstone Daten und Künstliche Intelligenz	B	B	B	B	B	B	B	B	B
BIN201: Weiterführende Programmentwicklung	A	A	B	B	A	A			A
BIN202: DevOps	A		A	B	B	B		A	A
BIN203: UX und Web-Engineering	A		A	B	A	B		A	A
BIN204: Software Engineering	B		A	B	B	B		B	B
BIN205: Capstone Software-Entwicklung	B	B	B	B	B	B		B	B
BIN301: Logik, Diskrete Strukturen und Lineare Algebra	A	B							A
BIN302: Algorithmen und Datenstrukturen	B	B	B	A					A
BIN303: Theoretische Informatik	B	B	A						A
BIN304: Verteilte und parallele Programmierung	A	A	A	B		B			A
BIN305: Informatisches Denken und Konzepte - Capstone	B	B	B	A		A		B	B
BIN401: Wahlpflicht-Modul 1	x	x	x	x	x	x	x	x	x
BIN402: Wahlpflicht-Modul 2	x	x	x	x	x	x	x	x	x
BIN403: Wahlpflicht-Modul 3	x	x	x	x	x	x	x	x	x
BIN404: Wahlpflicht-Modul 4	x	x	x	x	x	x	x	x	x
BIN405: Capstone: Wahlpflicht	B	A	A	A	A	B		B	B
BIN501: Anwendungsbezogenes Informatik-Projekt	B	B	B	B	B	B	B	B	B
BIN601: Praxisphase	B	B	B	B	B	B	B	B	B
BIN602: Bachelorarbeit mit Kolloquium	B	B	B	B	B	B	B	B	B

Studienpläne

6	Praxisphase			Bachelorarbeit mit Kolloquium		S
5	Multimodale KI-Systeme	Capstone Daten und Künstliche Intelligenz	Anwendungsbezogenes Informatik-Projekt		Wahlmodul 4	Studienprofil - Capstone	W
4	Machine Learning	Software Engineering	Capstone Software-Entwicklung	Verteilte und parallele Programmierung	Informatisches Denken und Konzepte - Capstone	Wahlmodul 3	S
3	Data Engineering	Datenanalyse und Statistik	UX und Web-Engineering	Theoretische Informatik	Wahlmodul 1	Wahlmodul 2	W
2	Hello World 2 - Ideen entwickeln, Lösungen gestalten	Weiterführende Programmentwicklung	DevOps	Logik, Diskrete Strukturen und Lineare Algebra	Algorithmen und Datenstrukturen	Grundlagen der IT-Sicherheit	S
1	Hello World 1 - Ankommen, Ausprobieren, Umsetzen	Mathematik-Grundlagen der Informatik	Einführung in die Programmierung	Systemsoftware	Systemhardware	Datennetze	W

Abbildung 1: Studienplan Informatik, Vollzeit, Studienbeginn im Wintersemester

6	Praxisphase			Bachelorarbeit mit Kolloquium		W
5	Software Engineering	Capstone Software-Entwicklung	Anwendungsbezogenes Informatik-Projekt		Verteilte und parallele Programmierung	Informatisches Denken und Konzepte - Capstone	S
4	UX und Web-Engineering	Multimodale KI-Systeme	Capstone Daten und Künstliche Intelligenz	Wahlmodul 4	Studienprofil - Capstone	Theoretische Informatik	W
3	Weiterführende Programmentwicklung	DevOps	Machine Learning	Wahlmodul 3	Logik, Diskrete Strukturen und Lineare Algebra	Algorithmen und Datenstrukturen	S
2	Hello World 2 - Ideen entwickeln, Lösungen gestalten	Data Engineering	Datenanalyse und Statistik	Wahlmodul 1	Wahlmodul 2	Grundlagen der IT-Sicherheit	W
1	Hello World 1 - Ankommen, Ausprobieren, Umsetzen	Mathematik-Grundlagen der Informatik	Einführung in die Programmierung	Systemsoftware	Systemhardware	Datennetze	S

Abbildung 2: Studienplan Informatik, Vollzeit, Studienbeginn im Sommersemester

9	Praxisphase		Bachelorarbeit mit Kolloquium		W
8	Verteilte und parallele Programmierung	Informatisches Denken und Konzepte - Capstone	Anwendungsbezogenes Informatik-Projekt		S
7	Theoretische Informatik		Wahlmodul 4	Studienprofil - Capstone	W
6	Logik, Diskrete Strukturen und Lineare Algebra	Algorithmen und Datenstrukturen	Wahlmodul 3		S
5	Multimodale KI-Systeme	Capstone Daten und Künstliche Intelligenz	Wahlmodul 1	Wahlmodul 2	W
4	Machine Learning	Software Engineering	Capstone Software-Entwicklung	Grundlagen der IT-Sicherheit	S
3	Data Engineering	Datenanalyse und Statistik	UX und Web-Engineering	Systemhardware	W
2	Hello World 2 - Ideen entwickeln, Lösungen gestalten	Weiterführende Programmentwicklung	DevOps	Datennetze	S
1	Hello World 1 - Ankommen, Ausprobieren, Umsetzen	Mathematik-Grundlagen der Informatik	Einführung in die Programmierung	Systemsoftware	W

Abbildung 3: Studienplan Informatik, Teilzeit, Studienbeginn im Wintersemester

9	Praxisphase		Bachelorarbeit mit Kolloquium		S
8	Anwendungsbezogenes Informatik-Projekt		Wahlmodul 4	Studienprofil - Capstone	W
7	Verteilte und parallele Programmierung	Informatisches Denken und Konzepte - Capstone	Wahlmodul 3		S
6	Theoretische Informatik		Wahlmodul 1	Wahlmodul 2	W
5	Logik, Diskrete Strukturen und Lineare Algebra	Algorithmen und Datenstrukturen	Software Engineering	Capstone Software-Entwicklung	S
4	Grundlagen der IT-Sicherheit	Multimodale KI-Systeme	Capstone Daten und Künstliche Intelligenz	UX und Web-Engineering	W
3	Systemhardware	Machine Learning	Weiterführende Programmentwicklung	DevOps	S
2	Hello World 2 - Ideen entwickeln, Lösungen gestalten	Data Engineering	Datenanalyse und Statistik	Datennetze	W
1	Hello World 1 - Ankommen, Ausprobieren, Umsetzen	Mathematik-Grundlagen der Informatik	Einführung in die Programmierung	Systemsoftware	S

Abbildung 4: Studienplan Informatik, Teilzeit, Studienbeginn im Sommersemester

Modulbeschreibungen

Dieses Modulhandbuch beschreibt die Module des Bachelorstudiengangs Informatik mit Angaben zu Verwendbarkeit, Modultyp, Angebotshäufigkeit, Dauer, Credits, Benotung, Semesterwochenstunden, Workload, Voraussetzungen, Lernergebnissen, Inhalten, Prüfungsformen, Literatur und fachlicher Zuordnung. Die Modulbeschreibungen sind nach Modulbereichen, Lernpfaden, Wahlpflichtmodulen sowie Projekt-, Praxis- und Abschlussformaten geordnet und ermöglichen damit eine Orientierung über Studienverlauf, fachliche Profilbildung und Verzahnung der Module mit den Qualifikationszielen.

So liest man eine Modulbeschreibung

Jede Modulbeschreibung folgt dem gleichen Aufbau: Zuerst stehen organisatorische Angaben wie Verantwortliche, Credits, Semesterwochenstunden (SWS) und Workload. Die **Lernergebnisse** sind im Format **WAS** (Kompetenz nach Abschluss), **WOMIT** (Tätigkeiten und Arbeitsweisen, mit denen die Kompetenz erworben wird) und **WOZU** (Nutzen für Studium, Praxis und Beruf) beschrieben. Ein farbiges Abzeichen zeigt, ob ein Modul **Pflicht** oder **Wahl** ist. Am Ende jedes Moduls gibt ein **Kurzporträt** einen schnellen, allgemeinverständlichen Überblick. Unklare Abkürzungen wie ECTS, SWS oder V/Ü/P/S sind im Glossar am Ende des Handbuchs erklärt.

Einführung in die Informatik

Der Modulbereich Einführung in die Informatik legt die gemeinsame fachliche, technische und methodische Grundlage für das Informatikstudium. Die Studierenden entwickeln erste informatische Artefakte, erwerben mathematische und programmierbezogene Grundlagen und lernen zentrale Systemebenen von Hardware, Betriebssystemen, Netzen und IT-Sicherheit kennen.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Modulbereichs einfache informatische Problemstellungen strukturieren, erste technische Prototypen entwickeln, grundlegende mathematische und programmierbezogene Methoden anwenden sowie zentrale Komponenten von Rechnern, Betriebssystemen, Datennetzen und IT-Sicherheit fachlich einordnen. Sie entwickeln dabei Kompetenzen im projektorientierten Arbeiten, im systematischen Problemlösen, in technischer Dokumentation und in der Reflexion informatischer Systeme.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Modulbereich
  - in einführenden Projektformaten technische Systeme planen, prototypisch umsetzen, testen, dokumentieren und präsentieren,
  - mathematische Sprache, Analysis und lineare Algebra für informatische Aufgabenstellungen anwenden,
  - einfache Programme entwickeln, ausführen, testen und nachvollziehbar verbessern,
  - das Zusammenspiel von Programmen, Toolchains, Betriebssystemen und Hardware analysieren,
  - IP-basierte Datennetze konzipieren, in Laborumgebungen prüfen und grundlegende Kommunikationsabläufe erklären,
  - Sicherheitsanforderungen, Bedrohungen und grundlegende Schutzmaßnahmen für digitale Systeme einordnen.
- **WOZU**: Damit sie in den anschließenden Lernpfaden Software, Daten, KI, theoretische Konzepte, Systeme und Sicherheitsfragen fundiert bearbeiten können. Der Modulbereich unterstützt Studienorientierung, technische Selbstwirksamkeit und den Aufbau einer gemeinsamen fachlichen Sprache für projektorientierte, mathematische und systemnahe Aufgaben im weiteren Studium.

Kurzporträt Einführung in die Informatik

Worum geht es? Der Modulbereich führt in zentrale Denk- und Arbeitsweisen der Informatik ein. Die Studierenden entwickeln erste Prototypen, lernen Programmierung und mathematisches Arbeiten kennen und betrachten Rechner, Betriebssysteme, Netze und Sicherheit als zusammenhängende technische Grundlagen. Dadurch entsteht ein breiter Einstieg in das Fach.
Wofür braucht man es? Die Grundlagen werden in nahezu allen späteren Informatikmodulen benötigt. Sie helfen, Programme zu verstehen, technische Systeme zu analysieren, Datenkommunikation einzuordnen, Sicherheitsfragen zu erkennen und Projektaufgaben strukturiert zu bearbeiten. Zugleich schaffen sie Orientierung für die spätere Profilbildung.
Wieso ist es interessant? Informatik wird hier von Anfang an praktisch und systemisch erfahrbar. Die Studierenden bauen, testen, analysieren und reflektieren technische Lösungen und sehen, wie Software, Hardware, Netze, Mathematik und Sicherheit zusammenwirken. Das macht den Einstieg greifbar und zeigt die Breite des Studiengangs.

BIN001: Hello World 1 - Ankommen, Ausprobieren, Umsetzen

Modulverantwortung	Prof. Dr. Matteo Zella	**Lehrperson(en)**	Prof. Dr. Matteo Zella
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	unbenotet
SWS	1 V | - Ü | 3 P | - S	**Engl. Titel**	Hello World 1 - Arriving, Experimenting, Implementing
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, ein einfaches spielerisches technisches System iterativ zu entwickeln, als funktionsfähigen Prototyp umzusetzen und die dabei getroffenen technischen und organisatorischen Entscheidungen nachvollziehbar zu reflektieren.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- in Teams eine offene technische Challenge analysieren, Teilaufgaben identifizieren und eine einfache Umsetzungsstrategie entwickeln,
- grundlegende Elemente aus Programmierung, Sensorik, Aktorik und mechanischem Prototyping praktisch erproben und zu einem einfachen Gesamtsystem verbinden,
- einen Prototyp schrittweise aufbauen, testen, Fehler suchen, verbessern und für eine Demonstration vorbereiten,
- Arbeitsstände in Meilensteingesprächen vorstellen, Feedback aufnehmen und daraus nächste Entwicklungsschritte ableiten,
- einfache technische Entscheidungen, Probleme, Lösungswege und Lernerfahrungen schriftlich dokumentieren,
- den eigenen Lernprozess, den Umgang mit Unsicherheit sowie das Scheitern und Verbessern im Team reflektieren,
- grundlegende Werkzeuge und Arbeitsweisen wie einfache technische Dokumentation und Versionierung zur Zusammenarbeit im Projekt nutzen,
- die Wirkung technischer Systeme in ersten Ansätzen auch hinsichtlich Nutzbarkeit, Verantwortung und gesellschaftlicher Einordnung betrachten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um sich früh im Informatikstudium zu orientieren, technische Selbstwirksamkeit aufzubauen und grundlegende kooperative Arbeitsweisen für projektorientierte Folgemodule zu entwickeln. Zugleich verstehen sie technische Aufgaben als Zusammenspiel von Anforderungen, Umsetzung, Test, Teamarbeit und Reflexion.

Inhalte

- Ankommen im Informatikstudium und Orientierung in projektorientierten Arbeitsformen
- Informatik als gestaltende, technische und kooperative Disziplin
- Grundlegende Programmierkonzepte im Kontext eines einfachen technischen Systems
- Grundlagen des Aufbaus und Tests einfacher technischer Prototypen
- Debugging, Fehlersuche und iterative Verbesserung
- Versionierung und nachvollziehbare Ablage von Projektständen
- Einfache technische Dokumentation
- Teamarbeit, Aufgabenverteilung und Kommunikation in technischen Projekten
- Meilensteinbasierte Vorstellung und Diskussion von Arbeitsständen
- Demonstration eines funktionsfähigen Prototyps im abschließenden Wettbewerb
- Reflexion von Lernprozessen, Fehlern, Verbesserungen und Zusammenarbeit
- Erste Einordnung gesellschaftlicher und ethischer Aspekte technischer Systeme

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Testat. Das Prüfungskonzept besteht aus einer unbenoteten Projektarbeit mit Vorstellung des Projektstands in den vorgesehenen Meilensteingesprächen, funktionsfähigem Prototyp, Demonstration im abschließenden Wettbewerb, kurzer technischer Dokumentation und individueller Reflexion des Lernprozesses.

Literatur

- Don Norman: *The Design of Everyday Things*. Revised and Expanded Edition, Basic Books, 2013.
- Tom Kelley; David Kelley: *Creative Confidence*. Crown Business, 2013.
- Donald A. Schön: *The Reflective Practitioner*. Basic Books, 1983.

Kurzporträt BIN001: Hello World 1 - Ankommen, Ausprobieren, Umsetzen

Worum geht es? In diesem Modul kommen die Studierenden im Informatikstudium an und probieren erste technische Arbeitsweisen praktisch aus. Sie entwickeln gemeinsam ein kleines interaktives System und erleben Informatik als Zusammenspiel von Software, Hardware, Sensorik, Aktorik, einfacher Mechanik und Teamarbeit. Am Ende steht ein funktionsfähiger Prototyp, der in einem abschließenden Wettbewerb demonstriert wird.
Wofür braucht man es? Die erworbenen Kompetenzen bilden eine Grundlage für spätere Projekt-, Praktikums- und Entwicklungsaufgaben im Informatikstudium. Studierende lernen, technische Aufgaben nicht isoliert als Programmierprobleme zu betrachten, sondern Anforderungen, Umsetzung, Test, Zusammenarbeit und Dokumentation miteinander zu verbinden. Das erleichtert den Einstieg in projektorientierte Folgemodule.
Wieso ist es interessant? Informatik wird hier direkt erfahrbar: ausprobieren, Fehler finden, verbessern und gemeinsam etwas Vorzeigbares bauen. Wer Freude daran hat, im Team zu tüfteln und ein technisches System Schritt für Schritt zum Laufen zu bringen, bekommt einen praktischen und motivierenden Einstieg ins Studium.

BIN002: Hello World 2 - Ideen entwickeln, Lösungen gestalten

Modulverantwortung	Prof. Dr. Matteo Zella	**Lehrperson(en)**	Prof. Dr. Matteo Zella
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	unbenotet
SWS	- V | - Ü | 4 P | - S	**Engl. Titel**	Hello World 2 - Developing Ideas, Designing Solutions
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN001: Hello World 1 - Ankommen, Ausprobieren, Umsetzen: Teamarbeit, technisches Ausprobieren, einfache Prototypen, Dokumentation, Präsentation.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, aus einer technischen Idee ein nutzerorientiertes Produktkonzept mit funktionsfähigem Minimum Viable Prototype zu entwickeln.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine einfache Problemstellung aus Nutzer- oder Anwendungsperspektive beschreiben und daraus eine Produktidee ableiten,
- Zielgruppen, Nutzungssituationen und Nutzenversprechen eines einfachen informatischen Produkts analysieren,
- Anforderungen und User Stories formulieren, priorisieren und in ein umsetzbares technisches Grobkonzept überführen,
- einen funktionsfähigen Prototypen mit Software-, Hardware- oder Interaktionsanteilen planen, umsetzen und iterativ verbessern,
- agile Grundelemente wie Backlog, Meilensteine, Reviews, Feedbackschleifen und Retrospektiven in einem studentischen Teamprojekt anwenden,
- Peer-Feedback und fiktives Nutzerfeedback aufnehmen, bewerten und für die Weiterentwicklung des Prototyps nutzen,
- einfache wirtschaftliche und kommunikative Aspekte wie Ressourcenbedarf, Kosten-Nutzen-Überlegung, Produktbotschaft und Präsentationsform berücksichtigen,
- ihre Produktidee in einem frühen Pitch, in Meilensteingesprächen sowie in einer abschließenden Produktmesse darstellen und den Projektverlauf, die Teamarbeit und die getroffenen Entscheidungen reflektieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um spätere Software-, System- und Transferprojekte stärker an Nutzerbedarf, Umsetzbarkeit und nachvollziehbarer Kommunikation auszurichten. In späteren Studienprojekten, Praxisprojekten und Abschlussarbeiten unterstützt dies die Verbindung technischer Entwicklung mit Nutzerperspektive, Teamarbeit und Produktkommunikation.

Inhalte

- Von der technischen Idee zur Problemstellung
- Zielgruppen, Nutzungssituationen und Nutzenversprechen
- Anforderungen, Priorisierung und User Stories
- Minimum Viable Product und Minimum Viable Prototype
- Einfache Produkt- und Systemarchitektur
- Prototyping mit Software, Hardware oder interaktiven Systemen
- Agile Grundelemente in studentischen Projekten
- Feedbackmethoden, Peer-Feedback und simuliertes Nutzerfeedback
- Grundlagen nutzerorientierter Gestaltung und User Experience
- Einfache Tests und Demonstrationen von Prototypen
- Niedrigschwellige Kosten-, Ressourcen- und Nutzenbetrachtung
- Produktkommunikation, Storytelling, Pitch und Produktmesse

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Testat. Das Prüfungskonzept besteht aus einer unbenoteten Projektarbeit mit Prototyp, Pitch, Dokumentation und Reflexion. Es umfasst insbesondere einen frühen Produktpitch, Meilensteingespräche, Kurzpräsentationen, einen funktionsfähigen Prototypen, ein Produktposter, eine Landingpage oder ein Demo-Video, Peer-Feedback sowie einen Reflexionsbericht.

Literatur

- Eric Ries: *The Lean Startup*. Crown Business, 2011.
- Jake Knapp; John Zeratsky; Braden Kowitz: *Sprint*. Simon & Schuster, 2016.
- Jeff Patton; Peter Economy: *User Story Mapping*. O'Reilly Media, 2014.

Kurzporträt BIN002: Hello World 2 - Ideen entwickeln, Lösungen gestalten

Worum geht es? In diesem Modul entwickeln Studierende aus einer technischen Idee ein einfaches Produktkonzept. Dabei betrachten sie nicht nur die technische Funktion, sondern auch Zielgruppe, Nutzen, Feedback, Präsentation und grundlegende wirtschaftliche Überlegungen. Das Modul verbindet praktische Informatik mit produktorientiertem Denken.
Wofür braucht man es? Die Kompetenzen werden in späteren Studienprojekten, im Software Engineering, in Praxisprojekten und in Abschlussarbeiten benötigt. Dort sollen aus offenen Aufgabenstellungen tragfähige technische Lösungen entstehen, die verständlich begründet und adressatengerecht kommuniziert werden können. In der Berufspraxis unterstützt dies die Verbindung von Entwicklung, Nutzerperspektive und Produktkommunikation.
Wieso ist es interessant? Informatik besteht nicht nur daraus, dass technische Systeme funktionieren, sondern auch daraus, sinnvolle Lösungen für Menschen und Anwendungskontexte zu entwickeln. Studierende erleben, wie aus einer ersten Idee ein verständlicher Prototyp mit Zielgruppe, Nutzenversprechen und überzeugender Präsentation entsteht.

BIN003: Mathematik-Grundlagen der Informatik

Modulverantwortung	Prof. Dr. Ulrich Tipp	**Lehrperson(en)**	Prof. Dr. Ulrich Tipp Prof. Dr. Steffen Goebbels
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	3 V | 2 Ü | - P | - S	**Engl. Titel**	Mathematical Foundations of Computer Science
Workload	• Präsenz: 55 Std. • Selbst: 80 Std.		

Empfohlene Voraussetzungen

- Schulmathematik: elementare Algebra, Funktionen, Grundbegriffe der Geometrie.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, einfache mathematische Herleitungen aus Analysis und linearer Algebra nachzuvollziehen sowie Aufgaben in präziser mathematischer Sprache selbst zu formulieren und zu lösen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Problemstellungen in Begriffe, Symbole und Gleichungen überführen und Lösungswege strukturiert dokumentieren,
- grundlegende Verfahren der Analysis auf typische Aufgaben anwenden und Rechenschritte begründen,
- Konzepte der linearen Algebra zur Modellierung und Berechnung einfacher Zusammenhänge einsetzen,
- lineare Gleichungssysteme im Rahmen geometrischer Anwendungen aufstellen und lösen,
- Ergebnisse auf Korrektheit, Plausibilität und nachvollziehbare Darstellung prüfen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um formale Modelle und Herleitungen in nachfolgenden Informatikmodulen sicher zu verstehen und eigenständig für die Lösung konzeptioneller Aufgaben einzusetzen. Dies unterstützt insbesondere Module mit algorithmischen, datenbezogenen und theoretischen Anteilen.

Inhalte

- Mathematische Sprache, Begriffe und Notation
- Umformungen und strukturierte Lösungsdarstellung
- Funktionen, Folgen, Reihen und Grenzwerte
- Differentiation und Integration einer Variablen
- Vektorrechnung und geometrische Interpretation
- Matrizen als Rechenobjekte
- Lineare Gleichungssysteme als grundlegendes Lösungsverfahren
- Geraden, Ebenen, Winkel- und Abstandsberechnungen

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Klausurarbeit (90 Minuten).

Literatur

- Peter Hartmann: *Mathematik für Informatiker*. 7. Auflage, Springer Vieweg, 2019.
- Gerald Teschl; Susanne Teschl: *Mathematik für Informatiker*, Band 1 und 2. Springer, 2013.
- Steffen Goebbels; Jochen Rethmann: *Mathematik für Informatiker*. Springer Vieweg, 2014.

Kurzporträt BIN003: Mathematik-Grundlagen der Informatik

Worum geht es? Das Modul führt in mathematische Werkzeuge ein, mit denen Informatiker:innen Größen, Änderungen, Räume und einfache Modelle beschreiben. Im Mittelpunkt stehen symbolische Darstellungen, Analysis, lineare Algebra und geometrische Anwendungen. Die Studierenden üben, Rechenwege nachzuvollziehen, Ergebnisse zu interpretieren und mathematische Notation sicherer zu verwenden.
Wofür braucht man es? Diese Grundlagen helfen, technische Zusammenhänge in Datenanalyse, Grafik, Simulation, Optimierung und KI nicht nur zu benutzen, sondern rechnerisch einzuordnen. Sie schaffen eine gemeinsame Sprache für quantitative Beschreibungen und machen spätere Module zugänglicher, in denen Vektoren, Funktionen, Matrizen oder Änderungsraten auftauchen.
Wieso ist es interessant? Mathematik wird hier als Werkzeugkasten für informatische Anwendungen erfahrbar. Wer versteht, was Ableitungen, Vektoren oder lineare Gleichungssysteme ausdrücken, erkennt technische Modelle hinter Software, Daten und Grafik klarer. Das Modul zeigt an grundlegenden Beispielen, wie aus Rechnungen belastbare Aussagen werden.

BIN004: Einführung in die Programmierung

Modulverantwortung	Prof. Dr. Michael Gref	**Lehrperson(en)**	Prof. Dr. Michael Gref Prof. Dr. Duc Duy Pham
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Introduction to Programming
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, einfache Programme anhand gegebener Anforderungen und algorithmischer Lösungsansätze zu verstehen, zu erstellen, zu überprüfen und nachvollziehbar weiterzuentwickeln.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- grundlegende Konzepte einer höheren, dynamisch typisierten Programmiersprache anwenden und einfache Anforderungen in strukturierte Programme übertragen,
- geeignete Daten- und Sammlungstypen zur Verarbeitung einfacher Datenbestände einsetzen,
- vorhandenen Code lesen, das Programmverhalten nachvollziehen und Fehler lokalisieren,
- Programme mit einer zeitgemäßen Entwicklungsumgebung ausführen, untersuchen und verbessern,
- Funktionen und kleinere Programme durch einfache Tests überprüfen,
- grundlegende Prinzipien lesbarer und nachvollziehbarer Programmgestaltung anwenden,
- ausgewählte Bibliotheken, Standardfunktionen und grundlegende objektorientierte Konzepte zur Lösung einfacher Aufgaben verwenden,
- einfache Entwicklungsabläufe unter Nutzung einer Versionsverwaltung nachvollziehen und eigene Arbeitsergebnisse versionieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um in nachfolgenden Informatikmodulen programmierbezogene Aufgabenstellungen eigenständig bearbeiten, vorhandenen Code kritisch nachvollziehen und eigene Lösungen systematisch umsetzen zu können.

Inhalte

- Grundbegriffe der Programmierung und Programmausführung
- Entwicklungsumgebung, Debugger und Versionsverwaltung
- Variablen, Datentypen, Ausdrücke und Operatoren
- Ein- und Ausgabe, Zeichenketten und einfache Textverarbeitung
- Kontrollstrukturen und strukturierter Programmablauf
- Funktionen, Parameter, Rückgabewerte und Modularisierung
- Sammlungstypen und einfache Datenstrukturen
- Umsetzung einfacher algorithmischer Lösungsansätze
- Lesen, Nachvollziehen und Bewerten von Programmcode
- Fehlersuche, Debugging und einfache Fehlerbehandlung
- Grundlagen des Testens kleiner Programme
- Grundlegende objektorientierte Konzepte

Prüfungsvorleistung

Testat durch erfolgreiche Bearbeitung semesterbegleitender Programmieraufgaben.

Prüfungsleistung

Computergestützte Klausurarbeit (90 Minuten). Das Prüfungskonzept umfasst Aufgaben zur Codeanalyse, Fehlersuche, Erklärung des Programmverhaltens sowie zur eigenständigen Implementierung kleiner Programme.

Literatur

- Eric Matthes: *Python Crash Course*. 3rd edition, No Starch Press, 2023.
- Allen B. Downey: *Think Python*. 3rd edition, O'Reilly Media, 2024.
- Al Sweigart: *Automate the Boring Stuff with Python*. 2nd edition, No Starch Press, 2019.
- Johannes Ernesti; Peter Kaiser: *Python 3: Das umfassende Handbuch*. Rheinwerk Verlag, 2023.

Kurzporträt BIN004: Einführung in die Programmierung

Worum geht es? Das Modul führt in das Programmieren ein. Die Studierenden lernen, einfache Anforderungen in Programme zu überführen, Programme auszuführen, Fehler zu finden und Lösungen nachvollziehbar zu verbessern. Dabei geht es sowohl um grundlegende Sprachkonzepte als auch um praktische Entwicklungswerkzeuge.
Wofür braucht man es? Programmierkenntnisse werden in nahezu allen weiteren Informatikmodulen benötigt. Sie sind Grundlage für Softwareentwicklung, Datenanalyse, Systemprogrammierung, Webanwendungen, Machine Learning und viele Projektaufgaben. Wer Programme lesen und selbst schreiben kann, kann technische Ideen im Studium praktisch umsetzen.
Wieso ist es interessant? Programmieren macht abstrakte Ideen ausführbar. Schon mit einfachen Konzepten lassen sich kleine Werkzeuge, Simulationen, Analysen oder interaktive Anwendungen bauen. Das Modul zeigt, wie aus einer Problemstellung schrittweise ein funktionierendes Programm entsteht und wie man Fehler systematisch als Teil des Lernprozesses nutzt.

BIN005: Systemsoftware

Modulverantwortung	Prof. Dr. Jens Brandt	**Lehrperson(en)**	Prof. Dr. Jens Brandt
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	System Software
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, das Zusammenwirken von systemnahen Programmen, Toolchain und Betriebssystem-Schnittstellen zu erklären, zu untersuchen und in einfachen Programmen zu nutzen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Programme in einer Unix-/Linux-ähnlichen Umgebung analysieren,
- einfache systemnahe Programme in C entwickeln und debuggen,
- den Weg von Quelltext bis zum laufenden Prozess nachvollziehen,
- Betriebssystemabstraktionen praktisch verwenden,
- Werkzeuge zur Systembeobachtung einsetzen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um die Ausführung und das Verhalten von Programmen auf Systemebene nachvollziehen und einfache systemnahe Software entwickeln und beurteilen zu können. Dies bildet eine Grundlage für spätere Aufgaben in Betriebssystemen, hardwarenaher Entwicklung, IT-Sicherheit und technischer Fehlersuche.

Inhalte

- Userland, Kernel und Betriebssystem-Schnittstellen
- Shell, Prozesse, Dateien und Dateideskriptoren
- C als Sprache für systemnahe Programmierung
- Compiler, Linker und Loader
- Speicher, Adressräume und Debugging
- Systemaufrufe und Fehlerbehandlung
- Prozessmodell und Scheduling
- Interrupts und Virtualisierung im Überblick
- Systemnahes Programmierprojekt

Prüfungsvorleistung

Regelmäßige Bearbeitung von Übungsaufgaben.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Hausarbeit in Form eines systemnahen Programmierprojekts mit technischer Dokumentation und Fachgespräch ausgestaltet. Die Nutzung von KI-Werkzeugen ist zulässig, sofern sie in der Dokumentation kenntlich gemacht wird und die Studierenden die abgegebene Lösung fachlich erklären und vertreten können.

Literatur

- Brian W. Kernighan; Dennis M. Ritchie: *The C Programming Language*. 2nd edition, Prentice Hall, 1988.
- Michael Kerrisk: *The Linux Programming Interface*. No Starch Press, 2010.
- Randal E. Bryant; David R. O'Hallaron: *Computer Systems: A Programmer's Perspective*. 3rd edition, Pearson, 2016.
- W. Richard Stevens; Stephen A. Rago: *Advanced Programming in the UNIX Environment*. 3rd edition, Addison-Wesley, 2013.
- Andrew S. Tanenbaum; Herbert Bos: *Modern Operating Systems*. 4th edition, Pearson, 2015.

Kurzporträt BIN005: Systemsoftware

Worum geht es? Das Modul zeigt, was zwischen Quelltext, ausführbarem Programm und Betriebssystem passiert. Die Studierenden arbeiten mit systemnahen Programmen, Werkzeugketten, Prozessen, Dateien und Schnittstellen eines Unix-/Linux-ähnlichen Systems. Dadurch wird sichtbar, wie Programme auf Systemebene ausgeführt und beobachtet werden.
Wofür braucht man es? Systemsoftware-Kenntnisse werden überall dort gebraucht, wo Programme zuverlässig, effizient oder sicher mit Betriebssystemressourcen umgehen müssen. Sie sind wichtig für Betriebssysteme, IT-Sicherheit, Embedded Systems, DevOps und die Analyse technischer Fehler. Das Modul schafft dafür eine praktische Grundlage.
Wieso ist es interessant? Viele Eigenschaften von Software werden erst verständlich, wenn man unter die Oberfläche schaut. Warum ein Programm startet, wie Prozesse entstehen, wie Dateien und Speicher funktionieren oder wie Werkzeuge Fehler sichtbar machen, wird hier praktisch erfahrbar. Das macht Software weniger zur Blackbox.

BIN006: Systemhardware

Modulverantwortung	Prof. Dr. Edwin Naroska	**Lehrperson(en)**	Prof. Dr. Edwin Naroska Prof. Dr. Benedikt Janßen
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	System Hardware
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, die Ausführung eines einfachen Programms von der digitalen Datenrepräsentation und der Instruktionssatzebene bis zu ausgewählten mikroarchitektonischen, speicherbezogenen und hardwaregestützten Schutzmechanismen zu analysieren und fachlich zu erklären.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- festbreite Ganzzahl- und Gleitkommadarstellungen sowie boolesche und sequentielle Logik untersuchen,
- einfache Instruktionsfolgen einschließlich Register- und Speicherzuständen, Kontrollfluss, Stack-Nutzung und Funktionsaufrufen analysieren,
- Pipeline-Ausführung, Daten- und Kontrollabhängigkeiten sowie vereinfachte Out-of-Order-Ausführung modellieren,
- Speicherzugriffsmuster, Cache-Verhalten und grundlegende Cache-Kohärenz in Mehrkernsystemen beurteilen,
- virtuelle Adressierung, Speicherrechte und mikroarchitektonische Sicherheitswirkungen an vereinfachten Fallstudien einordnen,
- Funktion und Zusammenspiel von Komponenten moderner Computersysteme erläutern.

WOZU. Die Studierenden nutzen diese Kompetenzen, um das Verhalten systemnaher Software an der Hardware- und Betriebssystemschnittstelle fachlich beurteilen zu können. Dies unterstützt spätere Module zu Systemsoftware, hardwarenaher Entwicklung, Parallelität, Sicherheit und eingebetteten Systemen.

Inhalte

- Komponenten und Organisationsformen moderner Rechnersysteme
- Binär- und Hexadezimaldarstellung
- Festbreite Ganzzahl- und Gleitkommaformate
- Boolesche und sequentielle Logik
- Register, Taktung und Von-Neumann-Rechnermodell
- RV32I-Registermodell und Instruktionen
- Load-/Store-Prinzip, Adressierung und Speicherzugriffe
- Stack, Funktionsaufrufe und Aufrufkonventionen
- Pipelining und Spekulation im Überblick
- Speicherhierarchie, Lokalität und Cache-Verhalten
- Multiprozessorsysteme und Cache-Kohärenz
- Virtueller Speicher, Speicherrechte und Betriebssystemgrenze

Prüfungsvorleistung

Unbenotetes Testat durch Abgabe und Diskussion von Übungsaufgaben.

Prüfungsleistung

Computergestützte Klausurarbeit (90 Minuten). Das Prüfungskonzept besteht aus der geführten technischen Analyse eines kleinen Programms anhand der im Modul eingeführten Modelle zu Datenrepräsentation, Aufrufkonventionen, Pipeline- und Cache-Verhalten, Out-of-Order-Ausführung sowie Speicher- und Schutzmechanismen.

Literatur

- David Money Harris; Sarah L. Harris: *Digital Design and Computer Architecture, RISC-V Edition*. Morgan Kaufmann, 2021.
- David A. Patterson; John L. Hennessy: *Computer Organization and Design RISC-V Edition: The Hardware/Software Interface*. 2nd edition, Morgan Kaufmann, 2021.
- John L. Hennessy; David A. Patterson: *Computer Architecture: A Quantitative Approach*. 6th edition, Morgan Kaufmann, 2017.

Kurzporträt BIN006: Systemhardware

Worum geht es? Das Modul erklärt, wie Computer Programme auf Hardwareebene ausführen. Dazu gehören Datenrepräsentation, Instruktionen, Register, Speicher, Pipeline-Verarbeitung, Caches und grundlegende Schutzmechanismen. Die Studierenden lernen, das Verhalten einfacher Programme aus Sicht der Rechnerarchitektur zu analysieren.
Wofür braucht man es? Hardwareverständnis ist wichtig, wenn Softwareleistung, Speicherverhalten, Systemnähe oder Sicherheit eine Rolle spielen. Die Kompetenzen unterstützen spätere Module zu Systemsoftware, eingebetteten Systemen, paralleler Programmierung und IT-Sicherheit. Sie helfen auch, technische Grenzen und Nebenwirkungen von Programmen besser einzuschätzen.
Wieso ist es interessant? Programme wirken oft abstrakt, laufen aber immer auf konkreter Hardware. Wer versteht, wie Daten im Speicher liegen, wie Instruktionen ausgeführt werden und warum Caches oder Schutzmechanismen Verhalten beeinflussen, kann Software tiefer verstehen. Das Modul öffnet diese technische Ebene schrittweise.

BIN007: Datennetze

Modulverantwortung	Prof. Dr. Thomas Meuser	**Lehrperson(en)**	Prof. Dr. Thomas Meuser Prof. Dr. Martin Grothe
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Computer Networks
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, ein grundlegendes IP-basiertes Datennetz für überschaubare Anwendungsszenarien fachgerecht zu konzipieren, in einer Laborumgebung in Betrieb zu nehmen und dessen Funktionsfähigkeit systematisch zu prüfen und nachzuweisen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Kommunikationsvorgänge mithilfe von Schichtenmodellen, Protokollhierarchien und Internetarchitekturen strukturieren und erklären,
- Anforderungen an einfache Datennetze erfassen und daraus Netzstrukturen, Topologien und Adressierungskonzepte ableiten,
- IPv4- und IPv6-Adressierungen, Subnetze und Präfixe für kleine Netze berechnen, dokumentieren und begründen,
- Funktionen von LANs, Switching, VLANs, Routing und ausgewählten IP-Diensten in Laborumgebungen konfigurieren,
- Protokollabläufe und Netzverhalten anhand von Paket- und Zustandsinformationen analysieren,
- typische Fehler in kleinen Datennetzen systematisch eingrenzen und Prüfergebnisse nachvollziehbar dokumentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um die technische Kommunikation verteilter IT-Systeme im weiteren Informatikstudium und in beruflichen Einstiegssituationen fachlich fundiert beurteilen zu können. Dies ist Grundlage für Websysteme, verteilte Systeme, IT-Sicherheit, DevOps und viele Infrastrukturaufgaben.

Inhalte

- Einsatzbereiche, Anforderungen und Grundbegriffe von Datennetzen
- Netzkomponenten, Übertragungsmedien und Topologien
- Grundlagen des Internets und der TCP/IP-Protokollfamilie
- Anwendungsschicht und ausgewählte Internetdienste
- IPv4- und IPv6-Adressierung, Subnetting und Präfixe
- Ethernet, Switching, VLANs und einfaches LAN-Design
- Routingtabellen, statisches Routing und Routingverfahren im Überblick
- Ausgewählte IP-Dienste wie ARP, ICMP, DNS, DHCP und NAT
- Grundlagen der Paket- und Protokollanalyse
- Netzwerktools, SNMP und Syslog im Überblick
- Systematische Fehlersuche in kleinen IP-basierten Datennetzen
- Praktische Umsetzung in virtuellen Laborumgebungen

Prüfungsvorleistung

Testat über die erfolgreiche Bearbeitung von Übungs- und Laboraufgaben.

Prüfungsleistung

Online-Klausurarbeit (90 Minuten). Die Prüfung wird als Moodle-Exam-Prüfung durchgeführt.

Literatur

- James F. Kurose; Keith W. Ross: *Computer Networking: A Top-Down Approach*. 9th edition, Pearson, 2025.
- Andrew S. Tanenbaum; Nick Feamster; David J. Wetherall: *Computer Networks*. 6th edition, Pearson, 2021.
- Anatol Badach; Erwin Hoffmann: *Technik der IP-Netze*. 5. Auflage, Hanser, 2022.

Kurzporträt BIN007: Datennetze

Worum geht es? Das Modul behandelt die Grundlagen der Kommunikation in Datennetzen. Die Studierenden lernen, wie Geräte über IP-basierte Netze kommunizieren, wie Adressen, Switching, Routing und Dienste zusammenspielen und wie sich Netzverhalten beobachten lässt. Praktische Laboraufgaben machen die Konzepte nachvollziehbar.
Wofür braucht man es? Datennetze bilden die Grundlage verteilter IT-Systeme, Webanwendungen, Cloud-Dienste, IT-Sicherheit und vieler betrieblicher Infrastrukturen. Wer Netze planen, konfigurieren und prüfen kann, versteht besser, warum Systeme erreichbar sind oder ausfallen. Diese Kompetenzen werden in vielen späteren Modulen und Berufsfeldern benötigt.
Wieso ist es interessant? Netzwerke machen sichtbar, dass digitale Systeme nicht isoliert arbeiten. Schon kleine Konfigurationsfehler können Kommunikation verhindern, während gute Analysewerkzeuge Protokollabläufe transparent machen. Das Modul zeigt, wie man Verbindungen nicht nur nutzt, sondern technisch versteht und überprüft.

BIN008: Grundlagen der IT-Sicherheit

Modulverantwortung	Prof. Dr. Martin Grothe	**Lehrperson(en)**	Prof. Dr. Martin Grothe Prof. Dr. Jürgen Quade
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Foundations of IT Security
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Programmiergrundlagen, Codeanalyse, einfache Tests.
- BIN005: Systemsoftware: Betriebssystemgrundlagen, systemnahe Software, Prozesse, Dateien.
- BIN007: Datennetze: Protokolle, IP-Netze, Routing, Netzwerkdienste.
- BIN006: Systemhardware: Datenrepräsentation, Speicher, Rechnerarchitektur.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, grundlegende Sicherheitsanforderungen an Rechner-, Netzwerk- und Softwaresysteme zu analysieren, zuzuordnen und durch geeignete Schutzmaßnahmen umzusetzen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Schutzziele, Bedrohungen, Risiken sowie rechtliche und ethische Rahmenbedingungen der IT-Sicherheit erläutern,
- einfache Bedrohungsmodelle erstellen und daraus Schutzmaßnahmen ableiten,
- grundlegende kryptografische Verfahren, Hashfunktionen, digitale Signaturen und PKI-Verfahren anwenden und bewerten,
- Authentifizierungs- und Zugriffskontrollmechanismen konzeptionell einordnen und praktisch umsetzen,
- grundlegende Netzwerksicherheitsmechanismen wie Firewalls, VPN und IDS/IPS konfigurieren und bewerten,
- Betriebssysteme hinsichtlich typischer Sicherheitsmechanismen und Konfigurationsfehler untersuchen,
- sicheren Softwareentwurf und gängige Implementierungsfehler einordnen,
- Maßnahmen zum Datenschutz und zum Schutz der Privatsphäre auswählen sowie bei Sicherheitsvorfällen geeignete erste Reaktionsmaßnahmen ableiten.

WOZU. Die Studierenden nutzen diese Kompetenz, um digitale Systeme gegen typische einfache Angriffe abzusichern, Sicherheitsrisiken nachvollziehbar zu bewerten und Grundlagen für Vertiefungsmodule im IT-Sicherheitsschwerpunkt zu legen. Die Kompetenzen werden außerdem in Sicherheitsanalysen, Incident Response, digitaler Forensik und Sicherheitsaudits benötigt.

Inhalte

- Schutzziele, Bedrohungen und Risiken
- Rechtliche und ethische Rahmenbedingungen der IT-Sicherheit
- Symmetrische und asymmetrische Kryptografie
- Hashfunktionen, digitale Signaturen und Public-Key-Infrastrukturen
- Authentifizierung, Autorisierung und Zugriffskontrolle
- Betriebssystemsicherheit und sichere Konfiguration
- Netzwerksicherheit, Firewalls, VPN und IDS/IPS
- Sicherheitsarchitekturen und Defense in Depth
- Least Privilege und Zero-Trust-Grundlagen
- Sicherer Softwareentwurf und gängige Schwachstellen
- Datenschutz, Privatsphäre und gesellschaftliche Einordnung
- Erste Reaktionsmaßnahmen bei Sicherheitsvorfällen

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Klausurarbeit (90 Minuten).

Literatur

- Claudia Eckert: *IT-Sicherheit: Konzepte - Verfahren - Protokolle*. 10. Auflage, De Gruyter Oldenbourg, 2018.
- William Stallings; Lawrie Brown: *Computer Security: Principles and Practice*. 4th edition, Pearson, 2018.
- Christof Paar; Jan Pelzl: *Understanding Cryptography*. Springer, 2010.

Kurzporträt BIN008: Grundlagen der IT-Sicherheit

Worum geht es? Das Modul vermittelt Grundlagen der IT-Sicherheit für Rechner, Netzwerke und Software. Die Studierenden lernen, digitale Systeme hinsichtlich ihrer Schutzziele zu analysieren, Gefährdungen zu bewerten und geeignete Schutzmaßnahmen auszuwählen. Praktische Übungen machen zentrale Sicherheitsmechanismen nachvollziehbar.
Wofür braucht man es? Die Kompetenzen werden benötigt, um Sicherheitsrisiken in digitalen Systemen zu erkennen, zu bewerten und angemessen zu behandeln. Sie bilden die Grundlage für spätere Vertiefungen wie Angewandte Kryptografie, Sichere Ausführungsumgebungen, Ethisches Hacken und Digitale Forensik. Auch in der beruflichen Praxis sind sie für Entwicklung, Betrieb und Analyse von IT-Systemen zentral.
Wieso ist es interessant? Sicherheitsvorfälle zeigen, wie eng Technik, Organisation, Recht und Verantwortung zusammenhängen. Das Modul öffnet die Blackbox zentraler Schutzmechanismen und erklärt, warum manche Systeme trotz funktionierender Technik angreifbar bleiben. Dadurch entsteht ein fundierter Einstieg in ein besonders relevantes Informatikfeld.

Lernpfad Daten und Künstliche Intelligenz

Der Lernpfad Daten und Künstliche Intelligenz befähigt die Studierenden, datengetriebene Systeme von der Datenbasis über statistische Analyse und Machine Learning bis zu multimodalen KI-Anwendungen zu konzipieren, umzusetzen und kritisch zu bewerten. Dabei entwickeln sie Kompetenzen in Datenmodellierung, Datenintegration, Analyse, Modelltraining, Evaluation, Dokumentation und verantwortungsvoller Einordnung von KI-Systemen.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Lernpfads Datenarchitekturen und Datenpipelines entwerfen, Daten statistisch analysieren, Machine-Learning-Verfahren auswählen und evaluieren sowie einfache multimodale KI-Systeme prototypisch umsetzen. Sie entwickeln dabei insbesondere Kompetenzen im qualitätsgesicherten Umgang mit Daten, in reproduzierbaren Analyse- und KI-Workflows sowie in der kritischen Bewertung datengetriebener Ergebnisse.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Lernpfad
  - relationale, analytische und ausgewählte NoSQL-Datenmodelle entwerfen und mit Abfragesprachen praktisch nutzen,
  - Datenquellen integrieren, transformieren, qualitätsgesichert bereitstellen und Datenschutz- sowie Governance-Aspekte berücksichtigen,
  - Daten mit statistischen Kennzahlen, Visualisierungen und Wahrscheinlichkeitsmodellen beschreiben und bewerten,
  - Machine-Learning-Aufgaben erkennen, Daten vorbereiten, Modelle trainieren, evaluieren und methodische Grenzen analysieren,
  - multimodale Datenquellen und vortrainierte Modelle in prototypischen KI-Pipelines kombinieren,
  - im Capstone-Projekt eine datengetriebene Anwendung entwickeln, dokumentieren, präsentieren und ethisch sowie rechtlich reflektieren.
- **WOZU**: Damit sie datengetriebene Anwendungen und KI-basierte Lösungen fachlich fundiert entwickeln, Ergebnisse belastbar interpretieren und Risiken verantwortungsvoll einordnen können. Der Lernpfad führt von Dateninfrastruktur und Statistik über Machine Learning zur integrierten Projektaufgabe und bereitet auf berufliche Aufgaben in Data Engineering, Data Science, KI-Anwendungsentwicklung und datenbasierter Softwareentwicklung vor.

Kurzporträt Lernpfad Daten und Künstliche Intelligenz

Worum geht es? Der Lernpfad zeigt, wie aus Daten belastbare Informationen und daraus datengetriebene Software- und KI-Funktionen entstehen. Die Studierenden arbeiten mit Datenbanken, Pipelines, Statistik, Machine Learning und multimodalen KI-Systemen. Im Capstone-Projekt werden diese Bausteine zu einer zusammenhängenden Anwendung verbunden.
Wofür braucht man es? Daten und KI prägen viele moderne Softwaresysteme, Produkte und Entscheidungsprozesse. Die Kompetenzen helfen, Daten zuverlässig bereitzustellen, Modelle sinnvoll auszuwählen, Ergebnisse kritisch zu bewerten und Anwendungen verantwortungsvoll zu gestalten. Sie sind relevant für Analyse, Entwicklung, Betrieb und Beratung datengetriebener Systeme.
Wieso ist es interessant? Datengetriebene Systeme wirken oft intelligent, sind aber nur so gut wie ihre Daten, Modelle und Bewertungsmethoden. Der Lernpfad macht nachvollziehbar, wie technische Entscheidungen, Datenqualität, Statistik, Modellwahl und ethische Fragen zusammenhängen. Dadurch entsteht ein praktischer und kritischer Zugang zu KI.

BIN101: Data Engineering

Modulverantwortung	Prof. Dr. Klaus Weidenhaupt	**Lehrperson(en)**	Prof. Dr. Klaus Weidenhaupt
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Data Engineering
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, Datenarchitekturen für operative und analytische Zwecke zu entwerfen, diese mittels SQL und modernen Engineering-Tools zu implementieren sowie automatisierte, qualitätsgesicherte und sichere Datenpipelines zu entwickeln und zu betreiben.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- SQL-Abfragen von einfachen Selektionen bis zu komplexen Analysen mit Window Functions und Common Table Expressions sicher formulieren,
- Datenbankschemata mittels Datendefinitionssprache erstellen und geeignete Datentypen sowie Integritätsbedingungen zur Datenkonsistenz wählen,
- Anwendungsanforderungen in Entity-Relationship-Modelle überführen und durch Normalisierung in relationale Strukturen transformieren,
- analytische Datenmodelle für Data-Warehouse-Szenarien entwerfen und spaltenorientierte Speicherung technisch begründen,
- automatisierte ETL- und ELT-Pipelines für heterogene Datenquellen implementieren,
- DataOps-Prinzipien durch Versionierung, Orchestrierung und automatisierte Qualitätskontrollen anwenden,
- Datensicherheitskonzepte und regulatorische Anforderungen durch Zugriffskontrollen, Maskierung und Anonymisierung berücksichtigen,
- NoSQL-Datenbanken und verteilte Data-Processing-Systeme anhand technischer Kriterien einordnen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um als Data Engineers die Brücke zwischen Softwareentwicklung, IT-Betrieb und Data Science zu schlagen. Sie stellen sicher, dass Daten in hoher Qualität, skalierbar, sicher und unter Berücksichtigung moderner Entwicklungsprozesse für geschäftskritische Anwendungen und Analysen zur Verfügung stehen.

Inhalte

- Einführung in das Data Engineering: Rollenbilder und Data Engineering Lifecycle
- Relationaler Einstieg und SQL
- Fortgeschrittenes SQL mit Window Functions und CTEs
- Datenmodellierung, Integrität und Normalisierung
- DBMS-Performance, Transaktionsmanagement und Indizierung
- Analytical Engineering mit OLTP, OLAP und Dimensional Modeling
- Datenintegration und ETL-/ELT-Pipelines
- DataOps, Versionierung, Orchestrierung und Monitoring
- Datenqualität, Governance und Data Lineage
- Datensicherheit, Datenschutz und Zugriffskontrolle
- CAP-Theorem und verteilte Datenverarbeitung
- NoSQL und Big-Data-Speicherformen

Prüfungsvorleistung

Erfolgreiche Bearbeitung semesterbegleitender Aufgaben zur praktischen Anwendung von Data-Engineering-Konzepten und Methoden.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept umfasst den Entwurf von Datenmodellen, das Schreiben von SQL- und NoSQL-Abfragen, die Implementierung von Data Pipelines sowie den Transfer von Konzepten zu DataOps, Sicherheit und Data-Engineering-Architekturen.

Literatur

- Alfons Kemper, Andre Eickler: *Datenbanksysteme: Eine Einführung*. 10. Auflage, De Gruyter Oldenbourg, 2015.
- Alan Beaulieu: *Learning SQL*. 3rd Edition, O'Reilly Media, 2020.
- Joe Reis, Matt Housley: *Fundamentals of Data Engineering*. 1st Edition, O'Reilly Media, 2022.
- Martin Kleppmann: *Designing Data-Intensive Applications*. 2nd Edition, O'Reilly Media, 2026.
- Ralph Kimball, Margy Ross: *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. 3rd Edition, Wiley, 2013.
- Michael Kaufmann, Andreas Meier: *SQL und NoSQL Datenbanken*. 9. Auflage, Springer Vieweg, 2023.

Kurzporträt BIN101: Data Engineering

Worum geht es? Das Modul behandelt den Aufbau verlässlicher Dateninfrastrukturen. Die Studierenden modellieren Daten, arbeiten mit relationalen und NoSQL-Datenbanken und entwickeln Pipelines, die Daten aus unterschiedlichen Quellen aufnehmen, transformieren und bereitstellen. Dabei spielen Qualität, Sicherheit, Governance und automatisierte Betriebsprozesse eine zentrale Rolle.
Wofür braucht man es? Data Engineering ist die Grundlage für datengetriebene Anwendungen, Reporting, Data Science und KI-Systeme. Wer Daten zuverlässig speichern, integrieren und aufbereiten kann, schafft belastbare Voraussetzungen für Analysen und Entscheidungen. Die Kompetenzen werden in Softwareprojekten, Plattformteams, Analytics-Umgebungen und betrieblichen Datenarchitekturen benötigt.
Wieso ist es interessant? Daten wirken oft erst dann wertvoll, wenn sie in der richtigen Form, Qualität und Geschwindigkeit verfügbar sind. Das Modul zeigt, wie aus verstreuten Rohdaten belastbare Datenprodukte entstehen. Besonders spannend ist die Verbindung aus Modellierung, Programmierung, Automatisierung und praktischer Verantwortung für sichere Datenflüsse.

BIN102: Datenanalyse und Statistik

Modulverantwortung	Prof. Dr. Christoph Dalitz	**Lehrperson(en)**	Prof. Dr. Christoph Dalitz
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Data Analysis and Statistics
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Diese Veranstaltung behandelt die mathematische Beschreibung zufälliger Vorgänge. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, zufällige Prozesse numerisch und grafisch zusammenzufassen, mit einem Modell formal zu beschreiben, auf Basis des Modells Kenngrößen zu berechnen und Eigenschaften des Modells aus Messungen zu folgern.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- empirische Kennzahlen für die Datenauswertung auswählen und berechnen,
- Visualisierungen erstellen, die die Verteilung von Daten beschreiben und Zusammenhänge in Datensätzen darstellen,
- Prozesse mit wahrscheinlichkeitstheoretischen Modellen beschreiben und Wahrscheinlichkeiten sowie theoretische Kennzahlen berechnen,
- Punktschätzer aus Messdaten ableiten und Konfidenzintervalle bestimmen,
- die statistische Signifikanz von Beobachtungen beurteilen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um auf Basis von Datenanalysen Systeme der Informationstechnik zu entwerfen. Ferner nutzen sie sie, um Systeme im Produkteinsatz auszuwerten, deren Betrieb zu begleiten und Verbesserungen fachlich zu begründen.

Inhalte

- Quantitative und grafische Datenbeschreibung
- Wahrscheinlichkeitsrechnung
- Regressions- und Korrelationsanalyse
- Statistisches Schätzen und Schließen

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Klausurarbeit (120 Minuten).

Literatur

- Ludwig Fahrmeir, Christian Heumann, Rita Künstler, Iris Pigeot, Gerhard Tutz: *Statistik: Der Weg zur Datenanalyse*. 9. Auflage, Springer Spektrum, 2023.
- David Dalpiaz: *Applied Statistics with R*. OpenIntro, 2021.

Kurzporträt BIN102: Datenanalyse und Statistik

Worum geht es? Das Modul führt in die Beschreibung und Auswertung zufälliger Vorgänge ein. Die Studierenden lernen, Daten mit Kennzahlen und Grafiken zusammenzufassen, Zufallsprozesse mit Wahrscheinlichkeitsmodellen zu beschreiben und statistische Aussagen aus Messungen abzuleiten. Damit verbindet das Modul mathematische Grundlagen mit praktischer Datenauswertung.
Wofür braucht man es? Statistik wird benötigt, um Datenanalysen einzuordnen, Unsicherheit zu quantifizieren und beobachtete Effekte belastbar zu bewerten. Diese Kompetenzen unterstützen spätere Module zu Datenanalyse, Machine Learning, Evaluation und datengetriebener Softwareentwicklung. Auch im Betrieb von IT-Systemen helfen sie, Messdaten und Kennzahlen sinnvoll zu interpretieren.
Wieso ist es interessant? Daten erzählen selten von selbst eine klare Geschichte. Das Modul zeigt, wie man Muster, Streuung und Zufall voneinander unterscheidet und wie aus Messwerten begründete Aussagen entstehen. Interessant ist besonders, dass dieselben Methoden in sehr unterschiedlichen Anwendungen von Experimenten bis Systemmonitoring eingesetzt werden können.

BIN103: Machine Learning

Modulverantwortung	Prof. Dr. Michael Gref	**Lehrperson(en)**	Prof. Dr. Michael Gref
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Machine Learning
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN101: Data Engineering: SQL, Datenmodellierung, Datenpipelines.
- BIN102: Datenanalyse & Statistik: Wahrscheinlichkeitsrechnung, Datenanalyse, Datenvisualisierung.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, für gegebene einfache bis mittelschwere datenbasierte Problemstellungen geeignete Machine-Learning-Lösungen als reproduzierbare Prototypen zu entwickeln und deren Leistungsfähigkeit, Grenzen und Einsatzvoraussetzungen fachlich begründet zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- typische Machine-Learning-Aufgaben wie Klassifikation, Regression, Clustering und Dimensionsreduktion erkennen und geeigneten Lösungsansätzen zuordnen,
- Daten für Machine-Learning-Verfahren untersuchen, bereinigen, transformieren und methodisch sinnvoll aufteilen,
- Merkmale auswählen, erzeugen, kodieren und skalieren sowie Auswirkungen auf unterschiedliche Modellklassen begründen,
- ausgewählte Verfahren des überwachten und unüberwachten Lernens anwenden, parametrisieren und vergleichen,
- Modelle mithilfe geeigneter Gütemaße, Validierungsverfahren und Baselines evaluieren,
- methodische Probleme wie Overfitting, Data Leakage, unausgewogene Datensätze und mangelnde Generalisierbarkeit erkennen,
- einfache neuronale Netze entwerfen, trainieren und evaluieren,
- Machine-Learning-Workflows mit geeigneten Entwicklungsumgebungen und Bibliotheken praktisch umsetzen und dokumentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um datengetriebene Funktionen in Software- und Anwendungssystemen fundiert umzusetzen, ihre Ergebnisse kritisch zu bewerten und den sinnvollen Einsatz von Machine Learning in berufspraktischen Informatik-Kontexten fachlich begründen zu können.

Inhalte

- Einordnung von Machine Learning in datengetriebene KI-Anwendungen
- Typische Aufgabenstellungen wie Klassifikation, Regression, Clustering und Dimensionsreduktion
- Machine-Learning-Pipeline von Problemformulierung bis Dokumentation
- Datenaufbereitung und Feature Engineering
- Überwachtes Lernen mit ausgewählten linearen und nichtlinearen Verfahren
- Ensemble-Methoden und weitere Modellfamilien
- Unüberwachtes Lernen und Dimensionsanalyse
- Modelltraining, Modellauswahl und Modellvergleich
- Evaluation, Interpretierbarkeit und Fehleranalyse
- Grundlagen und Training einfacher neuronaler Netze
- Praktisches Training, Testen und Evaluieren einfacher Modelle
- Einordnung ausgewählter Deep-Learning-Architekturen und vortrainierter Modelle

Prüfungsvorleistung

Testat durch erfolgreiche Bearbeitung semesterbegleitender Aufgaben zur praktischen Anwendung von Machine-Learning-Verfahren.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept kann theoretische, analytische und praktische Aufgaben umfassen, bei denen Studierende Machine-Learning-Problemstellungen einordnen, Verfahren auswählen, Modelle anwenden oder implementieren, Ergebnisse interpretieren und methodische Entscheidungen begründen.

Literatur

- Aurelien Geron: *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. 3rd Edition, O'Reilly Media, 2022.
- Christopher M. Bishop: *Pattern Recognition and Machine Learning*. Springer, 2006.
- Kevin P. Murphy: *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
- Ian Goodfellow, Yoshua Bengio, Aaron Courville: *Deep Learning*. MIT Press, 2016.
- Sebastian Raschka, Vahid Mirjalili: *Python Machine Learning*. 3rd Edition, Packt Publishing, 2019.

Kurzporträt BIN103: Machine Learning

Worum geht es? Das Modul vermittelt Grundlagen des Machine Learning und deren praktische Umsetzung: Daten aufbereiten, Modelle auswählen, trainieren, evaluieren und Entscheidungen dokumentieren. Behandelt werden klassische Verfahren, einfache neuronale Netze und vortrainierte Modelle.
Wofür braucht man es? Die Kompetenzen werden für datengetriebene Software benötigt, die Muster erkennt, Vorhersagen trifft oder Entscheidungen unterstützt. Sie bilden eine Grundlage für KI-, Datenanalyse- und Projektaufgaben.
Wieso ist es interessant? Das Modul zeigt, wie entscheidend Datenqualität, Modellwahl und Evaluation für verlässliche Ergebnisse sind, und eröffnet so einen praktischen Zugang zu einem zentralen Feld der Informatik.

BIN104: Multimodale KI-Systeme

Modulverantwortung	Prof. Dr. Duc Duy Pham	**Lehrperson(en)**	Alle Lehrenden des Studiengangs
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Multimodal AI Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN103: Machine Learning: Modelltraining, Evaluation, Python-Workflows.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, multimodale KI-Systeme für anwendungsnahe Problemstellungen zu analysieren, geeignete Datenrepräsentationen und Modellbausteine auszuwählen, Verarbeitungspipelines prototypisch umzusetzen und deren Ergebnisse, Grenzen und Einsatzvoraussetzungen fachlich begründet zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- grundlegende Eigenschaften unterschiedlicher Datenmodalitäten und Repräsentationen einordnen,
- vortrainierte Modelle und geeignete Bibliotheken für ausgewählte Aufgaben nutzen,
- Verarbeitungspipelines mit unterschiedlichen Datenmodalitäten praktisch umsetzen,
- grundlegende Prinzipien zur Fusion von Modalitäten in Pipelines und Modellarchitekturen nachvollziehen,
- Ergebnisse mit passenden Baselines und Gütemaßen prüfen,
- typische Fehlerquellen, Abhängigkeiten und Grenzen multimodaler Systeme analysieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um multimodale KI-Funktionen in Software- und Anwendungssystemen fachlich fundiert einzuordnen, prototypisch umzusetzen und kritisch zu bewerten, beispielsweise für Anwendungen mit Bild-, Text-, Audio-, Sensor- oder anderen Datenquellen.

Inhalte

- Einordnung multimodaler KI-Systeme als Erweiterung bekannter Machine-Learning-Workflows
- Datenmodalitäten und Repräsentationen wie Bild, Text, Audio, Sensorik und strukturierte Daten
- Verarbeitungsschritte monomodaler und multimodaler KI-Systeme
- Auswahl und Nutzung vortrainierter Modelle und Modellbibliotheken
- Kombination und Fusion von Datenmodalitäten
- Aufbau und prototypische Umsetzung einfacher multimodaler Verarbeitungspipelines
- Evaluation und Fehleranalyse multimodaler KI-Systeme

Prüfungsvorleistung

Testat durch erfolgreiche Bearbeitung semesterbegleitender Aufgaben zur praktischen Anwendung von multimodalen KI-Systemen.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept kann theoretische, analytische und praktische Aufgaben umfassen, bei denen Studierende im Kontext multimodaler KI-Systeme Problemstellungen einordnen, Verfahren auswählen, Modelle anwenden oder implementieren, Ergebnisse interpretieren und methodische Entscheidungen begründen.

Literatur

- Ian Goodfellow, Yoshua Bengio, Aaron Courville: *Deep Learning*. MIT Press, 2016.
- Simon J. D. Prince: *Understanding Deep Learning*. MIT Press, 2023.
- Aston Zhang, Zachary C. Lipton, Mu Li, Alexander J. Smola: *Dive into Deep Learning*. Cambridge University Press, 2023.
- Christopher M. Bishop, Hugh Bishop: *Deep Learning: Foundations and Concepts*. Springer, 2024.

Kurzporträt BIN104: Multimodale KI-Systeme

Worum geht es? Das Modul erweitert Machine Learning auf Anwendungen, die mehrere Datenarten gleichzeitig nutzen. Die Studierenden arbeiten mit Bildern, Texten, Audio-, Sensor- oder strukturierten Daten und setzen einfache Verarbeitungspipelines mit passenden Modellen und Bibliotheken um. Ein Schwerpunkt liegt auf der Kombination von Modalitäten und der Bewertung der Ergebnisse.
Wofür braucht man es? Viele moderne KI-Anwendungen verarbeiten nicht nur eine einzelne Datenquelle, sondern kombinieren verschiedene Signale. Diese Kompetenzen werden für Assistenzsysteme, Dokumentanalyse, Mensch-Computer-Interaktion, industrielle Sensorik und KI-gestützte Softwarefunktionen benötigt. Das Modul unterstützt Studierende dabei, solche Systeme fachlich einzuordnen und praktisch zu prototypisieren.
Wieso ist es interessant? Multimodale Systeme wirken besonders anschaulich, weil sie Informationen aus unterschiedlichen Quellen zusammenführen. Spannend ist, dass gute Ergebnisse nicht nur vom Modell abhängen, sondern auch von Repräsentationen, Vorverarbeitung, Fusion und Evaluation. Das Modul zeigt, wie solche Systeme aus bekannten Bausteinen entstehen und wo ihre Grenzen liegen.

BIN105: Capstone Daten und Künstliche Intelligenz

Modulverantwortung	Prof. Dr. Christoph Quix	**Lehrperson(en)**	Prof. Dr. Christoph Quix Prof. Dr. Christoph Dalitz
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | 2 P | 2 S	**Engl. Titel**	Capstone Data and Artificial Intelligence
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN101: Data Engineering: Datenintegration, Datenaufbereitung, Datenqualität.
- BIN102: Datenanalyse & Statistik: statistische Analyse, Bewertung von Daten.
- BIN103: Machine Learning: Modellauswahl, Anwendung, Evaluation.

Zugehörige Lehrveranstaltungsteile

- *BIN105a: Capstone-Projekt Daten und KI*: 3 ECTS, 2 P
- *BIN105b: Ethische und gesellschaftliche Aspekte der Informatik*: 2 ECTS, 2 S

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine offene datengetriebene Problemstellung durch die eigenständige Entwicklung und Evaluation eines funktionsfähigen Software-Prototyps nachvollziehbar zu bearbeiten und dabei die im Themenbereich Daten und KI erworbenen Kompetenzen zielgerichtet zu integrieren. Ferner können sie ethische Bewertungen durchführen und rechtliche Rahmenbedingungen darlegen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine praxisnahe Problemstellung analysieren und fachliche sowie technische Anforderungen ableiten (*BIN105a*),
- Datenquellen identifizieren, integrieren, aufbereiten und hinsichtlich ihrer Eignung bewerten (*BIN105a*),
- geeignete Verfahren der Datenanalyse und des Maschinellen Lernens auswählen, anwenden und begründen (*BIN105a*),
- einen funktionsfähigen Software-Prototypen entwickeln, testen und evaluieren (*BIN105a*),
- technische Entscheidungen, Annahmen und Arbeitsschritte nachvollziehbar dokumentieren (*BIN105a*),
- den Einsatz generativer KI-Werkzeuge transparent dokumentieren und deren Ergebnisse kritisch überprüfen (*BIN105a*),
- Auswirkungen der entwickelten Lösung hinsichtlich Datenschutz, Fairness, Transparenz oder anderer ethischer Fragestellungen reflektieren (*BIN105a*, *BIN105b*),
- ethische, rechtliche oder gesellschaftliche Aspekte des IT-Einsatzes recherchieren, präsentieren und schriftlich darstellen (*BIN105b*),
- ihre Ergebnisse präsentieren und im Fachgespräch begründen (*BIN105a*, *BIN105b*).

WOZU. Die Studierenden nutzen diese Kompetenzen, um datengetriebene Anwendungen und KI-basierte Lösungen in beruflichen Anwendungsfeldern strukturiert zu konzipieren, umzusetzen, kritisch zu bewerten und verantwortungsvoll einzusetzen. Das Modul bereitet darauf vor, Datenquellen, Analyseverfahren, maschinelle Lernverfahren und Softwarekomponenten zu einer nachvollziehbaren und belastbaren Gesamtlösung zu integrieren.

Inhalte

Veranstaltungsteil *BIN105a: Capstone-Projekt Daten und KI*

- Bearbeitung einer offenen datengetriebenen Problemstellung in Kleingruppen
- Anforderungsanalyse und Strukturierung von Daten- und KI-Projekten
- Datenintegration, Datenaufbereitung und Datenqualität
- Auswahl, Anwendung und Bewertung von Verfahren der Datenanalyse und des Maschinellen Lernens
- Entwicklung eines funktionsfähigen Software-Prototyps
- Dokumentation technischer Entscheidungen und Projektartefakte
- Nachvollziehbare Nutzung generativer KI-Werkzeuge im Entwicklungsprozess
- Präsentation, Demonstration und fachliche Diskussion der Ergebnisse

Veranstaltungsteil *BIN105b: Ethische und gesellschaftliche Aspekte der Informatik*

- Reflexion ethischer, rechtlicher und gesellschaftlicher Fragestellungen im Zusammenhang der IT-Anwendung
- Ethische Theorien und ihre Anwendung auf Informatikbeispiele
- Rechtliche Normen und deren Anwendungsbereiche im IT-Kontext
- Recherche, Vortrag und schriftliche Darstellung ethischer, rechtlicher oder gesellschaftlicher Aspekte

Prüfungsvorleistung

Bestehen der unbenoteten Studienleistung im Veranstaltungsteil *BIN105b: Ethische und gesellschaftliche Aspekte der Informatik* durch Vortrag, Bericht und Teilnahme am Seminar.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Portfolio mit Präsentation und Fachgespräch im Veranstaltungsteil *BIN105a: Capstone-Projekt Daten und KI* ausgestaltet. Das Portfolio besteht aus einem funktionsfähigen Software-Prototypen für eine datengetriebene, KI-basierte Anwendung und der zugehörigen Projektdokumentation. Die Projektdokumentation dient der nachvollziehbaren Darstellung der Projektarbeit und der individuellen Beiträge. Bei Teamleistungen müssen die individuellen Beiträge der Studierenden erkennbar und bewertbar sein.

Literatur

- Foster Provost, Tom Fawcett: *Data Science for Business*. O'Reilly Media, 2013.
- Catherine Nelson: *Software Engineering for Data Scientists: From Notebooks to Scalable Systems*. O'Reilly Media, 2024.
- Douglas Birsch: *Ethical Insights: A Brief Introduction*. Mayfield Publishing, 2002.
- Michael J. Quinn: *Ethics for the Information Age*. 6th Edition, Pearson, 2015.

Kurzporträt BIN105: Capstone Daten und Künstliche Intelligenz

Worum geht es? Das Modul bündelt die Kompetenzen des Lernpfads Daten und Künstliche Intelligenz in einem abschließenden Projekt. Studierende bearbeiten in Kleingruppen eine offene datengetriebene Problemstellung und entwickeln einen funktionsfähigen Software-Prototypen. Ergänzend reflektieren sie ethische, rechtliche und gesellschaftliche Aspekte des IT- und KI-Einsatzes.
Wofür braucht man es? In der beruflichen Praxis entstehen datengetriebene Lösungen selten durch die isolierte Anwendung einzelner Verfahren. Das Modul trainiert die Verbindung von Datenintegration, Analyse, Maschinellem Lernen, Softwareentwicklung, Dokumentation und verantwortungsvoller Bewertung. Dadurch bereitet es auf Praxisprojekte, Abschlussarbeiten und berufliche Entwicklungsaufgaben vor.
Wieso ist es interessant? Das Capstone-Modul macht sichtbar, wie aus den zuvor erlernten Methoden eine zusammenhängende Anwendung entsteht. Studierende erleben, welche technischen, organisatorischen und gesellschaftlichen Entscheidungen ein KI-Projekt prägen. Besonders reizvoll ist der offene Projektcharakter, in dem eigene Lösungswege entwickelt, begründet und präsentiert werden.

Lernpfad: Software-Entwicklung

Der Lernpfad Software-Entwicklung befähigt die Studierenden, Software systematisch zu entwerfen, umzusetzen, zu testen, bereitzustellen und in bestehenden Projektkontexten qualitätsorientiert weiterzuentwickeln. Dabei entwickeln sie Kompetenzen in fortgeschrittener Programmierung, DevOps, Web-Engineering, nutzerzentrierter Gestaltung, Software Engineering, Teamarbeit und rechtlich reflektierter Entwicklungsarbeit.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Lernpfads nichttriviale Softwarelösungen methodisch entwickeln, Entwicklungs- und Bereitstellungsprozesse gestalten, Webanwendungen nutzerorientiert umsetzen und größere Softwareprojekte anhand geeigneter Prozesse, Modelle und Qualitätsmaßnahmen bearbeiten. Sie entwickeln dabei insbesondere Kompetenzen in Entwurf, Implementierung, Test, Dokumentation, Zusammenarbeit, Qualitätssicherung und kritischer Bewertung von Softwareartefakten.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Lernpfad
  - fortgeschrittene Programmierkonzepte, Typsysteme, Datenstrukturen, Entwurfsmuster und Tests in größeren Anwendungen einsetzen,
  - Versionsmanagement, Code-Reviews, automatisierte Builds, Tests, CI/CD und sicherheitsbezogene Prüfungen nutzen,
  - Nutzungskontexte analysieren und Webanwendungen mit Frontend, Backend, Schnittstellen und Datenhaltung entwickeln,
  - Anforderungen erheben, Softwaresysteme modellieren, Architekturentscheidungen begründen und Qualitätsmaßnahmen planen,
  - KI-generierte und fremde Softwareartefakte prüfen, verbessern und nachvollziehbar integrieren,
  - im Capstone-Modul einen Beitrag zu einem bestehenden größeren Softwareprojekt leisten und rechtliche Rahmenbedingungen reflektieren.
- **WOZU**: Damit sie professionelle Softwareentwicklungsaufgaben planbar, teamfähig, prüfbar und verantwortungsvoll bearbeiten können. Der Lernpfad führt von der vertieften Implementierung über Entwicklungsprozesse und Websysteme zu größeren Softwareprojekten und bereitet auf Praxisprojekte, Abschlussarbeiten und berufliche Tätigkeiten in der Softwareentwicklung vor.

Kurzporträt Lernpfad: Software-Entwicklung

Worum geht es? Der Lernpfad behandelt Softwareentwicklung als zusammenhängenden Prozess von der Idee bis zum wartbaren System. Die Studierenden vertiefen Programmierung, gestalten Entwicklungs- und Bereitstellungsprozesse, entwickeln Webanwendungen und bearbeiten größere Softwareprojekte. Dabei werden Technik, Qualität, Nutzerperspektive und Teamarbeit verbunden.
Wofür braucht man es? Software entsteht in der Praxis selten als isolierte Einzelaufgabe. Die Kompetenzen werden gebraucht, um Anforderungen zu verstehen, tragfähige Entwürfe zu erstellen, Code qualitätsgesichert umzusetzen, Änderungen nachvollziehbar bereitzustellen und rechtliche sowie organisatorische Rahmenbedingungen zu berücksichtigen. Das ist zentral für fast alle beruflichen Entwicklungsrollen.
Wieso ist es interessant? Der Lernpfad zeigt, wie aus Programmierkenntnissen professionelle Softwareentwicklung wird. Besonders spannend ist das Zusammenspiel aus Architektur, Werkzeugen, Automatisierung, Tests, Nutzererfahrung und bestehenden Projekten. Studierende erleben, warum gute Software nicht nur funktioniert, sondern auch verständlich, prüfbar und weiterentwickelbar sein muss.

BIN201: Weiterführende Programmentwicklung

Modulverantwortung	Prof. Dr. Michael Gref	**Lehrperson(en)**	Prof. Dr. Michael Gref
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Advanced Program Development
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: strukturierte Programmierung, Funktionen, elementare Datenstrukturen, Klassen, Versionsverwaltung.
- BIN006: Systemhardware: Speicherorganisation, Zeiger.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, aus gegebenen, auch unvollständigen Problemstellungen nichttriviale Anwendungen in einer kompilierten und streng typisierten Programmiersprache qualitätsorientiert zu entwerfen, zu realisieren, zu testen und begründet weiterzuentwickeln.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- funktionale und technische Anforderungen in umsetzbare Teilschritte und geeignete Klassen-, Objekt- und Schnittstellenstrukturen mit UML überführen,
- objektorientierte Konzepte wie Kapselung, Konstruktion und Zerstörung von Objekten, Klassenbeziehungen, Vererbung und Polymorphie problemangemessen anwenden,
- Datenstrukturen, generische Konzepte sowie Bibliothekskomponenten wie Container und Algorithmen auswählen und effizient einsetzen,
- Wert-, Referenz- und Zeigersemantik, Objektlebensdauer sowie Ressourcenverwaltung sicher handhaben und Ein- und Ausgaben, Fehler- und Ausnahmesituationen, Datenvalidierung und defensive Programmierung angemessen gestalten,
- Entwurfsmuster für wiederkehrende Entwurfsprobleme auswählen und begründen sowie eigene, gegebene oder automatisiert erzeugte Codeartefakte analysieren, testen, korrigieren und erweitern,
- Unit-Tests entwickeln, Qualitätsindikatoren auswerten, automatisierte Build- und Testabläufe nutzen und wesentliche Annahmen dokumentieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um in nachfolgenden Entwicklungsmodulen und in Softwareprojekten nachvollziehbare, qualitätsgesicherte und langfristig weiterentwickelbare Softwarebeiträge leisten zu können. Das Modul bildet damit eine Grundlage für DevOps, Web-Engineering, Software Engineering und größere teamorientierte Entwicklungsaufgaben.

Inhalte

- Anforderungen, Annahmen, Abgrenzungen und technische Entwurfsentscheidungen aus Problembeschreibungen ableiten
- Kompilierte, streng typisierte Programmiersprachen: Übersetzungsprozess, Typsystem, Namensräume sowie Wert- und Referenzsemantik
- Objektorientierte Programmierung mit Klassen und Objekten, Kapselung, Konstruktion und Zerstörung, Klassenbeziehungen, Delegation, Vererbung, Polymorphie und UML-Klassendiagrammen
- Ressourcen- und Speicherverwaltung, Objektlebensdauer, Ownership-Konzepte und sprachliche Abstraktionen
- Datei- und Konsolenein- und -ausgabe sowie Fehler-, Ausnahme- und defensive Behandlung von Eingaben und typischen Fehlerquellen
- Generische Programmierung und Bibliothekskomponenten, insbesondere Container, Iteratoren und Algorithmen
- Ausgewählte Entwurfsmuster aus den Kategorien Erzeugung, Struktur und Verhalten
- Testen, Testfallgestaltung, Testabdeckung und Qualitätsindikatoren; Lesen, Bewerten, Refactoring und Erweiterung von Code sowie automatisierte Build- und Testabläufe

Prüfungsvorleistung

Erfolgreiche Bearbeitung semesterbegleitender Übungs- und Praktikumsaufgaben. Die Prüfungsvorleistung kann Programmierabgaben, Tests, kurze Dokumentationen, Code-Walkthroughs oder die Erläuterung eigener beziehungsweise gegebener Lösungen beinhalten.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept umfasst Aufgaben zur Analyse, Fehlersuche, Bewertung, Erweiterung und Implementierung von Programmen.

Literatur

- Stroustrup, B.: *A Tour of C++*. 3. Auflage, Addison-Wesley, 2022.
- Lippman, S. B.; Lajoie, J.; Moo, B. E.: *C++ Primer*. 5. Auflage, Addison-Wesley, 2012.
- Meyers, S.: *Effective Modern C++*. O'Reilly, 2014.
- Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J.: *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1994.
- Fowler, M.: *Refactoring: Improving the Design of Existing Code*. 2. Auflage, Addison-Wesley, 2018.
- C++ Core Guidelines: *C++ Core Guidelines*. https://isocpp.github.io/CppCoreGuidelines/CppCoreGuidelines.

Kurzporträt BIN201: Weiterführende Programmentwicklung

Worum geht es? Das Modul vertieft die Programmierung über einfache Einstiegsaufgaben hinaus. Die Studierenden entwerfen und implementieren nichttriviale Anwendungen in einer kompilierten, streng typisierten Sprache und arbeiten dabei mit Klassen, Objekten, Speicher- und Ressourcenverwaltung, Bibliotheken, Entwurfsmustern und Tests. Ein besonderer Schwerpunkt liegt darauf, Anforderungen nachvollziehbar in robuste Software zu überführen.
Wofür braucht man es? Die Kompetenzen werden in nahezu allen späteren Entwicklungsmodulen benötigt, besonders bei größeren Softwareprojekten, Web- und Backend-Systemen, DevOps-Prozessen und Software Engineering. Wer Software nicht nur schreiben, sondern auch strukturieren, testen, erweitern und erklären kann, kann belastbare Beiträge in Teamprojekten leisten. Das Modul stärkt damit die Grundlage für professionelle Softwareentwicklung.
Wieso ist es interessant? Viele Programmieraufgaben werden spannend, sobald Lösungen größer werden und über einzelne Funktionen hinausgehen. Dann entscheiden Entwurf, Typisierung, Speicherverhalten, Tests und Wartbarkeit darüber, ob Software verständlich und verlässlich bleibt. Das Modul zeigt, wie aus Programmierkenntnissen systematische Entwicklungskompetenz wird.

Lernpfad

Software-Entwicklung

BIN202: DevOps

Modulverantwortung	Prof. Dr. Benedikt Janßen	**Lehrperson(en)**	Prof. Dr. Benedikt Janßen
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	DevOps
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Programmierung, einfache Softwareprojekte.
- BIN201: Weiterführende Programmentwicklung: Softwaretests, Versionsmanagement, Entwicklungswerkzeuge.
- BIN005: Systemsoftware: Betriebssysteme, Kommandozeile, Ausführungsumgebungen.
- BIN007: Datennetze: Netzwerke, internetbasierte Anwendungen.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, einen reproduzierbaren, sicheren und teamfähigen Entwicklungs-, Prüf- und Bereitstellungsprozess für Software zu gestalten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Versionsmanagement für kollaborative Softwareentwicklung einsetzen und typische Arbeitsabläufe wie Branching, Merging und Reviews durchführen,
- Code-Reviews vorbereiten, durchführen und Review-Ergebnisse zur Qualitätsverbesserung nutzen,
- Entwicklungs-, Test- und Ausführungsumgebungen mit Virtualisierungs- oder Containerkonzepten reproduzierbar konfigurieren,
- automatisierte Build-, Test-, Qualitätssicherungs- und Deployment-Abläufe konzipieren, umsetzen und auswerten,
- automatisierte Tests auf Code-, Komponenten- und Systemebene einrichten, ausführen und interpretieren,
- Sicherheitsprüfungen wie grundlegendes SAST, SCA, DAST, Secret Scanning und Fuzzing in CI/CD-Prozesse integrieren,
- Ergebnisse sicherheitsbezogener Prüfungen interpretieren, priorisieren und daraus begründete Maßnahmen ableiten,
- Bestandteile von Software erfassen, Supply-Chain-Risiken einordnen und KI-gestützte Werkzeuge kritisch in den Arbeitsprozess einbinden.

WOZU. Die Studierenden nutzen diese Kompetenzen, um Softwareentwicklungsprozesse in Teams so zu gestalten, dass Änderungen nachvollziehbar, automatisiert prüfbar, sicher integrierbar und zuverlässig bereitstellbar sind. Dies unterstützt spätere Projektmodule, Software Engineering, den Betrieb moderner Anwendungen und berufliche Entwicklungsprozesse.

Inhalte

- Grundbegriffe und Ziele von DevOps und DevSecOps
- Grundlegende DevOps-Metriken wie Deployment Frequency, Lead Time, Change Failure Rate und Recovery Time
- Versionsmanagement, Branching-Strategien, Merge-Prozesse und Code-Reviews
- Automatisierung von Build-, Test-, Analyse- und Deployment-Prozessen
- Continuous Integration und Continuous Delivery/Deployment als Konzepte
- Reproduzierbare Entwicklungs- und Testumgebungen, Virtualisierung und Containerkonzepte
- Automatisierte Tests auf unterschiedlichen Ebenen
- Statische Analyse, Dependency-Checks und sicherheitsbezogene Qualitätssicherung
- Software-Supply-Chain, Abhängigkeiten, Signaturen und Integrität von Artefakten, SBOM-Grundlagen und Provenance
- Zusammenarbeit im Team, Arbeitsorganisation, Nachvollziehbarkeit und kritischer Einsatz KI-gestützter Werkzeuge

Prüfungsvorleistung

Erfolgreiche Bearbeitung und Diskussion von Übungsaufgaben zu Versionsmanagement, Automatisierung, Tests, Qualitätssicherung und reproduzierbaren Entwicklungsumgebungen.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept umfasst praktische Aufgaben zu Analyse, Konfiguration, Bewertung und Erweiterung von DevOps- und DevSecOps-Abläufen.

Literatur

- Kim, G.; Humble, J.; Debois, P.; Willis, J.: *The DevOps Handbook*. 2. Auflage, IT Revolution Press, 2021.
- Humble, J.; Farley, D.: *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Addison-Wesley, 2010.
- Forsgren, N.; Humble, J.; Kim, G.: *Accelerate: The Science of Lean Software and DevOps*. IT Revolution Press, 2018.
- Beyer, B.; Jones, C.; Petoff, J.; Murphy, N. R.: *Site Reliability Engineering*. O'Reilly, 2016.
- OWASP: *Application Security Verification Standard*. https://owasp.org/www-project-application-security-verification-standard/.

Kurzporträt BIN202: DevOps

Worum geht es? Das Modul behandelt die Arbeitsweisen, Werkzeuge und Prozesse, mit denen Software im Team zuverlässig entwickelt, geprüft und bereitgestellt wird. Im Mittelpunkt stehen Versionsmanagement, Code-Reviews, automatisierte Builds, Tests, CI/CD, reproduzierbare Umgebungen und sicherheitsbezogene Prüfungen. Die Studierenden lernen, Softwareentwicklung als nachvollziehbaren technischen Prozess zu gestalten.
Wofür braucht man es? Moderne Software entsteht selten auf einzelnen Rechnern und ohne Abstimmung. DevOps-Kompetenzen werden gebraucht, damit Teams Änderungen sicher integrieren, Fehler früh erkennen, Abhängigkeiten kontrollieren und Anwendungen reproduzierbar bereitstellen können. Das ist für Studienprojekte, Abschlussarbeiten und berufliche Softwareentwicklung gleichermaßen relevant.
Wieso ist es interessant? DevOps macht sichtbar, wie aus Quellcode ein verlässliches Softwareprodukt wird. Kleine Automatisierungen können große Wirkung haben, weil sie wiederkehrende Fehler vermeiden und Qualität kontinuierlich überprüfbar machen. Das Modul verbindet technische Werkzeuge mit Teamarbeit, Sicherheit und professioneller Entwicklungskultur.

Lernpfad

Software-Entwicklung

BIN203: UX und Web-Engineering

Modulverantwortung	Prof. Dr. Thomas Nitsche	**Lehrperson(en)**	Prof. Dr. Thomas Nitsche Prof. Dr. Nils Kopal
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	UX and Web Engineering
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Programmierung.
- BIN007: Datennetze: internetbasierte Anwendungen, Schnittstellen.
- BIN201: Weiterführende Programmentwicklung: Softwarestrukturierung, Tests, Entwicklungswerkzeuge.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, nutzerzentrierte Benutzungsschnittstellen zu entwickeln und zu bewerten sowie webbasierte Anwendungen zu konzipieren und zu implementieren.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Nutzungskontexte, Zielgruppen und Anforderungen für Systeme analysieren und daraus UX-Ziele ableiten,
- Websysteme in Frontend, Backend, API und Datenhaltung gliedern und die Aufgaben der jeweiligen Komponenten erläutern,
- statische webbasierte Benutzungsschnittstellen mit HTML und CSS implementieren,
- interaktive Benutzungsschnittstellen mit JavaScript und zeitgemäßen Frontend-Frameworks implementieren,
- einfache Full-Stack-Anwendungen mit Frontend, Backend, API und Datenbankanbindung entwerfen, umsetzen und testen,
- API-Schnittstellen und Webarchitekturen mit geeigneten Notationen oder Spezifikationen beschreiben und bewerten,
- grundlegende Datenmodelle für Anwendungen erstellen und Datenzugriffe in Anwendungen integrieren,
- Benutzungsschnittstellen anhand von Gestaltgesetzen, MCI-Kriterien, Barrierefreiheitsanforderungen, Nutzerfeedback, Standards und technischen Rahmenbedingungen beurteilen und verbessern.

WOZU. Die Studierenden nutzen diese Kompetenzen, um fachliche Anforderungen und Nutzerbedürfnisse in gebrauchstaugliche, zugängliche und technisch tragfähige Websysteme zu übersetzen. Damit verbinden sie Softwareentwicklung mit Nutzerperspektive, Schnittstellengestaltung und technischer Webarchitektur.

Inhalte

- Grundlagen nutzerzentrierter Gestaltung und User Experience
- Nutzungskontexte, Zielgruppen, Anforderungen und UX-Ziele
- Navigationsstrukturen, Nutzerschnittstellenaufbau, Prototyping und Nutzerfeedback
- Gestaltgesetze, MCI-Kriterien und Grundlagen der Barrierefreiheit
- Architektur webbasierter Systeme: Frontend, Backend, API und Datenhaltung
- Frontend-Implementierung statischer und dynamischer Webseiten
- Serverseitige Verarbeitung, API-Entwurf und einfache Datenbankanbindung
- Modellierung und Spezifikation von Schnittstellen und Webarchitekturen
- Test und Bewertung von Webanwendungen aus technischer und nutzungsbezogener Perspektive

Prüfungsvorleistung

Testat als Projektarbeit mit Konzeption, Umsetzung, Test und Dokumentation einer einfachen Full-Stack-Webanwendung einschließlich UX-Begründung und Bewertung der Benutzungsschnittstelle; optional ergänzt durch Abnahmepräsentation oder Fachgespräch.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept umfasst Aufgaben zu UX-Grundlagen, Webarchitektur, Schnittstellen, Frontend- und Backend-Konzepten sowie zur Bewertung von Benutzungsschnittstellen.

Literatur

- Richter, M.; Flückiger, M. D.: *Usability und UX kompakt: Produkte für Menschen*. 5. Auflage, Springer Vieweg, 2026.
- Yablonski, J.: *Laws of UX: 10 praktische Grundprinzipien für intuitives und menschenzentriertes Design*. 2. Auflage, dpunkt.verlag, 2024.
- Krug, S.: *Don't make me think!: Web Usability: Das intuitive Web*. 3. Auflage, mitp Verlag, 2014.
- Jacobsen, J.; Meyer, L.: *Praxisbuch Usability und UX: Was alle wissen sollten, die Websites und Apps entwickeln*. 4. Auflage, Rheinwerk Verlag, 2024.
- Wolf, J.: *HTML und CSS: Das umfassende Handbuch zum Lernen und Nachschlagen*. 5. Auflage, Rheinwerk Verlag, 2023.
- Aubele, T.; Girke, D.: *Barrierefreie Webseiten - Grundlagen, Anwendung, Praxis*. Rheinwerk Computing, 2025.

Kurzporträt BIN203: UX und Web-Engineering

Worum geht es? Das Modul verbindet User Experience mit Web-Engineering. Die Studierenden analysieren Nutzungskontexte, entwerfen Benutzungsschnittstellen und setzen einfache Webanwendungen mit Frontend, Backend, Schnittstellen und Datenhaltung um. Dabei geht es sowohl um technische Realisierung als auch um Gebrauchstauglichkeit, Barrierefreiheit und nachvollziehbare Gestaltung.
Wofür braucht man es? Webanwendungen sind ein zentraler Anwendungskontext der Informatik. Die Kompetenzen helfen dabei, fachliche Anforderungen nicht nur technisch korrekt, sondern auch verständlich, zugänglich und nutzerorientiert umzusetzen. Sie werden in Softwareprojekten, Produktentwicklung, Webentwicklung, Schnittstellenentwurf und Qualitätssicherung benötigt.
Wieso ist es interessant? Bei Websystemen sehen Nutzerinnen und Nutzer unmittelbar, ob Technik verständlich funktioniert. Schon kleine Gestaltungsentscheidungen können darüber entscheiden, ob eine Anwendung hilfreich oder frustrierend ist. Das Modul zeigt, wie technische Architektur und menschliche Nutzung zusammenkommen.

Lernpfad

Software-Entwicklung

BIN204: Software Engineering

Modulverantwortung	Prof. Dr. Thomas Nitsche	**Lehrperson(en)**	Prof. Dr. Thomas Nitsche Prof. Dr. Nils Kopal
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Software Engineering
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Programmierung.
- BIN201: Weiterführende Programmentwicklung: Softwarestrukturierung, Tests, Versionsmanagement, Modellierung.
- BIN202: DevOps: Entwicklungsprozesse, Build- und Testautomatisierung.
- BIN203: UX und Web-Engineering: Anforderungen, Websysteme, nutzerorientierte Gestaltung.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, größere Softwaresysteme methodisch anhand geeigneter Softwareentwicklungsprozesse in Phasen wie Anforderung, Entwurf, Implementierung, Test und Qualitätssicherung zu konzipieren und zu entwickeln.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Anforderungen an Softwaresysteme mit geeigneten Verfahren erheben, strukturieren, priorisieren und in umsetzbare Spezifikationen überführen,
- Vorgehensmodelle, Notationen, Werkzeuge und Technologien für konkrete Softwareentwicklungsaufgaben vergleichen und begründet auswählen,
- größere Softwarelösungen im Team mit geeigneten Entwicklungsprozessen, Modellierungsansätzen und Werkzeugen entwerfen und umsetzen,
- fremde und KI-generierte Softwareartefakte prüfen, bewerten und verbessern,
- KI-gestützte Werkzeuge für Entwurf, Implementierung, Deployment und Betrieb zielgerichtet anleiten und ihre Ergebnisse kritisch validieren,
- Testverfahren für fachliche, technische und sicherheitsbezogene Anforderungen auswählen, ausführen und auswerten,
- Coding Standards, Code-Prinzipien und Dokumentationsstandards anwenden und deren Einhaltung bewerten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um komplexere Softwareprojekte im beruflichen Kontext planbar, prüfbar und qualitätsorientiert durchführen zu können. Sie können dadurch technische, organisatorische und qualitätssichernde Aspekte größerer Entwicklungsvorhaben zusammenführen.

Inhalte

- Ziele, Begriffe und Tätigkeitsfelder des Software Engineerings
- Vorgehensmodelle und Prozessgestaltung für Softwareentwicklungsprojekte
- Anforderungsanalyse, Anforderungsdokumentation, Priorisierung und Nachverfolgbarkeit
- Modellierung von Struktur, Verhalten, Schnittstellen und Systemkontexten
- Architekturentscheidungen, Entwurfsbegründungen und Analyse bestehender Systeme
- Qualitätssicherung, Teststrategien und sicherheitsbezogene Testfälle
- Teamorientierte Planung und Umsetzung größerer Softwarelösungen
- Coding Standards, Dokumentationsstandards und technische Schulden
- Bewertung und Integration KI-generierter Softwareartefakte
- Werkzeugunterstützung für Entwurf, Implementierung, Deployment und Betrieb

Prüfungsvorleistung

Erfolgreiche Projektarbeit mit Dokumentation und Fachgespräch, in der eine größere Softwareentwicklungsaufgabe einschließlich Anforderungen, Design, Implementierung, Tests und Qualitätsbewertung nachvollziehbar bearbeitet wurde.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept umfasst Aufgaben zu Anforderungen, Modellierung, Entwurf, Entwicklungsprozessen, Qualitätssicherung und zur Bewertung von Softwareartefakten.

Literatur

- Sommerville, I.: *Software Engineering*. 10. Auflage, Pearson Studium, 2018.
- Ludewig, J.; Lichter, H.: *Software Engineering: Grundlagen, Menschen, Prozesse, Techniken*. 4. Auflage, dpunkt.verlag, 2023.
- Kecher, C.; Salvanos, A.: *UML 2.5: Das umfassende Handbuch*. 5. Auflage, Rheinwerk Verlag, 2015.
- Spillner, A.; Linz, T.: *Basiswissen Softwaretest*. 7. Auflage, dpunkt.verlag, 2024.
- Beneken, G.; Hummel, F.; Kucich, M.: *Grundkurs agiles Software-Engineering: Ein Handbuch für Studium und Praxis*. Springer Vieweg, 2023.

Kurzporträt BIN204: Software Engineering

Worum geht es? Das Modul behandelt Softwareentwicklung als systematischen Prozess für größere Systeme. Die Studierenden arbeiten mit Anforderungen, Modellen, Architekturentscheidungen, Entwicklungsprozessen, Tests, Dokumentation und Qualitätssicherung. Dabei geht es nicht nur um Code, sondern um nachvollziehbare Entscheidungen über den gesamten Entwicklungsverlauf.
Wofür braucht man es? Größere Softwareprojekte benötigen planbare Abläufe, klare Anforderungen, tragfähige Entwürfe und überprüfbare Qualität. Diese Kompetenzen werden in professionellen Entwicklungsteams, Praxisprojekten, Abschlussarbeiten und technischen Leitungsaufgaben benötigt. Das Modul unterstützt dabei, Komplexität im Team beherrschbar zu machen.
Wieso ist es interessant? Software Engineering zeigt, warum gute Software nicht zufällig entsteht. Unterschiedliche Interessen, technische Randbedingungen, Zeitdruck und Qualitätsanforderungen müssen zusammengebracht werden. Das Modul macht erfahrbar, wie Methoden, Modelle und Werkzeuge helfen, aus offenen Aufgaben belastbare Systeme zu entwickeln.

Lernpfad

Software-Entwicklung

BIN205: Capstone Software-Entwicklung

Modulverantwortung	Prof. Dr. Thomas Nitsche	**Lehrperson(en)**	Prof. Dr. Thomas Nitsche Prof. Dr. Nils Kopal
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | 2 P | 2 S	**Engl. Titel**	Capstone Software Development
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- Erfolgreicher Abschluss der vorherigen Module des Lernpfads Software-Entwicklung oder vergleichbare Kompetenzen in Programmierung, DevOps, Web-Engineering, Software Engineering, Test, Dokumentation und Teamarbeit.

Zugehörige Lehrveranstaltungsteile

- *BIN205a: Capstone-Projekt Software-Entwicklung*: 3 ECTS, 2 P
- *BIN205b: Rechtliche Aspekte der Informatik*: 2 ECTS, 2 S

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine didaktisch abgegrenzte Problemstellung der Softwareentwicklung in einem vorbereiteten Projektkontext zu bearbeiten und dabei Kompetenzen aus Programmierung, Web-Engineering, Software Engineering, DevOps, Test und Dokumentation zu einem funktionsfähigen und qualitätsgesicherten Softwareinkrement zu integrieren. Ferner können sie ausgewählte rechtliche Rahmenbedingungen der entwickelten Lösung identifizieren, deren Auswirkungen erläutern und bei der Umsetzung berücksichtigen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine vorgegebene, fachlich abgegrenzte Problemstellung analysieren und daraus funktionale und nichtfunktionale Anforderungen ableiten (*BIN205a*),
- Kompetenzen aus Programmierung, Web-Engineering, Software Engineering, DevOps, Test und Dokumentation zielgerichtet zu einer Gesamtlösung verbinden (*BIN205a*),
- sich in die für die Aufgabenstellung relevanten Teile eines vorbereiteten Softwareprojekts einarbeiten (*BIN205a*),
- einen abgegrenzten Softwarebeitrag entwerfen, implementieren und in den bereitgestellten Projektkontext integrieren (*BIN205a*),
- Versionsverwaltung sowie bereitgestellte Build-, Integrations- und Deployment-Prozesse anwenden und ihren Softwarebeitrag durch automatisierte Tests und Reviews qualitätssichern (*BIN205a*),
- die für den eigenen Softwarebeitrag relevanten urheber-, lizenz-, datenschutz- oder haftungsrechtlichen Rahmenbedingungen identifizieren (*BIN205b*),
- verwendete Softwarekomponenten, Bibliotheken, Daten und digitale Inhalte hinsichtlich ihrer rechtlich zulässigen Nutzung prüfen und daraus Konsequenzen für Implementierung, Dokumentation, Lizenzierung und Bereitstellung der Lösung ableiten (*BIN205b*),
- die entwickelte Lösung demonstrieren, technische und rechtliche Entscheidungen nachvollziehbar dokumentieren und begründen sowie Feedback aus Reviews in die Weiterentwicklung einbeziehen (*BIN205a*, *BIN205b*).

WOZU. Die Studierenden nutzen diese Kompetenzen, um Methoden und Werkzeuge aus mehreren Veranstaltungen des Lernpfads Software-Entwicklung zu einer funktionsfähigen und qualitätsgesicherten Softwarelösung zu integrieren. Sie werden dadurch auf umfangreichere Informatikprojekte vorbereitet, in denen Aufgabenstellung, Vorgehen, Projektorganisation und technische Entscheidungen in höherem Maße eigenständig ausgestaltet werden müssen.

Inhalte

Veranstaltungsteil *BIN205a: Capstone-Projekt Software-Entwicklung*

- Bearbeitung einer vorgegebenen und didaktisch abgegrenzten Problemstellung
- Ableitung und Priorisierung funktionaler und nichtfunktionaler Anforderungen
- Einarbeitung in die relevanten Teile eines vorbereiteten Softwareprojekts
- Entwurf eines abgegrenzten, vertikalen Softwareinkrements
- Implementierung und Integration des Softwarebeitrags
- Versionsverwaltung, Build-Prozess und grundlegende Continuous Integration
- Automatisierte Tests, Reviews und ausgewählte Qualitätssicherungsmaßnahmen
- Technische Dokumentation, Demonstration und Begründung der Lösung

Veranstaltungsteil *BIN205b: Rechtliche Aspekte der Informatik*

- Grundlagen des Urheber- und Softwarerechts
- Softwarelizenzen, Open-Source-Lizenzen und Lizenzkompatibilität
- Schutz und Nutzung von Software, Quellcode, Daten und digitalen Inhalten
- Ausgewählte datenschutz-, haftungs- oder vertragsrechtliche Fragestellungen der Softwareentwicklung
- Anwendung rechtlicher Grundlagen auf konkrete Fallbeispiele

Prüfungsvorleistung

Bestehen einer unbenoteten Studienleistung in der Vorlesung *BIN205b: Rechtliche Aspekte der Informatik*.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Portfolio mit Präsentation und Fachgespräch ausgestaltet. Das Portfolio umfasst

- ein funktionsfähiges und in den bereitgestellten Projektkontext integriertes Softwareinkrement,
- den zugehörigen Quellcode einschließlich nachvollziehbarer Versionshistorie,
- ausgewählte automatisierte Tests und weitere Nachweise der Qualitätssicherung sowie
- eine kompakte technische Dokumentation der Anforderungen, der wesentlichen Entscheidungen und der individuellen Beiträge.

In der Präsentation wird die entwickelte Lösung demonstriert und in den Kontext der Aufgabenstellung eingeordnet. Das Fachgespräch dient insbesondere der individuellen Prüfung des Verständnisses der gewählten Lösung, der eingesetzten Methoden und der vorgenommenen Qualitätssicherungsmaßnahmen. Bei Teamleistungen müssen die individuellen Beiträge erkennbar und bewertbar sein.

Literatur

- Chacon, S.; Straub, B.: *Pro Git*. 2. Auflage, Apress, 2014. https://git-scm.com/book.
- Fogel, K.: *Producing Open Source Software*. 2. Auflage, O'Reilly, 2017.
- Open Source Initiative: *Open Source Licenses*. https://opensource.org/licenses/.
- Spindler, G.; Schuster, F. (Hrsg.): *Recht der elektronischen Medien*. Kommentar, aktuelle Auflage, C.H. Beck.
- Hoeren, T.: *Internetrecht*. Skript, Universität Münster, aktuelle Fassung.

Kurzporträt BIN205: Capstone Software-Entwicklung

Worum geht es? Das Modul führt die Kompetenzen des Lernpfads Software-Entwicklung in einer vorbereiteten Entwicklungsaufgabe zusammen. Studierende bearbeiten eine abgegrenzte Problemstellung, integrieren ein funktionsfähiges Softwareinkrement in einen Projektkontext und berücksichtigen rechtliche Rahmenbedingungen.
Wofür braucht man es? In realen Softwareprojekten müssen Anforderungen, Entwurf, Implementierung, Tests, Build-Prozesse, Dokumentation und rechtliche Fragen zusammen gedacht werden. Das Modul bereitet auf größere Informatikprojekte, Praxisphase und Abschlussarbeit vor.
Wieso ist es interessant? Die Studierenden erleben, wie aus mehreren zuvor getrennt gelernten Methoden eine zusammenhängende Softwarelösung entsteht. Dabei wird sichtbar, dass technische Qualität, rechtliche Zulässigkeit, nachvollziehbare Entscheidungen und Feedback aus Reviews zusammenwirken.

Lernpfad

Software-Entwicklung

Lernpfad Informatisches Denken und Konzepte

Der Lernpfad Informatisches Denken und Konzepte befähigt die Studierenden, informatische Problemstellungen formal zu strukturieren, algorithmisch zu lösen, mathematisch zu begründen und hinsichtlich Berechenbarkeit, Komplexität, Parallelität und Verteilung einzuordnen. Dabei entwickeln sie Kompetenzen im abstrakten Denken, im Beweisen, in der Analyse von Algorithmen und in der wissenschaftlichen Erschließung informatischer Fachtexte.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Lernpfads formale Modelle und mathematische Argumentationen verstehen, geeignete Algorithmen und Datenstrukturen auswählen, Grenzen der Berechenbarkeit und Effizienz einordnen sowie nebenläufige, parallele und verteilte Lösungen fachlich begründet entwickeln. Sie entwickeln dabei insbesondere Kompetenzen in Abstraktion, Modellierung, Korrektheits- und Laufzeitanalyse sowie wissenschaftlicher Kommunikation.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Lernpfad
  - mathematische Strukturen, Logik, Beweisprinzipien und lineare Algebra auf informatische Fragestellungen anwenden,
  - Datenstrukturen und Algorithmen hinsichtlich Korrektheit, Laufzeit, Speicherbedarf und Einsatzbedingungen analysieren,
  - Automaten, formale Sprachen, Berechenbarkeit und Komplexitätsklassen zur Einordnung informatischer Probleme nutzen,
  - nebenläufige, parallele und verteilte Programme entwerfen, synchronisieren, implementieren und bewerten,
  - formale Argumentationen aus Fachtexten rekonstruieren und durch Beispiele, Gegenbeispiele oder Demonstratoren veranschaulichen,
  - wissenschaftliche Quellen recherchieren, bewerten, dokumentieren und adressatengerecht präsentieren.
- **WOZU**: Damit sie informatische Aufgaben nicht nur praktisch umsetzen, sondern auch formal begründen, Grenzen erkennen und technische Entscheidungen nachvollziehbar argumentieren können. Der Lernpfad schafft eine Grundlage für anspruchsvolle Software-, System-, Daten- und Forschungskontexte, in denen Korrektheit, Effizienz, Skalierbarkeit und wissenschaftliche Präzision wichtig sind.

Kurzporträt Lernpfad Informatisches Denken und Konzepte

Worum geht es? Der Lernpfad führt in die theoretischen, mathematischen und konzeptionellen Grundlagen der Informatik ein. Die Studierenden beschäftigen sich mit Logik, Beweisen, Algorithmen, Datenstrukturen, formalen Sprachen, Berechenbarkeit, Komplexität sowie paralleler und verteilter Verarbeitung. Im Capstone-Modul werden formale Argumentationen wissenschaftlich erschlossen und vermittelt.

Wofür braucht man es? Diese Kompetenzen helfen, Software und Systeme nicht nur zu bauen, sondern ihre Eigenschaften zu verstehen und zu begründen. Sie sind wichtig, wenn Laufzeit, Korrektheit, Skalierbarkeit, Entscheidbarkeit oder formale Modellierung eine Rolle spielen. Der Lernpfad unterstützt damit viele spätere technische und wissenschaftliche Aufgaben.

Wieso ist es interessant? Informatik besteht nicht nur aus Implementierung, sondern auch aus präzisem Denken über Probleme, Lösungen und Grenzen. Der Lernpfad zeigt, warum manche Aufgaben effizient lösbar sind, andere nur schwer oder gar nicht, und wie formale Argumente praktische Entscheidungen beeinflussen. Dadurch wird die konzeptionelle Tiefe des Fachs sichtbar.

BIN301: Logik, Diskrete Strukturen und Lineare Algebra

Modulverantwortung	Prof. Dr. Steffen Goebbels	**Lehrperson(en)**	Prof. Dr. Ulrich Tipp Prof. Dr. Steffen Goebbels
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Logic, Discrete Structures, and Linear Algebra
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN003: Mathematik-Grundlagen der Informatik: mathematische Grundlagen, einfache Programmierung.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, mathematische Formulierungen nachzuvollziehen, grundlegende Sachverhalte mathematisch zu formulieren und zentrale Verfahren der diskreten Mathematik und linearen Algebra auf informatische Problemstellungen anzuwenden.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Problemstellungen abstrahieren und mithilfe geeigneter prädikatenlogischer Ausdrücke modellieren,
- selbstständig bereitgestellte Übungsaufgaben bearbeiten,
- kurze Beweise nachvollziehen und verständlich aufbereiten,
- Grundstrukturen wie Körper, Vektorräume und Matrizen untersuchen,
- mathematische Verfahren anhand informatischer Beispiele anwenden,
- empfohlene Literatur zur Vertiefung nutzen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um sich exakt und wissenschaftlich ausdrücken zu können, beispielsweise bei der Analyse des Laufzeitverhaltens von Algorithmen und bei Korrektheitsuntersuchungen von Programmen. Die Kompetenzen legen eine Grundlage zum Verständnis theoretischer Grenzen der Informatik.

Inhalte

- Mengen, Relationen und Abbildungen
- Aussagen- und Prädikatenlogik
- Beweisprinzipien, vollständige Induktion und rekursive Definitionen
- Anwendung logischer Strukturen in der Informatik
- Algebraische Grundstrukturen
- Vektorräume, lineare Abbildungen und Matrizen als Strukturmodelle
- Determinanten, Eigenwerte und Eigenvektoren
- Lineare Gleichungssysteme in informatischen Anwendungen
- Gradienten und Gradientenabstieg

Prüfungsvorleistung

Regelmäßige Teilnahme an den Übungen.

Prüfungsleistung

Klausurarbeit (90 Minuten).

Literatur

- Steffen Goebbels; Jochen Rethmann: *Eine Einführung in die Mathematik an Beispielen aus der Informatik*. 2. Auflage, Springer Spektrum, Berlin, 2023, https://doi.org/10.1007/978-3-662-67675-2.
- Steffen Goebbels; Stefan Ritter: *Mathematik verstehen und anwenden: Differenzial- und Integralrechnung, Lineare Algebra*. 4. Auflage, Band 1, Springer Spektrum, Berlin, 2023, https://doi.org/10.1007/978-3-662-68367-5.
- Gerald Teschl; Susanne Teschl: *Mathematik für Informatiker*, Band 1: Diskrete Mathematik und Lineare Algebra. 4. Auflage, Springer, Berlin, 2013, https://doi.org/10.1007/978-3-642-37972-7.

Kurzporträt BIN301: Logik, Diskrete Strukturen und Lineare Algebra

Worum geht es? Das Modul vertieft die formale Sprache, mit der Informatik über Strukturen, Aussagen und Korrektheit spricht. Im Mittelpunkt stehen Logik, Mengen, Relationen, Beweisprinzipien, diskrete Strukturen und lineare Algebra. Die Studierenden üben, Problemstellungen zu abstrahieren, Aussagen präzise zu formulieren und kurze Argumentationen nachvollziehbar aufzubauen.
Wofür braucht man es? Die Kompetenzen werden benötigt, um Algorithmen, Datenstrukturen, Programme und formale Modelle fachlich sauber zu analysieren. Sie unterstützen Laufzeit- und Korrektheitsargumente, theoretische Informatik, Modellierung und mathematisch geprägte Vertiefungen. Damit schließt das Modul an die mathematischen Grundlagen an und führt stärker in formales Begründen ein.
Wieso ist es interessant? Viele informatische Probleme verändern sich, sobald man sie als Relation, logische Aussage oder strukturelles Modell beschreibt. Das Modul zeigt, wie aus Beispielen belastbare Argumente werden und warum präzise Formulierungen technische Missverständnisse vermeiden. Dadurch wird sichtbar, wie eng Informatik mit Beweisen, Abstraktion und klaren Begriffen verbunden ist.

BIN302: Algorithmen und Datenstrukturen

Modulverantwortung	Prof. Dr. Jochen Rethmann	**Lehrperson(en)**	Prof. Dr. Jochen Rethmann
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Algorithms and Data Structures
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Variablen, Kontrollstrukturen, Funktionen, einfache Datenstrukturen.
- BIN003: Mathematik-Grundlagen der Informatik: mathematische Verfahren, logisches Denken, Beweise.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, für typische algorithmische Problemstellungen geeignete, korrekte und effiziente Lösungen unter Verwendung angemessener Datenstrukturen systematisch zu entwickeln.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Problemstellungen abstrahieren und mithilfe geeigneter abstrakter Datentypen und Datenstrukturen modellieren,
- Datenstrukturen anhand ihrer Operationen sowie ihres Zeit- und Speicherbedarfs untersuchen und auswählen,
- grundlegende Entwurfsparadigmen auf algorithmische Problemstellungen anwenden,
- die Korrektheit einfacher Algorithmen insbesondere mithilfe von Invarianten begründen,
- iterative und rekursive Algorithmen asymptotisch sowie experimentell analysieren,
- Algorithmen und Datenstrukturen beschreiben, programmieren und testen,
- typische Operationen wie Suchen, Einfügen und Löschen beispielhaft für Datenstrukturen durchführen,
- Lösungsalternativen unter Berücksichtigung von Laufzeit, Speicherbedarf, Eingabeeigenschaften und Implementierungsaufwand bewerten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um softwarebasierte Problemlösungen hinsichtlich Effizienz und Skalierbarkeit fachlich fundiert einschätzen und gestalten zu können. Dies unterstützt spätere Module und Projekte, in denen Datenmengen, Ressourcenbedarf oder algorithmische Korrektheit eine zentrale Rolle spielen.

Inhalte

- Datenstrukturen: Arrays, Stacks, verkettete Listen, Bäume, Hash-Tabellen, Heaps und Warteschlangen
- Komplexität und asymptotische Aufwandsabschätzung
- Landau-Symbole und grundlegende Komplexitätsklassen
- Entwurfsmethoden: Divide and Conquer, Greedy, dynamische Programmierung, Branch and Bound und Backtracking
- Suchverfahren: binäre Suche, Interpolationssuche, Suchbäume und Hashverfahren
- Sortierverfahren: Quicksort, Mergesort, Heapsort, Radixsort sowie untere und obere Schranken
- Graphalgorithmen: Breitensuche, Tiefensuche, minimale Spannbäume, kürzeste Wege, TSP, Planarität, Färbungen und maximaler Fluss

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Klausurarbeit (120 Minuten).

Literatur

- Robert Sedgewick; Kevin Wayne: *Algorithmen*. 4. Auflage, Pearson Studium, 2014.
- Thomas Ottmann; Peter Widmayer: *Algorithmen und Datenstrukturen*. 5. Auflage, Spektrum Akademischer Verlag, 2012.
- Volker Heun: *Grundlegende Algorithmen*. Vieweg+Teubner, 2008.
- Thomas H. Cormen; Charles E. Leiserson; Ronald L. Rivest; Clifford Stein: *Algorithmen: Eine Einführung*. 4. Auflage, Oldenbourg Verlag, 2013.

Kurzporträt BIN302: Algorithmen und Datenstrukturen

Worum geht es? Das Modul behandelt grundlegende Verfahren zur Organisation und Verarbeitung von Daten. Die Studierenden lernen wichtige Datenstrukturen, Such- und Sortierverfahren, Entwurfsmethoden und Graphalgorithmen kennen. Dabei steht immer auch die Frage im Mittelpunkt, wie korrekt und effizient eine Lösung ist.
Wofür braucht man es? Algorithmen und Datenstrukturen sind eine Kernkompetenz für Softwareentwicklung, Datenverarbeitung, Simulation, Optimierung und technische Systemgestaltung. Wer geeignete Strukturen auswählt und Aufwände beurteilt, kann Programme robuster und skalierbarer entwickeln. Die Kompetenzen werden in vielen späteren Informatikmodulen vorausgesetzt.
Wieso ist es interessant? Oft entscheidet nicht nur die Idee einer Lösung, sondern ihre konkrete Struktur darüber, ob ein Programm bei großen Datenmengen noch praktikabel ist. Das Modul macht sichtbar, warum scheinbar kleine Entwurfsentscheidungen große Auswirkungen auf Laufzeit und Speicherbedarf haben. Dadurch entsteht ein technischer Blick für elegante und tragfähige Problemlösungen.

BIN303: Theoretische Informatik

Modulverantwortung	Prof. Dr. Christoph Dalitz	**Lehrperson(en)**	Prof. Dr. Christoph Dalitz Prof. Dr. Jochen Rethmann
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Theoretical Computer Science
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN301: Logik, Diskrete Strukturen und Lineare Algebra: Beweistechniken, Kombinatorik, Binärdarstellung, Logarithmen.
- BIN302: Algorithmen und Datenstrukturen: einfache Algorithmen, Sortieren, Suchen, Laufzeiten.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, grundlegende Fragen der Problemlösung mit Computern und von Sprachen formal zu beschreiben und dafür Lösungen zu entwickeln oder nachzuweisen, dass sie nicht lösbar oder vermutlich nicht effizient lösbar sind.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- die Einteilung von Problemen in Komplexitäts- und Berechenbarkeitsklassen erklären,
- für ein nichtberechenbares Problem dessen Nichtberechenbarkeit beweisen,
- Zugehörigkeiten zu einer Komplexitätsklasse beweisen,
- Laufzeitanalysen für Entscheidungsprobleme durchführen,
- einfache Probleme mit Automaten und formalen Sprachen modellieren,
- die Kategorie einer formalen Sprache erkennen und nachweisen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um IT-Systeme zu entwickeln, in denen Benutzereingaben oder maschinenlesbare Daten verarbeitet werden. Ferner vermeiden sie durch die erworbenen Kompetenzen sinnlose oder falsche Lösungen für nicht lösbare oder nicht effizient lösbare Probleme und können am wissenschaftlichen Diskurs über die Komplexität von Problemen teilnehmen.

Inhalte

- Berechenbarkeit und Turingmaschinen
- Komplexitätsklassen und die P != NP-Frage
- Grammatiken und Chomsky-Hierarchie
- Automatentheorie und ihr Bezug zu Grammatiken

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Klausurarbeit (90 Minuten).

Literatur

- Michael Sipser: *Introduction to the Theory of Computation*. 3. Auflage, Cengage Learning, 2014.
- Alexander Asteroth; Christel Baier: *Theoretische Informatik: Einführung in Berechenbarkeit, Komplexität und formale Sprachen*. Pearson Studium, 2019.
- John E. Hopcroft; Rajeev Motwani; Jeffrey D. Ullman: *Introduction to Automata Theory, Languages, and Computation*. 3. Auflage, Pearson, 2007.

Kurzporträt BIN303: Theoretische Informatik

Worum geht es? Das Modul fragt nach den formalen Grundlagen des Rechnens. Die Studierenden beschäftigen sich mit Automaten, formalen Sprachen, Grammatiken, Berechenbarkeit und Komplexität. Dabei geht es darum, Probleme präzise zu beschreiben und ihre Lösbarkeit oder ihren Aufwand fachlich zu begründen.
Wofür braucht man es? Theoretische Informatik hilft, Grenzen und Möglichkeiten algorithmischer Problemlösung zu verstehen. Die Kompetenzen sind wichtig für Compilerbau, Sprachverarbeitung, Verifikation, Sicherheitsfragen, Algorithmik und wissenschaftliche Argumentation. Sie schützen auch davor, für grundsätzlich unlösbare oder praktisch zu teure Probleme ungeeignete Lösungen zu erwarten.
Wieso ist es interessant? Das Modul zeigt, dass Informatik nicht nur aus Programmieren besteht, sondern auch aus Aussagen darüber, was überhaupt berechenbar ist. Manche einfache Fragen führen zu tiefen theoretischen Grenzen. Wer diese Grenzen versteht, kann technische Problemstellungen klarer einordnen und bessere Entscheidungen treffen.

BIN304: Verteilte und parallele Programmierung

Modulverantwortung	Prof. Dr. Thomas Nitsche	**Lehrperson(en)**	Prof. Dr. Thomas Nitsche Prof. Dr. Regina Pohle-Fröhlich
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Distributed and Parallel Programming
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Kontrollstrukturen, Funktionen, Fehleranalyse.
- BIN005: Systemsoftware: Prozesse, Threads, Speicher, Betriebssystemgrundlagen.
- BIN007: Datennetze: Protokolle, Schichtenmodelle, Socket-Grundlagen.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, Lösungen für nebenläufige, parallele und verteilte Probleme begründet zu entwickeln und hinsichtlich Synchronisation, Kommunikation, Laufzeit und Skalierbarkeit zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Problemstellungen in nebenläufige, parallel ausführbare oder verteilbare Teilaufgaben zerlegen,
- Abhängigkeiten, kritische Abschnitte und gemeinsame Ressourcen identifizieren,
- geeignete Synchronisationsverfahren auswählen und einsetzen,
- Kommunikations- und Koordinationsmechanismen für verteilte Programme verwenden,
- einfache Programme oder Mini-Projekte für Shared-Memory-, Message-Passing- und verteilte Szenarien realisieren,
- typische Effekte wie Race Conditions, Deadlocks, Latenz, Kommunikationsaufwand und Skalierbarkeit untersuchen,
- Laufzeit, Speedup, Effizienz und Ressourcenbedarf von Programmen und Algorithmen analytisch abschätzen,
- Architekturen und Programmiermodelle anhand fachlicher Anforderungen vergleichen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um Architekturentscheidungen für leistungsfähige, skalierbare oder robuste Softwaresysteme fachlich begründen zu können. Dies ist besonders relevant für moderne Mehrkernsysteme, verteilte Anwendungen, Serverarchitekturen und datenintensive Verarbeitung.

Inhalte

- Grundbegriffe von Nebenläufigkeit, Parallelität und Verteilung
- Threads und Prozesse in Abgrenzung
- Shared Memory, Message Passing, SISD, SIMD und MIMD
- Laufzeit, Speedup, Effizienz und Ressourcenbedarf
- Zerlegung von Problemstellungen und Datenaufteilung
- Abhängigkeiten, Kommunikationsbedarf und Koordinationsbedarf
- Thread-basierte Ausführung, kritische Abschnitte, Semaphore und Locks
- Race Conditions und Deadlocks
- Nachrichtenbasierte Kommunikation und typische MPI-Programmiermuster
- Client-Server-Modell, Socket-Kommunikation, Peer-to-Peer und Protokollabläufe
- Latenz, Ausfallszenarien, Skalierungsgrenzen und Kommunikationsaufwand
- Vergleich paralleler und verteilter Architekturen anhand fachlicher Anforderungen

Prüfungsvorleistung

Bestehen des Praktikums.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept sieht die Bearbeitung am Rechner unter Aufsicht vor.

Literatur

- Andrew S. Tanenbaum; Herbert Bos: *Moderne Betriebssysteme*. 5. Auflage, Pearson, 2025.
- Christoph Braun: *Betriebssysteme kompakt*. Springer Vieweg, 2025.
- Andrew S. Tanenbaum; Maarten van Steen: *Distributed Systems*. 4. Auflage, Pearson, 2023.
- Peter S. Pacheco: *An Introduction to Parallel Programming*. 2. Auflage, Morgan Kaufmann, 2021.

Kurzporträt BIN304: Verteilte und parallele Programmierung

Worum geht es? Das Modul behandelt Programme, die mehrere Dinge gleichzeitig oder verteilt auf mehrere Systeme ausführen. Die Studierenden lernen Konzepte für Nebenläufigkeit, Parallelität, Synchronisation, Nachrichtenkommunikation und verteilte Ausführung kennen. Dabei werden sowohl technische Umsetzung als auch Laufzeit, Skalierbarkeit und typische Fehler betrachtet.
Wofür braucht man es? Moderne Software läuft häufig auf Mehrkernsystemen, Serverclustern oder verteilten Plattformen. Wer parallele und verteilte Programme entwerfen kann, versteht besser, wie leistungsfähige, robuste und skalierbare Systeme entstehen. Die Kompetenzen sind wichtig für Systementwicklung, Cloud-Anwendungen, datenintensive Verarbeitung und technische Architekturentscheidungen.
Wieso ist es interessant? Gleichzeitige Ausführung eröffnet große Leistungsgewinne, bringt aber auch schwer sichtbare Fehler wie Race Conditions und Deadlocks mit sich. Das Modul zeigt, warum verteilte und parallele Systeme anders gedacht werden müssen als einfache sequentielle Programme. Dadurch wird nachvollziehbar, wie anspruchsvoll moderne Softwarearchitekturen tatsächlich sind.

BIN305: Informatisches Denken und Konzepte - Capstone

Modulverantwortung	Prof. Dr. Michael Gref	**Lehrperson(en)**	Prof. Dr. Michael Gref Prof. Dr. Christoph Dalitz
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | 3 P | 1 SL	**Engl. Titel**	Computational Thinking and Concepts - Capstone
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN301: Logik, Diskrete Strukturen und Lineare Algebra: Aussagenlogik, Prädikatenlogik, Beweise, Argumentation.
- BIN302: Algorithmen und Datenstrukturen: Algorithmen, Datenstrukturen, Komplexitätsanalyse.
- BIN303: Theoretische Informatik: formale Sprachen, Automaten, Berechenbarkeit, Entscheidbarkeit.
- BIN304: Verteilte und parallele Programmierung: Nebenläufigkeit, Parallelität, Verteilung.

Zugehörige Lehrveranstaltungsteile

- *BIN305a: Formale Argumentationen in der Informatik*: 3 ECTS, 2 P
- *BIN305b: Wissenschaftliches Arbeiten in der Informatik*: 2 ECTS, 1 P | 1 SL

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, formale informatische Argumentationen aus wissenschaftlichen Fachtexten selbstständig zu erschließen, fachlich begründet zu rekonstruieren und unter Anwendung grundlegender wissenschaftlicher Arbeitsweisen adressatengerecht zu vermitteln.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine wissenschaftsnahe informatische Fragestellung und geeignete Fachtexte auswählen, analysieren und eingrenzen (*BIN305a*),
- zentrale Begriffe, Voraussetzungen, Aussagen, Beweisideen und Argumentationsstrukturen fachlich erschließen und einordnen (*BIN305a*),
- verdichtete oder ausgelassene Argumentationsschritte rekonstruieren sowie durch Beispiele, Gegenbeispiele, formale Darstellungen oder kleine Demonstratoren prüfen und veranschaulichen (*BIN305a*),
- wissenschaftliche Quellen recherchieren, hinsichtlich ihrer Eignung bewerten und unter Beachtung wissenschaftlicher Standards verwenden (*BIN305b*, angewendet in *BIN305a*),
- ihre fachlichen Ergebnisse kompakt, quellenbasiert und adressatengerecht dokumentieren und präsentieren (*BIN305a* und *BIN305b*),
- Rückmeldungen im Peer-Review und in Fachgesprächen geben, aufnehmen und zur Überarbeitung sowie fachlichen Vertretung ihrer Ergebnisse nutzen (*BIN305a* und *BIN305b*).

WOZU. Die Studierenden nutzen diese Kompetenzen, um theoretisch geprägte informatische Fragestellungen selbstständig, methodisch fundiert und für andere nachvollziehbar bearbeiten zu können.

Inhalte

Veranstaltungsteil *BIN305a: Formale Argumentationen in der Informatik*

- Bearbeitung eines ausgewählten wissenschaftlichen Fachtextes oder Textauszugs
- Analyse von Problemstellung, Begriffen, Voraussetzungen und zentralen Aussagen
- Rekonstruktion und Kommentierung von Beweisideen, Argumentationsketten und formalen Zwischenschritten
- Entwicklung von Beispielen, Gegenbeispielen, Visualisierungen, formalen Darstellungen oder kleinen Demonstratoren
- Erstellung kompakter fachlicher Artefakte
- Präsentation, fachliche Diskussion und Überarbeitung der Ergebnisse

Mögliche Themenfelder sind insbesondere Logik und Beweise, Algorithmenanalyse, Datenstrukturen, Komplexität, formale Sprachen, Automaten und Grammatiken, Berechenbarkeit, Entscheidbarkeit, Korrektheit, Verifikation sowie nebenläufiges, paralleles und verteiltes Rechnen. Die konkreten Themen und wissenschaftlichen Texte werden passend zur Durchführung festgelegt.

Veranstaltungsteil *BIN305b: Wissenschaftliches Arbeiten in der Informatik*

- Prinzipien wissenschaftlichen Arbeitens
- Erkenntnistheorien und wissenschaftliche Methoden
- Publikationsformen sowie Lesen und Erschließen wissenschaftlicher Fachtexte
- Recherche, Auswahl und Bewertung wissenschaftlicher Quellen
- Statistische Signifikanz und Reproduzierbarkeitsfragen
- Strukturierung und kompakte Darstellung formal-informatischer Argumentationen
- Zitieren, Quellenarbeit, wissenschaftliche Redlichkeit und fachsprachliche Präzision
- Präsentation, Peer-Review, Feedback und Überarbeitung
- Transfer wissenschaftlicher Arbeitsweisen auf die Aufgabenstellungen aus *BIN305a*

Prüfungsvorleistung

Testat im Veranstaltungsteil *BIN305b: Wissenschaftliches Arbeiten in der Informatik*: aktive Teilnahme, gezeigt beispielsweise durch erfolgreiche Bearbeitung begleitender Aufgaben zur Literaturrecherche, Quellenarbeit, wissenschaftlichen Darstellung, Präsentation und zum Peer-Review.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Portfolio mit Präsentation und Fachgespräch in *BIN305a: Formale Argumentationen in der Informatik* ausgestaltet.

Mögliche Prüfungsartefakte sind die Aufarbeitung eines Fachtexts, die Rekonstruktion zentraler Argumentations- oder Beweisschritte sowie geeignete Beispiele, Visualisierungen, Materialien oder Demonstratoren. Die individuelle Leistung wird im Fachgespräch überprüft; die konkrete Ausgestaltung wird zu Beginn der Veranstaltung bekanntgegeben.

Literatur

- Michael Sipser: *Introduction to the Theory of Computation*. 3. Auflage, Cengage Learning, 2014.
- Thomas H. Cormen; Charles E. Leiserson; Ronald L. Rivest; Clifford Stein: *Algorithmen: Eine Einführung*. 4. Auflage, Oldenbourg Verlag, 2013.
- Manuel René Theisen: *Wissenschaftliches Arbeiten*. 18. Auflage, Vahlen, 2021.
- Helga Esselborn-Krumbiegel: *Richtig wissenschaftlich schreiben*. 6. Auflage, Brill Schöningh, 2022.
- Norbert Franck: *Handbuch Wissenschaftliches Arbeiten*. 3. Auflage, UTB, 2017.

Kurzporträt BIN305: Informatisches Denken und Konzepte - Capstone

Worum geht es? Das Capstone-Modul verbindet formale informatische Inhalte mit wissenschaftlichem Arbeiten. Die Studierenden erschließen ausgewählte Fachtexte, rekonstruieren Argumentationen und bereiten zentrale Aussagen fachlich nachvollziehbar auf. Dabei üben sie Recherche, Quellenarbeit, Präsentation, Peer-Review und präzise wissenschaftliche Kommunikation.
Wofür braucht man es? Die Kompetenzen helfen dabei, anspruchsvolle informatische Texte und Argumentationen selbstständig zu verstehen und für eigene Arbeiten nutzbar zu machen. Sie sind besonders wichtig für Seminararbeiten, Projektberichte, Abschlussarbeiten und fachliche Diskussionen. Zugleich stärken sie den Umgang mit formalen Begründungen aus dem gesamten Lernpfad.
Wieso ist es interessant? Das Modul macht sichtbar, wie wissenschaftliche Informatik argumentiert und wie aus verdichteten Fachtexten nachvollziehbare Erklärungen entstehen. Studierende wählen Themen mit Bezug zu theoretischen und konzeptionellen Informatikfeldern und arbeiten diese eigenständig auf. Dadurch entsteht ein Abschlussformat, das formales Denken mit verständlicher Vermittlung verbindet.

Wahlpflicht-Module

Die Wahlpflichtmodule ermöglichen den Studierenden, ihr informatisches Kompetenzprofil entsprechend persönlicher Interessen und beruflicher Zielrichtungen zu vertiefen und zu verbreitern. Sie ergänzen die Pflicht- und Lernpfadmodule um spezialisierte Themen aus eingebetteten Systemen, Robotik, Cyber-Physical Systems, IT-Sicherheit, betrieblichen Informationssystemen, fortgeschrittener Programmierung, aktuellen Anwendungsfeldern der Informatik sowie anerkannten externen oder wechselnden Lehrangeboten.

Studierende wählen genau eines von drei Wahlmodellen: den vollständigen Track Embedded Systems und Robotik (`BIN421` bis `BIN425`), den vollständigen Track IT-Sicherheit (`BIN431` bis `BIN435`) oder eine freie Zusammenstellung aus den Modulen ab `BIN45x` und dem jeweils gültigen Wahlpflichtangebot. Im freien Wahlmodell erfüllt `BIN415: Capstone: Aktuelle Themen der Informatik` die Capstone-Position; in den beiden gebundenen Tracks übernehmen diese Rolle `BIN425` beziehungsweise `BIN435`. Alle drei Capstones bestehen aus einem profilspezifischen Projektteil und dem gemeinsamen Veranstaltungsteil `BIN405b: Englische Sprachkompetenz`.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss ausgewählter Wahlpflichtmodule spezialisierte informatische Aufgabenstellungen fachlich vertieft analysieren, geeignete Methoden und Werkzeuge auswählen, prototypische oder konzeptionelle Lösungen entwickeln und deren Einsatzbedingungen kritisch bewerten. Sie entwickeln dabei je nach Wahlprofil Kompetenzen in systemnaher Entwicklung, Robotik, Sicherheit, Daten- und Prozessmodellierung, fortgeschrittener Softwareentwicklung oder aktuellen Informatikthemen.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie in den Wahlpflichtmodulen
  - hardwarenahe, eingebettete, echtzeitnahe und cyber-physische Systeme analysieren, entwickeln und bewerten,
  - robotische oder technische Systemintegrationsaufgaben planen, umsetzen, testen und dokumentieren,
  - Sicherheitsanforderungen, kryptografische Mechanismen, Ausführungsumgebungen, Sicherheitsprüfungen und forensische Verfahren anwenden,
  - Informationssysteme, Datenstrukturen, Geschäftsprozesse und betriebliche Modellierungsaufgaben untersuchen,
  - fortgeschrittene Programmiersprachen-, Framework- und Persistenzkonzepte zur Entwicklung größerer Anwendungen nutzen,
  - in Capstone-Formaten spezialisierte Kompetenzen integrieren, präsentieren und fachlich begründen.
- **WOZU**: Damit sie ihr Studium gezielt profilieren und informatische Aufgabenfelder vertieft erschließen können. Die Wahlpflichtmodule bereiten auf spezialisierte Praxis- und Abschlussarbeiten sowie auf berufliche Einstiegsfelder in Embedded Systems, Robotik, IT-Sicherheit, Softwareentwicklung, Informationsmanagement, Beratung und weiteren aktuellen Bereichen der Informatik vor.

Kurzporträt Wahlpflicht-Module

Worum geht es? Der Wahlpflichtbereich bietet Raum für fachliche Vertiefung und individuelle Schwerpunktsetzung. Die Module behandeln unter anderem eingebettete und cyber-physische Systeme, Robotik, IT-Sicherheit, Informations- und Prozessmanagement, fortgeschrittene Programmierung und aktuelle Informatikthemen. Mehrere Capstone-Formate führen spezialisierte Kompetenzen in größeren Aufgaben zusammen.
Wofür braucht man es? Die Wahlpflichtmodule helfen, ein eigenes Kompetenzprofil aufzubauen und sich auf bestimmte Berufs- oder Themenfelder vorzubereiten. Studierende können technische Vertiefungen wählen, Sicherheitskompetenzen ausbauen, anwendungsnahe Systeme integrieren oder zusätzliche Software- und Modellierungskompetenzen erwerben. Dadurch wird das Studium individueller und praxisnäher.
Wieso ist es interessant? Informatik ist breit und entwickelt sich schnell weiter. Der Wahlpflichtbereich macht diese Vielfalt sichtbar und erlaubt es, persönliche Interessen mit konkreten technischen Aufgaben zu verbinden. Besonders reizvoll ist, dass viele Module an realitätsnahen Systemen, Sicherheitsfällen, Projektaufgaben oder aktuellen Anwendungskontexten arbeiten.

BIN415: Capstone: Aktuelle Themen der Informatik

Modulverantwortung	Studiendekan:in des Studiengangs Informatik	**Lehrperson(en)**	Professor:innen des Studiengangs Informatik Lehrende des Sprachenzentrums
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch und Englisch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | 2 P | 2 S	**Engl. Titel**	Capstone: Current Topics in Computer Science
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

Keine angegeben.

Zugehörige Lehrveranstaltungsteile

- *BIN415a: Aktuelle Themen der Informatik - Capstone-Projekt*: 3 ECTS, 2 P
- *BIN405b: Englische Sprachkompetenz*: 2 ECTS, 2 S

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, sich eigenständig in ein neues informatisches Anwendungsgebiet einzuarbeiten, darin eine praxisnahe Aufgabenstellung fachlich fundiert zu bearbeiten und ihre Ergebnisse in englischer Sprache nachvollziehbar zu vertreten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine informatische Aufgabenstellung analysieren, eingrenzen und strukturieren (*BIN415a*),
- sich anhand geeigneter Materialien und Fachquellen selbstständig in das Themengebiet einarbeiten (*BIN415a*),
- geeignete Methoden auswählen, anwenden und fachlich begründen (*BIN415a*),
- fachliche Artefakte erstellen, evaluieren, dokumentieren und präsentieren (*BIN415a*),
- ihr englisches Sprachniveau bestimmen, geeignete Lernangebote nutzen und ihre Sprachkompetenz bis zum Niveau B2 weiterentwickeln (*BIN405b*).

WOZU. Die Studierenden nutzen diese Kompetenzen, um neue informatische Aufgabenfelder selbstständig zu erschließen und in international geprägten Studien- und Berufskontexten fachlich zu bearbeiten.

Inhalte

Veranstaltungsteil *BIN415a: Aktuelle Themen der Informatik - Capstone-Projekt*

- Bearbeitung einer praxisnahen informatischen Projektaufgabe
- Selbstständige Einarbeitung in fachliche Grundlagen und Methoden
- Analyse, Entwurf, Umsetzung oder Evaluation geeigneter Projektartefakte
- Dokumentation, Präsentation, fachliche Diskussion und Reflexion
- Nutzung von Feedback zur Überarbeitung
- Themen außerhalb der Modulbereiche Daten und Künstliche Intelligenz, Software-Entwicklung, Informatisches Denken und Konzepte, IT-Sicherheit sowie Robotik/Embedded Systems

Veranstaltungsteil *BIN405b: Englische Sprachkompetenz*

- Erhebung und Einordnung des individuellen Sprachniveaus
- Bestimmung des persönlichen Lernbedarfs
- Nutzung geeigneter Selbstlernangebote und Veranstaltungen des Sprachenzentrums
- Feedback und Reflexion des Lernfortschritts
- Vorbereitung auf englischsprachige Präsentation und Fachdiskussion

Prüfungsvorleistung

Testat im Veranstaltungsteil *BIN405b: Englische Sprachkompetenz*: Nachweis des englischen Sprachniveaus B2. Der Nachweis ist Voraussetzung für die Teilnahme an der Prüfungsleistung.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektportfolio mit Präsentation und/oder Fachgespräch in englischer Sprache im Veranstaltungsteil *BIN415a: Aktuelle Themen der Informatik - Capstone-Projekt* ausgestaltet. Bewertet wird ausschließlich die fachlich-informatische Leistung; mögliche Artefakte sind Konzepte, Prototypen, Analysen, Implementierungs- oder Evaluationsnachweise sowie technische Dokumentation.

Literatur

- themenabhängig

Kurzporträt BIN415: Capstone: Aktuelle Themen der Informatik

Worum geht es? Das Modul verbindet ein selbstständig bearbeitetes informatisches Capstone-Projekt außerhalb der festgelegten Profilbereiche mit englischer Fachkommunikation. Studierende erschließen ein neues oder aktuelles Themenfeld, bearbeiten eine praxisnahe Aufgabe und bereiten ihre Ergebnisse so auf, dass sie fachlich und sprachlich nachvollziehbar diskutiert werden können.
Wofür braucht man es? Die Kompetenzen werden gebraucht, wenn man sich in technische Felder einarbeitet, die nicht schon durch einen Lernpfad vorstrukturiert sind. Zugleich trainiert das Modul englische Fachkommunikation für Projekte, Abschlussarbeiten und berufliche Situationen, in denen Dokumentation, Präsentation und Diskussion international anschlussfähig sein müssen.
Wieso ist es interessant? Das Modul schafft Freiraum für Themen, die aktuell, spezialisiert oder quer zu den Profilen liegen. Studierende verbinden fachliche Eigenständigkeit mit sprachlicher Weiterentwicklung und erleben, wie man ein unbekanntes Aufgabenfeld strukturiert erschließt, bearbeitet und in englischer Fachsprache vertritt.

BIN421: Hardwarenahe Systementwicklung

Modulverantwortung	Prof. Dr. Edwin Naroska	**Lehrperson(en)**	Prof. Dr. Edwin Naroska Prof. Dr. Matteo Zella
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Hardware-Oriented Systems Development
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Programmierung, Datentypen, Software-Schnittstellen.
- BIN006: Systemhardware: Rechnerarchitektur, Speicherorganisation, digitale Grundlagen.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, hardwarenahe eingebettete Systeme aus Mikrocontroller-/SoC-nahen Plattformen, Sensoren, Aktoren und Softwarekomponenten systematisch zu analysieren, in Betrieb zu nehmen und modular so zu strukturieren, dass robuste Schnittstellen zwischen Hardware und Software entstehen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Mikrocontroller, Mikroprozessoren, SoC-nahe Plattformen, FPGA-nahe Plattformen und Beschleunigeroptionen hinsichtlich Eigenschaften, Einsatzbereichen und Grenzen einordnen,
- hardwarenahe APIs, Speicherzugriffe, Zustände und Nebenwirkungen in Software-Schnittstellen analysieren,
- Sensoren und Aktoren über GPIO, ADC, PWM, I²C, SPI und UART anbinden, testen und integrieren,
- physikalische Größen, Abtastung, Quantisierung, Messbereiche, Rauschen und Kalibrierung bei Sensordaten berücksichtigen,
- einfache digitale Filter zur Glättung und Vorverarbeitung von Sensordaten einsetzen und bewerten,
- ereignisgetriebene Abläufe mit Polling, Timern oder Interrupts hinsichtlich Reaktionszeit und Robustheit analysieren,
- Host-/Target-Entwicklungsprozesse mit Toolchains, Build-Prozessen, Deployment und Inbetriebnahme anwenden,
- Treiber- und Hardwareabstraktionsschichten entwerfen und ein praktisches Projekt mit Mikrocontroller-Board, Sensorik, Aktorik und einfacher Systemanbindung dokumentiert realisieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um softwareintensive Systeme an der Schnittstelle zur physischen Welt nachvollziehbar, wartbar und integrationsfähig entwickeln zu können. Sie bilden eine Grundlage für Aufgaben in Embedded Systems, IoT, Robotik, Cyber-Physical Systems, technischer Informatik und systemnaher Softwareentwicklung.

Inhalte

- Grundbegriffe eingebetteter und hardwarenaher Systeme
- Mikrocontroller, Mikroprozessoren, SoC-nahe Plattformen, FPGA-nahe Plattformen und Hardwarebeschleuniger
- Speicher, Zustände, Nebenwirkungen, Fehlerfälle und robuste API-Nutzung
- Peripherie und Schnittstellen: GPIO, ADC, PWM, I²C, SPI, UART
- Treiberkonzepte und Hardwareabstraktion
- Polling, Interrupts, Timer, einfache Nebenläufigkeit, Latenz und Race Conditions
- Sensoren und Aktoren: Anbindung, Messbereiche, Kalibrierung und Plausibilisierung
- Abtastung, Rauschen, Signalbezug und einfache digitale Filterung
- Host-/Target-Entwicklung, Toolchains, Deployment und Fehlersuche
- Modulare Embedded-Softwarearchitektur
- Praktisches Projekt mit Mikrocontroller-Board, Sensor-Kit, Aktorik und Systemintegration

Prüfungsvorleistung

Erfolgreiche Bearbeitung definierter Praktikums- und Projektaufgaben während des Semesters.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Hausarbeit mit Fachgespräch ausgestaltet. Die Hausarbeit dokumentiert und reflektiert Analyse, Inbetriebnahme, Strukturierung und Bewertung eines hardwarenahen Systems im praktischen Projekt.

Literatur

- Peter Marwedel: *Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems*. 4. Auflage, Springer, 2021.
- Elecia White: *Making Embedded Systems*. 2. Auflage, O'Reilly, 2024.
- David A. Patterson, John L. Hennessy: *Computer Organization and Design*. 6. Auflage, Morgan Kaufmann, 2020.
- Phillip Koopman: *Better Embedded System Software*. Drumnadrochit Education, 2010.
- Jonathan W. Valvano: *Embedded Systems: Introduction to Arm Cortex-M Microcontrollers*. 5. Auflage, Eigenverlag, 2017.

Kurzporträt BIN421: Hardwarenahe Systementwicklung

Worum geht es? Das Modul vermittelt, wie Software mit Hardware, Sensoren und Aktoren zusammenwirkt. Studierende lernen Mikrocontroller-nahe Plattformen, Schnittstellen und hardwarenahe Software nicht nur zu verwenden, sondern auch deren Grenzen, Nebenwirkungen und Integrationsanforderungen zu verstehen. Im Mittelpunkt steht ein praktisches eingebettetes System.
Wofür braucht man es? Die Kompetenzen werden benötigt, wenn Software Geräte, Maschinen, Messsysteme oder vernetzte Komponenten steuert. Sie helfen dabei, Schnittstellen zwischen Hardware und Software robust zu gestalten, Integrationsprobleme früh zu erkennen und technische Systeme nachvollziehbar zu dokumentieren.
Wieso ist es interessant? Viele Informatiksysteme enden nicht an einer Benutzeroberfläche, sondern wirken direkt auf die physische Welt. Das Modul macht erfahrbar, wie Code, elektrische Signale, Sensorwerte und Aktorik zusammen ein funktionierendes technisches System bilden.

BIN422: Echtzeitsysteme

Modulverantwortung	Prof. Dr. Jürgen Quade	**Lehrperson(en)**	Prof. Dr. Jürgen Quade Prof. Dr. Benedikt Janßen
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Real-Time Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN005: Systemsoftware: Prozesse, Threads, Speicher, I/O, Betriebssystem-Schnittstellen.
- BIN421: Hardwarenahe Systementwicklung: hardwarenahe Schnittstellen, Sensorik, Aktorik.
- Programmierung in C.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine vernetzte Echtzeitsystemlösung für einen physikalischen Prozess mit nachweisbarem Zeitverhalten und kontrollierter Fehlerreaktion zu entwickeln.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- funktionale, zeitliche und sicherheitsbezogene Anforderungen mit Zustands-, Task-, Kommunikations- und Zeitmodellen beschreiben,
- Echtzeitarchitekturen unter Berücksichtigung von Anwendung, Betriebssystemkern, Prozessen, Threads und Interrupts entwerfen,
- nebenläufige Echtzeitsoftware mit Betriebssystem-APIs, Synchronisation, Interprozesskommunikation und Zeitgebern implementieren,
- Verfahren zur Vermeidung unkontrollierter Seitenfehler durch Speichervorbelegung und -sperrung anwenden,
- Schedulingverfahren, Ressourcenkonflikte, Prioritätsinversion und deadlinebezogene Fragestellungen analysieren,
- lokale und verteilte Zeitquellen anhand von Offset, Drift, Jitter und Synchronisationsfehlern bewerten,
- Zeitverhalten und Ressourcenverbrauch durch Logging, Tracing und reproduzierbare Messungen diagnostizieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um softwareintensive Systeme zuverlässig mit zeitkritischen Vorgängen der physischen Welt zu koppeln. Sie können Echtzeitverhalten experimentell nachweisen und Fehlerreaktionen so gestalten, dass physikalische Prozesse kontrolliert bleiben.

Inhalte

- Harte und weiche Echtzeitanforderungen, Perioden, Deadlines, Latenz, Jitter und Zykluszeiten
- Zustandsautomaten, Tasks, Kommunikation, Zeitbedingungen und sichere Zustände
- Anwendungs-/Kernel-Grenze, Prozesse, Threads, Interrupt-Verarbeitung und Multicore-Zuordnung
- Synchronisation, kritische Abschnitte, Interprozesskommunikation, Deadlocks, Race Conditions und Prioritätsinversion
- Schedulingmodelle, Response-Time- und Auslastungsanalysen ohne formale Verifikation
- Schedulingklassen, CPU-Affinität, Zeitgeber, Memory Prefaulting und Speichersperrung
- Monotone und zivile Uhren, Zeitstempel, Offset, Drift und Ereignisordnung
- NTP und PTP als exemplarische Synchronisationsverfahren
- Logging, Tracing, Latenz- und Jittermessung
- Watchdogs, Timeouts, Plausibilitätsprüfungen, Fail-safe-Zustände und Fehlerinjektion
- Entwicklung und experimentelle Bewertung eines vernetzten Echtzeitsystems

Prüfungsvorleistung

Testat im Rahmen der Übung.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektarbeit in Gruppen mit individuellem Fachgespräch ausgestaltet. Gegenstand ist ein lauffähiges Softwareartefakt mit technischer Dokumentation; das Fachgespräch überprüft persönliches Systemverständnis und Entwurfsentscheidungen.

Literatur

- Jürgen Quade, Michael Mächtel: *Moderne Realzeitsysteme kompakt. Eine Einführung mit Embedded Linux*. dpunkt.verlag, 2012.
- Jane W. S. Liu: *Real-Time Systems*. Prentice Hall, 2000.
- Giorgio C. Buttazzo: *Hard Real-Time Computing Systems*. 3. Auflage, Springer, 2011.
- Hermann Kopetz: *Real-Time Systems: Design Principles for Distributed Embedded Applications*. 2. Auflage, Springer, 2011.

Kurzporträt BIN422: Echtzeitsysteme

Worum geht es? Echtzeitsysteme müssen nicht nur korrekt rechnen, sondern auch rechtzeitig reagieren. Das Modul behandelt, wie zeitkritische Software für physikalische Prozesse modelliert, implementiert und gemessen wird. Dabei spielen Nebenläufigkeit, Scheduling, Zeitquellen, Messbarkeit und kontrollierte Fehlerreaktionen eine zentrale Rolle.
Wofür braucht man es? Echtzeitkompetenzen werden in Robotik, eingebetteten Systemen, Automatisierung, Medizintechnik, Fahrzeugtechnik und industriellen Steuerungen benötigt. Wer solche Systeme entwickelt, muss begründen können, ob Reaktionen innerhalb vorgegebener Zeitgrenzen erfolgen und wie sich Last- und Fehlerfälle auswirken.
Wieso ist es interessant? Das Modul zeigt, dass Softwareverhalten in der realen Welt messbar und zeitabhängig ist. Kleine Verzögerungen, ungünstige Prioritäten oder Speicherzugriffe können große Wirkung haben. Dadurch wird sichtbar, wie eng Programmierung, Betriebssysteme und physikalische Prozesse zusammenhängen.

BIN423: Security and Safety for Embedded Systems

Modulverantwortung	Prof. Dr. Jens Brandt	**Lehrperson(en)**	Prof. Dr. Jens Brandt Prof. Dr. Benedikt Janßen
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch und Englisch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Security and Safety for Embedded Systems
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN004: Einführung in die Programmierung: Programmiergrundlagen, Softwareentwicklung, Tests.
- BIN005: Systemsoftware: Betriebssysteme, Prozesse, Systemschnittstellen.
- BIN007: Datennetze: Protokolle, IP-Netze, Netzwerkdienste.
- BIN421: Hardwarenahe Systementwicklung: eingebettete Systeme, Schnittstellen, Sensorik/Aktorik.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, für ein vernetztes eingebettetes System aus einer gegebenen Problemstellung einen normenorientierten Safety- und Security-Entwicklungsprozess zu gestalten, der Anforderungen, risikobasierte Entwurfsentscheidungen, Umsetzung und Nachweise nachvollziehbar miteinander verbindet.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Systemgrenzen, Nutzungskontexte, Betriebszustände und schützenswerte Werte bestimmen,
- Safety, Security und Zuverlässigkeit voneinander abgrenzen und in Beziehung setzen,
- normative und regulatorische Rahmenwerke hinsichtlich Lebenszyklus, Rollen, Arbeitsprodukten und Nachweisen untersuchen,
- Gefährdungen, Bedrohungen, Fehlerszenarien und Angriffsflächen analysieren und Risiken bewerten,
- Safety- und Security-Mechanismen vergleichen, auswählen und in eine Systemarchitektur integrieren,
- KI-agentenbasierte Entwicklungsumgebungen mit Kontextinformationen, Randbedingungen und Akzeptanzkriterien anleiten,
- Anforderungen, Risiken, Entwurfsentscheidungen, Implementierung und Prüfungen durch Traceability verknüpfen,
- Reviews, Tests sowie Fehler- und Angriffsinjektionen planen und auswerten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um in beruflichen Entwicklungsprojekten belastbare und verantwortbare Architektur- und Sicherheitsentscheidungen für vernetzte eingebettete Systeme mit physischer Wirkung zu treffen.

Inhalte

- Grundlagen und Abgrenzung von Safety, funktionaler Sicherheit, Security, Zuverlässigkeit und Risiko
- Normen- und regulierungsorientierte Entwicklungsprozesse
- Safety- und Security-Anforderungen, Systemkontext, Betriebs- und Fehlerzustände
- Gefährdungs- und Risikoanalyse, Bedrohungsmodellierung und Angriffsflächenanalyse
- Safety-Mechanismen: Fehlererkennung, Plausibilitätsprüfung, Watchdogs, Redundanz und Fail-safe-Verhalten
- Security-Mechanismen: Isolation, Authentizität, Integrität, sichere Programmierung und geschützte Update-Pfade
- Kryptografische und hardwaregestützte Sicherheitsmechanismen ohne mathematische oder schaltungstechnische Vertiefung
- KI-agentenbasierte Entwicklungsabläufe, Review, Validierung, Reproduzierbarkeit und menschliche Verantwortung
- Verifikation, Validierung, Simulation, Fehler- und Angriffsinjektion
- Traceability, Konfigurations- und Änderungsmanagement sowie strukturierte Sicherheitsargumentation

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Studienarbeit mit Fachgespräch ausgestaltet.

Literatur

- Nancy G. Leveson: *Engineering a Safer World*. MIT Press, 2011.
- Ross Anderson: *Security Engineering*. 3. Auflage, Wiley, 2020.
- Daniel J. Bernstein, Tanja Lange, Peter Schwabe: *Post-Quantum Cryptography*. Springer, 2009.
- ISO/IEC: *ISO/IEC 27001: Information Security Management Systems*. ISO, aktuelle Fassung.
- IEC: *IEC 61508: Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*. IEC, aktuelle Fassung.

Kurzporträt BIN423: Security and Safety for Embedded Systems

Worum geht es? Das Modul verbindet Safety und Security für eingebettete Systeme. Studierende betrachten vernetzte technische Systeme, die sowohl gegen gefährliche Fehlfunktionen als auch gegen Angriffe abgesichert werden müssen. Dabei geht es um Anforderungen, Risiken, Normen, Architekturentscheidungen, Nachweise und verantwortbare Entwicklung.
Wofür braucht man es? Die Inhalte sind relevant für Systeme mit physischer Wirkung, etwa in Robotik, IoT, Automatisierung, Medizintechnik oder Mobilität. Wer solche Systeme entwickelt, muss Sicherheitsentscheidungen nachvollziehbar begründen und prüfen können, ob technische Maßnahmen zu Risiko, Normenlage und Einsatzkontext passen.
Wieso ist es interessant? Safety und Security verfolgen unterschiedliche Blickrichtungen, treffen in realen Systemen aber aufeinander. Das Modul zeigt, wie Schutz vor Unfällen, Angriffen und Fehlfunktionen zusammengedacht wird und warum gute technische Lösungen auch belastbare Prozesse und Nachweise brauchen.

BIN424: Cyber-Physical Systems: Wahrnehmung, Interaktion und Kooperation

Modulverantwortung	Prof. Dr. Benedikt Janßen	**Lehrperson(en)**	Prof. Dr. Benedikt Janßen Prof. Dr. Matteo Zella
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 1 Ü | 1 P | - S	**Engl. Titel**	Cyber-Physical Systems: Perception, Interaction, and Cooperation
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN421: Hardwarenahe Systementwicklung: Sensorik, Aktorik, Embedded-Softwarearchitektur.
- BIN422: Echtzeitsysteme: Zustände, Tasks, Zeitverhalten, Synchronisation.
- BIN423: Security and Safety for Embedded Systems: Risiken, Safety/Security, Nachweise.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, Cyber-Physical Systems als Sense-Compute-Act-Systeme zu modellieren, zu analysieren und prototypisch umzusetzen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Cyber-Physical Systems durch Systemgrenzen, Umweltmodelle, Betriebsmodi, Zustände und Datenflüsse beschreiben,
- Sense-Compute-Act-Schleifen in funktionale und architektonische Modelle überführen,
- einfache Sensor-, Signal- und Bilddaten auf eingebetteten oder ressourcenbeschränkten Systemen verarbeiten,
- Verfahren zur Zustandsschätzung, Lokalisierung, Navigation, Kartenbezug und Planung einordnen,
- Mensch-CPS-Interaktionen modellieren und anhand einfacher Kriterien untersuchen,
- Embedded-Middleware, Robotik-Frameworks und verteilte CPS-Architekturen beurteilen,
- mobile Roboter als exemplarische Cyber-Physical Systems kinematisch, funktional und architektonisch einordnen,
- einfache Klassifikationsverfahren und Koordinationsprinzipien für Gesten, Umgebungszustände, Hindernisse oder vernetzte CPS prototypisch einsetzen und analysieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um informatische Lösungen für softwareintensive Systeme mit physischer Wirkung fachlich fundiert zu entwerfen und deren Möglichkeiten, Grenzen und Einsatzbedingungen beurteilen zu können. Die Kompetenzen unterstützen Tätigkeiten in Robotik, vernetzten eingebetteten Systemen, industriellen Assistenzsystemen, IoT-Anwendungen und autonomen Systemen.

Inhalte

- Grundbegriffe und Eigenschaften von Cyber-Physical Systems
- Sense-Compute-Act-Schleifen, Systemgrenzen, Umweltmodelle und physische Wirkung
- Betriebsmodi, Zustände, Zustandsautomaten und einfache Kontrollarchitekturen
- Funktionale und architektonische Modellierung von CPS
- Datenflüsse, Schnittstellen und Kopplung zwischen Sensorik, Verarbeitung und Aktorik
- Sensor-, Signal- und Bilddatenverarbeitung in CPS
- Merkmalsextraktion und Klassifikation für Gesten, Umgebungszustände oder Hindernisse
- Zustandsschätzung, Lokalisierung, Navigation und Planungsverfahren im Überblick
- Mensch-CPS-Interaktion und einfache Evaluationskriterien
- Mobile Roboter als exemplarische Cyber-Physical Systems
- Verteilte CPS, Embedded-Middleware und Robotik-Frameworks
- Koordination vernetzter CPS und Multi-Roboter-Systeme

Prüfungsvorleistung

Erfolgreiche Bearbeitung der Praktikumsaufgaben.

Prüfungsleistung

Computergestützte Klausurarbeit (90 Minuten). Im Rahmen des Prüfungskonzepts analysieren die Studierenden ein gegebenes Cyber-Physical System anhand einer fachlichen Problemstellung; Übungsartefakte können als Hilfsmittel vorgesehen werden.

Literatur

- Edward A. Lee, Sanjit A. Seshia: *Introduction to Embedded Systems: A Cyber-Physical Systems Approach*. 2. Auflage, MIT Press, 2017.
- Rajeev Alur: *Principles of Cyber-Physical Systems*. MIT Press, 2015.
- Bruno Siciliano, Oussama Khatib: *Springer Handbook of Robotics*. 2. Auflage, Springer, 2016.
- Roland Siegwart, Illah R. Nourbakhsh, Davide Scaramuzza: *Introduction to Autonomous Mobile Robots*. 2. Auflage, MIT Press, 2011.
- Stuart Russell, Peter Norvig: *Artificial Intelligence: A Modern Approach*. 4. Auflage, Pearson, 2020.

Kurzporträt BIN424: Cyber-Physical Systems: Wahrnehmung, Interaktion und Kooperation

Worum geht es? Cyber-Physical Systems verbinden Software, eingebettete Hardware, Sensorik, Aktorik und physische Umgebungen. Das Modul behandelt, wie solche Systeme ihre Umgebung wahrnehmen, Zustände verarbeiten, Entscheidungen treffen und physisch wirken. Mobile Robotik und vernetzte eingebettete Systeme dienen als anschauliche Beispiele.
Wofür braucht man es? Die Kompetenzen werden gebraucht, wenn Software reale Prozesse beeinflusst, etwa in Robotik, IoT, autonomen Systemen oder industriellen Assistenzsystemen. Studierende lernen, solche Systeme nicht nur zu programmieren, sondern ihre Architektur, Kopplung, Grenzen und Einsatzbedingungen informatisch zu beurteilen.
Wieso ist es interessant? Das Modul zeigt Informatik in Bewegung: Sensoren liefern unvollständige Daten, Software trifft Entscheidungen und Aktoren verändern die Umgebung. Dadurch entsteht ein unmittelbarer Zusammenhang zwischen Modell, Code, physischem Verhalten und Kooperation technischer Systeme.

BIN425: Capstone: Robotikwerkstatt

Modulverantwortung	Prof. Dr. Holger Dander	**Lehrperson(en)**	Prof. Dr. Holger Dander Prof. Dr. Matteo Zella Lehrende des Sprachenzentrums
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch und Englisch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | 2 P | 2 S	**Engl. Titel**	Capstone: Robotics Workshop
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN421: Hardwarenahe Systementwicklung: Embedded-Software, Schnittstellen, Hardwareabstraktion.
- BIN422: Echtzeitsysteme: Zustandsmodellierung, Ereignisse, Logging, Zeitverhalten.
- BIN423: Security and Safety for Embedded Systems: sicherheitsbewusste Systementwicklung, Risiken, Dokumentation.

Zugehörige Lehrveranstaltungsteile

- *BIN425a: Robotikwerkstatt - Capstone-Projekt*: 3 ECTS, 2 P
- *BIN405b: Englische Sprachkompetenz*: 2 ECTS, 2 S

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine abgegrenzte Entwicklungs- und Integrationsaufgabe in einem eingebetteten robotischen System im Team fachlich fundiert zu planen, umzusetzen, technisch zu dokumentieren und ihre Ergebnisse in englischer Sprache nachvollziehbar zu vertreten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Anforderungen und Schnittstellen eines robotischen Teilsystems aus einem gegebenen Gesamtsystem ableiten,
- eine modulare Softwarelösung für eine eingebettete, Linux-basierte Rechenplattform entwerfen und implementieren,
- Hardwarezugriffe kapseln und über definierte Kommunikationsschnittstellen mit vorhandenen Komponenten interagieren,
- Sensorinformationen, Zustände und Systemereignisse erfassen, auswerten und für Systemverhalten nutzbar machen,
- Integrationsprobleme durch Diagnose, Logging und Tests nachvollziehbar bearbeiten,
- ihre Arbeit in Projektmeetings abstimmen, technische Entscheidungen im Team begründen und Ergebnisse für fachfremde oder anwendungsnahe Zielgruppen verständlich dokumentieren und präsentieren (*BIN425a*),
- ihr englisches Sprachniveau bestimmen, geeignete Lernangebote nutzen und ihre Sprachkompetenz bis zum Niveau B2 weiterentwickeln (*BIN405b*).

WOZU. Die Studierenden nutzen diese Kompetenzen, um komplexe Entwicklungs- und Integrationsaufgaben in eingebetteten, robotischen und cyber-physischen Systemen zu bearbeiten und in international geprägten Studien- und Berufskontexten fachlich zu vertreten. Dies bereitet auf Tätigkeiten in Robotik, Embedded-Softwareentwicklung, Systemintegration, Automatisierungstechnik und technischer Produktentwicklung vor.

Inhalte

Veranstaltungsteil *BIN425a: Robotikwerkstatt - Capstone-Projekt*

- Entwicklungs- und Integrationsaufgaben in robotischen und cyber-physischen Systemen
- Analyse von Anforderungen, Schnittstellen und Systemgrenzen
- Modulare Embedded-Softwarearchitektur auf Anwendungsebene
- Linux-basierte eingebettete Rechenplattformen
- Hardwareabstraktion und Kommunikation mit Fahrplattformen, Sensoren oder Aktoren
- Zustandsverwaltung, Ereignisverarbeitung und strukturierte Systemdiagnose
- Sensornahe Informationsverarbeitung, Hinderniserkennung, QR-Code-Auswertung oder Abstandserfassung
- Lokalisierung, Koordination, Gestenerkennung oder kooperatives Transportverhalten
- Integration, Testbarkeit, Logging und Fehlersuche
- Technische Dokumentation von Schnittstellen, Architekturentscheidungen und Testergebnissen
- Adressatengerechte Präsentation technischer Ergebnisse

Veranstaltungsteil *BIN405b: Englische Sprachkompetenz*

- Erhebung und Einordnung des individuellen Sprachniveaus
- Bestimmung des persönlichen Lernbedarfs
- Nutzung geeigneter Selbstlernangebote und Veranstaltungen des Sprachenzentrums
- Feedback und Reflexion des Lernfortschritts
- Vorbereitung auf englischsprachige Präsentation und Fachdiskussion

Prüfungsvorleistung

Testat im Veranstaltungsteil *BIN405b: Englische Sprachkompetenz*: Nachweis des englischen Sprachniveaus B2. Der Nachweis ist Voraussetzung für die Teilnahme an der Prüfungsleistung.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektarbeit mit technischer Dokumentation und Präsentation und/oder Fachgespräch in englischer Sprache im Veranstaltungsteil *BIN425a: Robotikwerkstatt - Capstone-Projekt* ausgestaltet. Bewertet wird ausschließlich die fachlich-informatische Leistung; die Bewertung erfolgt individuell auf Grundlage des eigenen Beitrags, der technischen Qualität, der Dokumentation und der adressatengerechten Darstellung.

Literatur

- Roland Siegwart, Illah R. Nourbakhsh, Davide Scaramuzza: *Introduction to Autonomous Mobile Robots*. 2. Auflage, MIT Press, 2011.
- Len Bass, Paul Clements, Rick Kazman: *Software Architecture in Practice*. 4. Auflage, Addison-Wesley, 2021.
- Ian Sommerville: *Software Engineering*. 10. Auflage, Pearson, 2015.
- Christopher Hallinan: *Embedded Linux Primer*. 2. Auflage, Pearson, 2010.
- Peter Corke: *Robotics, Vision and Control*. 3. Auflage, Springer, 2023.

Kurzporträt BIN425: Capstone: Robotikwerkstatt

Worum geht es? In diesem Modul arbeiten Studierende an einem realen oder realitätsnahen robotischen System. Sie entwickeln ein abgegrenztes Teilsystem, integrieren Software, Sensorik, Kommunikation und Systemverhalten und vertreten ihre Ergebnisse in englischer Sprache. Ein möglicher Kontext ist ein kooperatives System mobiler Roboter.
Wofür braucht man es? Robotische Systeme entstehen durch das Zusammenspiel vieler Komponenten, nicht durch einzelne Algorithmen. Die Kompetenzen werden benötigt, wenn Software in bestehende technische Systeme integriert, getestet und in international geprägten Projektkontexten nachvollziehbar dokumentiert und diskutiert werden muss.
Wieso ist es interessant? Die Robotikwerkstatt macht sichtbar, wie aus Schnittstellen, Sensorwerten, Teamarbeit und Software ein integriertes System entsteht. Studierende verbinden technische Entwicklung unter realitätsnahen Bedingungen mit englischer Fachkommunikation und sichtbarem Systemverhalten.

BIN431: Angewandte Kryptografie

Modulverantwortung	Prof. Dr. Nils Kopal	**Lehrperson(en)**	Prof. Dr. Nils Kopal
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Applied Cryptography
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN008: Grundlagen der IT-Sicherheit: Sicherheitsziele, Bedrohungen, Schutzmaßnahmen.
- BIN301: Logik, Diskrete Strukturen und Lineare Algebra: diskrete Strukturen, modulare Arithmetik, Wahrscheinlichkeitsrechnung.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, die kryptografische Absicherung digitaler Systeme anforderungsgerecht zu gestalten, fachgerecht umzusetzen und ihre Wirksamkeit systematisch zu beurteilen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Sicherheitsanforderungen und Bedrohungsannahmen aus Anwendungsszenarien ableiten,
- mathematische und konzeptionelle Grundlagen symmetrischer und asymmetrischer Verfahren erläutern,
- Hashfunktionen, Message Authentication Codes, digitale Signaturen und Schlüsselaustauschverfahren einordnen,
- kryptografische Verfahren und Parameter anhand von Sicherheitsniveau, Einsatzzweck und Leistungsanforderungen auswählen,
- Demonstratoren mit etablierten kryptografischen Bibliotheken entwickeln,
- Schlüssel, Nonces, Initialisierungsvektoren, Salt und Zufallswerte sachgerecht verwenden,
- Konfigurationen und Implementierungen auf typische kryptografische Fehler untersuchen,
- Passwortspeicherung, Schlüsselmanagement, Schlüsselrotation, abgesicherte Updates, Postquantenkryptografie und kryptografische Agilität einordnen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um die Sicherheit bestehender digitaler Systeme hinsichtlich ihrer kryptografischen Absicherung fachlich fundiert zu bewerten. Das Modul schafft Grundlagen für sichere Ausführungsumgebungen, ethisches Hacken und das Capstone-Projekt IT-Sicherheit.

Inhalte

- Begriffe der Kryptologie, Kryptografie und Kryptoanalyse
- Historische Verfahren und mathematische Grundlagen
- Symmetrische und asymmetrische Verschlüsselung
- Betriebsarten, authentifizierte Verschlüsselung und Schlüsselaustausch
- Hashfunktionen, Message Authentication Codes und digitale Signaturen
- Zufall, Entropie, Schlüsselmanagement, Nonces, Initialisierungsvektoren und Salt
- Sichere Verwendung kryptografischer Bibliotheken und Schnittstellen
- Auswahl von Verfahren und Parametern für eingebettete Systeme bis Webanwendungen
- Typische Implementierungs- und Konfigurationsfehler
- Kryptografische Absicherung signierter Software- und Firmwareupdates
- Kryptografische Agilität, Verfahrensmigration und Grundlagen der Postquantenkryptografie

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept überprüft Verständnis, Auswahl, Analyse und Begründung kryptografischer Verfahren und kann Berechnungs-, Analyse-, Bewertungs- und Entwurfsaufgaben enthalten.

Literatur

- Christof Paar, Jan Pelzl: *Understanding Cryptography*. Springer, 2010.
- Jean-Philippe Aumasson: *Serious Cryptography*. 2. Auflage, No Starch Press, 2024.
- David Wong: *Real-World Cryptography*. Manning, 2021.
- Jonathan Katz, Yehuda Lindell: *Introduction to Modern Cryptography*. 3. Auflage, CRC Press, 2020.
- Niels Ferguson, Bruce Schneier, Tadayoshi Kohno: *Cryptography Engineering*. Wiley, 2010.
- Bundesamt für Sicherheit in der Informationstechnik: *Technische Richtlinien zur Kryptografie*. BSI, aktuelle Fassung.

Kurzporträt BIN431: Angewandte Kryptografie

Worum geht es? Kryptografie schützt gespeicherte und übertragene Informationen vor unberechtigtem Zugriff und Manipulation. Das Modul behandelt, wie kryptografische Verfahren funktionieren, wie sie eingesetzt werden und welche Fehler ihre Schutzwirkung beeinträchtigen. Anwendungsszenarien reichen von eingebetteten Systemen bis zu Webanwendungen.
Wofür braucht man es? Kryptografie ist Grundlage vieler Sicherheitsmechanismen in Software, Kommunikation, Updates und Identitätsprüfung. Wer digitale Systeme entwickelt oder prüft, muss Verfahren passend auswählen, Schlüssel korrekt behandeln und Fehlkonfigurationen erkennen können.
Wieso ist es interessant? Kryptografische Verfahren wirken oft unsichtbar, entscheiden aber über Vertrauen in digitale Systeme. Das Modul zeigt, warum kleine Details wie Zufall, Parameter oder Nonce-Wiederverwendung große Auswirkungen haben und wie sich Schutzwirkung fachlich beurteilen lässt.

BIN432: Sichere Ausführungsumgebungen

Modulverantwortung	Prof. Dr.-Ing. Martin Grothe	**Lehrperson(en)**	Prof. Dr.-Ing. Martin Grothe
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Secure Execution Environments
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN008: Grundlagen der IT-Sicherheit: Bedrohungsmodelle, Schutzmaßnahmen, Sicherheitsziele.
- BIN005: Systemsoftware: Betriebssysteme, Prozesse, Rechte.
- BIN007: Datennetze: Protokolle, IP-Netze, Netzwerkdienste.
- BIN431: Angewandte Kryptografie: Schlüssel, Authentisierung, Integrität.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, für ein gegebenes Bedrohungsmodell eine geeignete sichere Ausführungsumgebung zu entwerfen, umzusetzen und hinsichtlich ihrer Isolation und Widerstandsfähigkeit zu bewerten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Schutzgüter, Angreifermodelle, Systemgrenzen, Vertrauensbeziehungen und Angriffsflächen erfassen,
- Bedrohungsmodelle erstellen und Anforderungen an Isolation, Zugriffskontrolle, Kommunikation und Überwachung ableiten,
- prozessbasierte Isolation, Sandboxes, Container, virtuelle Maschinen und hardwareunterstützte Mechanismen vergleichen,
- Isolationstechnologien und Sicherheitsmechanismen für konkrete Szenarien auswählen und begründen,
- sichere Ausführungsumgebungen mit minimalen Berechtigungen, Zugriffskontrollen und Ressourcenbeschränkungen aufbauen,
- Demonstratoren zur Isolation von Anwendungen und Diensten entwickeln,
- Konfigurationen auf typische Fehler untersuchen und ausgewählte Angriffe gegen Ausführungsumgebungen kontrolliert nachvollziehen,
- Härtungsmaßnahmen durch Sicherheits-, Fehler- und Angriffstests überprüfen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um Software und vernetzte Systeme durch nachvollziehbare Vertrauensgrenzen gegen Fehlfunktionen, unberechtigte Zugriffe und Angriffe abzusichern. Das Modul schafft Grundlagen für ethisches Hacken, digitale Forensik und das Capstone-Projekt IT-Sicherheit.

Inhalte

- Schutzgüter, Sicherheitsziele, Angreifer- und Vertrauensmodelle
- Isolation, Defense in Depth, Least Privilege und Zero-Trust-Grundlagen
- Bedrohungsmodellierung, System- und Datenflussanalyse
- Isolation auf Betriebssystemebene, Berechtigungen, Capabilities und Systemaufrufbeschränkungen
- Sandbox-, Container- und VM-basierte Isolation
- Hardwareunterstützte Sicherheit, Virtualisierung, Secure Boot und TPM
- Authentisierung, Autorisierung, Dienstidentitäten, Geheimnisverwaltung und Netzwerksegmentierung
- Systemhärtung, sichere Konfiguration, Patch-/Update-Management und Überwachung
- Sichere Integration vernetzter und eingebetteter Systeme
- Schwachstellen, Fehlkonfigurationen und Angriffe auf Ausführungsumgebungen
- Sicherheits- und Negativtests, Konfigurationsanalyse und Risikodokumentation

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektarbeit mit technischer Dokumentation und individuellem Fachgespräch (20 Minuten) ausgestaltet. Die Studierenden entwerfen und realisieren für ein vorgegebenes Anwendungsszenario eine sichere Ausführungsumgebung und überprüfen deren Wirksamkeit anhand definierter Tests.

Literatur

- Ross Anderson: *Security Engineering*. 3. Auflage, Wiley, 2020.
- Matt Bishop: *Computer Security: Art and Science*. 2. Auflage, Addison-Wesley, 2018.
- William Stallings, Lawrie Brown: *Computer Security: Principles and Practice*. 4. Auflage, Pearson, 2018.
- Liz Rice: *Container Security*. O'Reilly, 2020.
- Bundesamt für Sicherheit in der Informationstechnik: *IT-Grundschutz-Kompendium*. BSI, aktuelle Fassung.
- National Institute of Standards and Technology: *Application Container Security Guide*. NIST SP 800-190, 2017.

Kurzporträt BIN432: Sichere Ausführungsumgebungen

Worum geht es? Sichere Ausführungsumgebungen begrenzen Rechte, Ressourcen und Kommunikationsmöglichkeiten von Software. Das Modul untersucht, wie Prozesse, Anwendungen und Dienste durch Sandboxes, Container, virtuelle Maschinen und weitere Schutzmechanismen voneinander isoliert werden. Studierende setzen ausgewählte Maßnahmen praktisch um und prüfen ihre Wirkung.
Wofür braucht man es? Moderne Anwendungen bestehen aus vielen Komponenten, die einander nicht uneingeschränkt vertrauen dürfen. Die Kompetenzen werden benötigt, um Sicherheitsgrenzen zu entwerfen, Fehlkonfigurationen zu erkennen und Angriffe oder Fehler auf einzelne Komponenten zu begrenzen.
Wieso ist es interessant? Isolation macht abstrakte Sicherheitsprinzipien praktisch sichtbar. Das Modul zeigt, wie Berechtigungen, Container, virtuelle Maschinen und Härtung zusammenwirken und warum eine scheinbar kleine Konfiguration über die Sicherheit eines Gesamtsystems entscheiden kann.

BIN433: Ethisches Hacken

Modulverantwortung	Prof. Dr. Nils Kopal	**Lehrperson(en)**	Prof. Dr. Nils Kopal Prof. Dr.-Ing. Martin Grothe
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Sommersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Ethical Hacking
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN008: Grundlagen der IT-Sicherheit: Bedrohungen, Schwachstellen, Schutzmaßnahmen.
- BIN005: Systemsoftware: Betriebssysteme, Prozesse, Dienste.
- BIN007: Datennetze: Protokolle, IP-Netze, Netzwerkdienste.
- BIN431: Angewandte Kryptografie: kryptografische Protokolle, Fehlkonfigurationen.
- BIN432: Sichere Ausführungsumgebungen: Isolation, Härtung, Zugriffskontrolle.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, autorisierte Sicherheitsprüfungen digitaler Systeme methodisch zu planen, kontrolliert durchzuführen, auszuwerten und daraus begründete Verbesserungsmaßnahmen abzuleiten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Ziele, Scope, Ausschlüsse und zulässige Prüfhandlungen für Sicherheitsprüfungen festlegen,
- rechtliche, ethische und organisatorische Rahmenbedingungen berücksichtigen,
- Systeme, Dienste und Vertrauensgrenzen erfassen sowie Angriffsflächen ableiten,
- Methoden der Informationsgewinnung, Enumeration und Schwachstellenanalyse anwenden,
- Netzwerkdienste, Betriebssysteme, Webanwendungen und Schnittstellen untersuchen,
- ausgewählte Schwachstellen kontrolliert verifizieren und deren Auswirkungen bewerten,
- Schwachstellenscans, Penetrationstests und Red-Team-Übungen abgrenzen,
- Befunde dokumentieren, priorisieren, adressatengerecht kommunizieren und Gegenmaßnahmen durch erneute Prüfungen überprüfen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um reale Angriffsmöglichkeiten auf bestehende digitale Systeme zu erkennen und deren technische Risiken nachvollziehbar zu bewerten. Das Modul schafft Grundlagen für digitale Forensik und das Capstone-Projekt IT-Sicherheit.

Inhalte

- Ziele, Grenzen und rechtlich-ethische Rahmen autorisierter Sicherheitsprüfungen
- Autorisierung, Scope, Rules of Engagement, Datenverhalten und Responsible Disclosure
- Phasenmodell von Informationsgewinnung bis Nachtest
- Abgrenzung von Vulnerability Assessment, Scan, Penetrationstest und Red/Purple Teaming
- Aktive und passive Informationsbeschaffung, OSINT und Enumeration
- Protokoll- und Schnittstellenanalyse, Segmentierung und Filterregeln
- Betriebssystem- und Infrastruktursicherheit
- Webanwendungen, Authentisierung, Autorisierung, Input Validation, Injection und XSS
- Eingebettete und vernetzte Systeme, IoT-Schnittstellen und Update-Mechanismen
- Kontrollierte Verifikation, Risikobewertung, Priorisierung und Abbruchkriterien
- Reporting, reproduzierbare Nachweise, Empfehlungen und Nachtest
- Automatisierung wiederkehrender Analysen und Validierung automatischer Befunde

Prüfungsvorleistung

Erfolgreiche Bearbeitung ausgewählter Laborübungen mit Kurzprotokollen; unbenotet.

Prüfungsleistung

Computergestützte Klausurarbeit (120 Minuten). Das Prüfungskonzept wird als praktische Laborprüfung in einer kontrollierten Laborumgebung mit schriftlichem Prüfungsprotokoll durchgeführt. Die Studierenden untersuchen ein vorbereitetes System, bewerten Befunde und dokumentieren Gegenmaßnahmen.

Literatur

- Georgia Weidman: *Penetration Testing*. No Starch Press, 2014.
- Patrick Engebretson: *The Basics of Hacking and Penetration Testing*. 2. Auflage, Syngress, 2013.
- David Kennedy, Jim O'Gorman, Devon Kearns, Mati Aharoni: *Metasploit*. No Starch Press, 2011.
- Andrew Hoffman: *Web Application Security*. O'Reilly, 2020.
- Bundesamt für Sicherheit in der Informationstechnik: *Leitfaden zur Durchführung von Penetrationstests*. BSI, 2003.
- National Institute of Standards and Technology: *Technical Guide to Information Security Testing and Assessment*. NIST SP 800-115, 2008.

Kurzporträt BIN433: Ethisches Hacken

Worum geht es? Ethisches Hacken untersucht digitale Systeme aus der Perspektive potenzieller Angreifer, aber mit klarer Autorisierung und Verantwortung. Studierende planen Sicherheitsprüfungen, analysieren Angriffsflächen, verifizieren Schwachstellen kontrolliert und dokumentieren Befunde nachvollziehbar.
Wofür braucht man es? Die Kompetenzen werden bei Penetrationstests, Sicherheitsanalysen, technischen Audits und Schwachstellenbewertungen benötigt. Sie helfen, reale Risiken nicht nur theoretisch zu beschreiben, sondern praktisch und schadensvermeidend zu überprüfen.
Wieso ist es interessant? Das Modul zeigt Sicherheit aus einer aktiven Prüfperspektive. Wer versteht, wie Angriffe methodisch entstehen, kann Schutzmaßnahmen besser bewerten und gezielter verbessern. Rechtliche und ethische Grenzen machen dabei deutlich, dass technische Fähigkeiten Verantwortung erfordern.

BIN434: Digitale Forensik

Modulverantwortung	Prof. Dr. Nils Kopal	**Lehrperson(en)**	Prof. Dr. Nils Kopal Prof. Dr.-Ing. Martin Grothe
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Digital Forensics
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN008: Grundlagen der IT-Sicherheit: Angriffe, Schwachstellen, Sicherheitsvorfälle.
- BIN005: Systemsoftware: Betriebssysteme, Dateisysteme, Prozesse.
- BIN007: Datennetze: Protokolle, IP-Netze, Netzwerkdienste.
- BIN432: Sichere Ausführungsumgebungen: Isolation, Härtung, Systemkonfiguration.
- BIN433: Ethisches Hacken: Schwachstellen, Angriffswege, Reporting.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, einen digitalen Sicherheitsvorfall methodengerecht zu untersuchen, relevante Spuren nachvollziehbar zu sichern und aus den erhobenen Befunden eine fachlich belastbare Ereignisrekonstruktion abzuleiten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Untersuchungsauftrag, rechtliche Rahmenbedingungen und Zuständigkeiten bestimmen,
- digitale Spuren und Beweismittel identifizieren und hinsichtlich Flüchtigkeit und Bedeutung priorisieren,
- forensische Sicherungen unter Wahrung von Integrität, Nachvollziehbarkeit und Chain of Custody planen,
- Datenträger-, Dateisystem-, Betriebssystem-, Speicher-, Netzwerk- und Logdaten untersuchen,
- Zeitangaben, Benutzeraktivitäten, Prozesse, Dateien und Netzwerkverbindungen korrelieren,
- Hypothesen über Ablauf, Ursache und Auswirkungen eines Sicherheitsvorfalls entwickeln und prüfen,
- Grenzen und Unsicherheiten forensischer Verfahren bewerten,
- Untersuchungsschritte, Befunde und Ergebnisse reproduzierbar für technische und nichttechnische Adressaten aufbereiten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um digitale Sicherheitsvorfälle nachvollziehbar zu rekonstruieren und belastbare Grundlagen für Reaktion, Wiederherstellung und weitere Entscheidungen bereitzustellen. Das Modul bereitet auf das Capstone-Projekt IT-Sicherheit vor.

Inhalte

- Ziele, Phasen und Grenzen digitaler Forensik
- Rechtliche und organisatorische Rahmenbedingungen, Autorisierung und Datenschutz
- Chain of Custody, Beweissicherung, Schreibschutz, forensische Abbilder und Hashwerte
- Datenträger- und Dateisystemforensik
- Betriebssystemforensik, Benutzerkonten, Sitzungen, Persistenz und Dienste
- Loganalyse, Korrelation und Manipulationserkennung
- Netzwerkforensik und Grenzen verschlüsselter Kommunikation
- Speicherforensik, Prozesse, Module, Verbindungen und Indikatoren
- Zeitlinienanalyse und Ereignisrekonstruktion
- Hypothesenbildung, Befundbewertung, Unsicherheiten und Lücken
- Incident Response, Eindämmung, Wiederherstellung und Lessons Learned
- Forensische Berichterstattung und adressatengerechte Präsentation

Prüfungsvorleistung

Erfolgreiche Bearbeitung ausgewählter forensischer Laborübungen mit Kurzprotokollen; unbenotet.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektarbeit in Form einer forensischen Falluntersuchung mit schriftlichem Untersuchungsbericht und individuellem Fachgespräch (20 Minuten) ausgestaltet. Grundlage ist ein bereitgestellter Datensatz zu einem simulierten Sicherheitsvorfall.

Literatur

- Brian Carrier: *File System Forensic Analysis*. Addison-Wesley, 2005.
- Eoghan Casey: *Digital Evidence and Computer Crime*. 3. Auflage, Academic Press, 2011.
- Bill Nelson, Amelia Phillips, Christopher Steuart: *Guide to Computer Forensics and Investigations*. 6. Auflage, Cengage, 2019.
- Jason Luttgens, Matthew Pepe, Kevin Mandia: *Incident Response and Computer Forensics*. 3. Auflage, McGraw-Hill, 2014.
- Michael Hale Ligh, Andrew Case, Jamie Levy, Aaron Walters: *The Art of Memory Forensics*. Wiley, 2014.

• National Institute of Standards and Technology: *Guide to Integrating Forensic Techniques into Incident Response*. NIST SP 800-86, 2006.

Kurzporträt BIN434: Digitale Forensik

Worum geht es? Digitale Forensik befasst sich mit der methodischen Sicherung, Untersuchung und Bewertung digitaler Spuren. Studierende analysieren Daten aus Datenträgern, Betriebssystemen, Arbeitsspeicher, Netzwerken und Protokolldateien. Ziel ist eine nachvollziehbare Rekonstruktion von Ereignissen.
Wofür braucht man es? Forensische Kompetenzen werden in Incident-Response-Teams, Security Operations Centern, Beratungsunternehmen, internen Sicherheitsabteilungen und Strafverfolgungskontexten benötigt. Sie schaffen belastbare Grundlagen für Eindämmung, Wiederherstellung und weitere Entscheidungen nach Sicherheitsvorfällen.
Wieso ist es interessant? Digitale Angriffe und Fehlhandlungen hinterlassen Spuren, die oft über viele Quellen verteilt sind. Das Modul zeigt, wie aus fragmentarischen Daten eine begründete Geschichte entsteht und warum Integrität, Reproduzierbarkeit und saubere Dokumentation entscheidend sind.

BIN435: Capstone IT-Sicherheit

Modulverantwortung	Prof. Dr. Nils Kopal	**Lehrperson(en)**	Prof. Dr. Nils Kopal Prof. Dr.-Ing. Martin Grothe Professor:innen des Studiengangs Informatik Lehrende des Sprachenzentrums
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch und Englisch
Angebot	Wintersemester	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | 2 P | 2 S	**Engl. Titel**	Capstone IT Security
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN431: Angewandte Kryptografie: Verfahren, Schlüsselmanagement, sichere Updates.
- BIN432: Sichere Ausführungsumgebungen: Isolation, Härtung, Zugriffskontrolle.
- BIN433: Ethisches Hacken: Penetrationstests, Schwachstellenbewertung, Reporting.
- BIN434: Digitale Forensik: Spurensicherung, Ereignisrekonstruktion, Incident Response.

Zugehörige Lehrveranstaltungsteile

- *BIN435a: IT-Sicherheit - Capstone-Projekt*: 3 ECTS, 2 P
- *BIN405b: Englische Sprachkompetenz*: 2 ECTS, 2 S

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, die Sicherheit eines komplexen digitalen Systems über den gesamten Bearbeitungszyklus von der Analyse über die kontrollierte Sicherheitsprüfung bis zur Absicherung und Wiederherstellung selbstständig im Team zu untersuchen, weiterzuentwickeln und ihre Ergebnisse in englischer Sprache nachvollziehbar zu vertreten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Zielsystem, Komponenten, Datenflüsse und Vertrauensbeziehungen erfassen sowie Schutzgüter, Bedrohungen und Angriffsflächen in ein Bedrohungsmodell überführen,
- technische und organisatorische Sicherheitsanforderungen formulieren,
- kryptografische, berechtigungsbezogene, netzwerkseitige und isolierende Schutzmechanismen bewerten,
- einen Prüfungsauftrag mit Zielen, Umfang, Ausschlüssen und Kommunikationswegen planen,
- Methoden für Informationsgewinnung, Schwachstellenanalyse und kontrollierte Verifikation einsetzen,
- digitale Spuren sichern, korrelieren und zur Ereignisrekonstruktion verwenden,
- Maßnahmen zur Härtung, Eindämmung, Bereinigung und Wiederherstellung entwickeln,
- Wirksamkeit durch Sicherheits-, Funktions- und Angriffstests nachweisen, Ergebnisse kommunizieren und rechtliche sowie ethische Auswirkungen reflektieren (*BIN435a*),
- ihr englisches Sprachniveau bestimmen, geeignete Lernangebote nutzen und ihre Sprachkompetenz bis zum Niveau B2 weiterentwickeln (*BIN405b*).

WOZU. Die Studierenden nutzen diese Kompetenzen, um komplexe Sicherheitsprobleme in realitätsnahen und international geprägten beruflichen Situationen ganzheitlich und verantwortungsbewusst zu bearbeiten. Sie verbinden Sicherheitsanalyse, Penetrationstest, Incident Response, Forensik, Systemhärtung, Projektorganisation und adressatengerechte Kommunikation.

Inhalte

Veranstaltungsteil *BIN435a: IT-Sicherheit - Capstone-Projekt*

- Realitätsnahes digitales Zielsystem, z. B. eingebettetes System, IoT-Gerät, Robotiksystem, Webanwendung oder Netzwerkkomponente
- Systemanalyse und Threat Modelling
- Schutzgüter, Bedrohungen, Angriffsflächen und Angriffspfade
- Kryptografie, Schlüssel- und Geheimnisverwaltung
- Sichere Kommunikation, Updates und Zugriffskontrolle
- Sandboxing, Containerisierung, Virtualisierung und Netzwerksegmentierung
- Planung, Durchführung und Dokumentation autorisierter Sicherheitsprüfungen
- Informationsgewinnung, Enumeration, Schwachstellenanalyse und kontrollierte Verifikation
- Sicherung, Korrelation und Auswertung digitaler Spuren
- Incident Response, Eindämmung, Bereinigung und Wiederherstellung
- Sicherheits-, Funktions- und Regressionstests
- Technische Dokumentation, Reporting, Management Summary, Präsentation, Teamarbeit, Reviews und Reflexion rechtlicher sowie ethischer Auswirkungen

Veranstaltungsteil *BIN405b: Englische Sprachkompetenz*

- Erhebung und Einordnung des individuellen Sprachniveaus

- Bestimmung des persönlichen Lernbedarfs
- Nutzung geeigneter Selbstlernangebote und Veranstaltungen des Sprachenzentrums
- Feedback und Reflexion des Lernfortschritts
- Vorbereitung auf englischsprachige Präsentation und Fachdiskussion

Prüfungsvorleistung

Testat im Veranstaltungsteil *BIN405b: Englische Sprachkompetenz*: Nachweis des englischen Sprachniveaus B2. Der Nachweis ist Voraussetzung für die Teilnahme an der Prüfungsleistung.

Im Veranstaltungsteil *BIN435a: IT-Sicherheit - Capstone-Projekt* sind zusätzlich Projektplan, Bedrohungsmodell, Freigabe der Rules of Engagement, Teilnahme an Meilensteingesprächen und nachvollziehbare Dokumentation individueller Projektbeiträge als unbenotete Leistungen zu erbringen. Die Sicherheitsprüfung beginnt erst nach Freigabe des Prüfungsplans und der Rules of Engagement.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektarbeit mit technischer Dokumentation, Systemdemonstration sowie Abschlusspräsentation und/oder individuellem Fachgespräch (20 Minuten) in englischer Sprache im Veranstaltungsteil *BIN435a: IT-Sicherheit - Capstone-Projekt* ausgestaltet. Bewertet wird ausschließlich die fachlich-informatische Leistung, insbesondere System- und Sicherheitsanalyse, Bedrohungsmodell, methodische Qualität der Prüfung, Schwachstellenbewertung, Ereignisrekonstruktion, Härtungsmaßnahmen, Wirksamkeitsnachweis, Dokumentationsqualität, Kommunikation, Teamarbeit sowie rechtliche und ethische Reflexion.

Literatur

- Ross Anderson: *Security Engineering*. 3. Auflage, Wiley, 2020.
- Adam Shostack: *Threat Modeling*. Wiley, 2014.
- Georgia Weidman: *Penetration Testing*. No Starch Press, 2014.
- Eoghan Casey: *Digital Evidence and Computer Crime*. 3. Auflage, Academic Press, 2011.
- William Stallings, Lawrie Brown: *Computer Security: Principles and Practice*. 4. Auflage, Pearson, 2018.
- Bundesamt für Sicherheit in der Informationstechnik: *IT-Grundschutz-Kompendium*. BSI, aktuelle Fassung.

Kurzporträt BIN435: Capstone IT-Sicherheit

Worum geht es? Im Capstone-Projekt bearbeiten Studierende einen zusammenhängenden Sicherheitsfall an einem realitätsnahen digitalen System. Sie analysieren das System, führen eine autorisierte Sicherheitsprüfung durch, sichern digitale Spuren, entwickeln Maßnahmen zur Absicherung und Wiederherstellung und vertreten ihre Ergebnisse in englischer Sprache.
Wofür braucht man es? Sicherheitsprobleme treten in der Praxis selten isoliert auf. Die Kompetenzen werden benötigt, wenn Angriffe, Fehlkonfigurationen, Schwachstellen, forensische Befunde und Härtungsmaßnahmen in einem Gesamtzusammenhang und in international geprägten Projektkontexten bearbeitet werden müssen.
Wieso ist es interessant? Das Modul bietet die Möglichkeit, ein komplexes System aus mehreren Sicherheitsperspektiven zu untersuchen. Studierende erleben, wie technische Analyse, Angriffssimulation, Forensik, Absicherung, Teamarbeit und englische Fachkommunikation zusammenwirken.

BIN451: Informations- und Prozessmanagement

Modulverantwortung	Prof. Dr. Christoph Quix	**Lehrperson(en)**	Prof. Dr. Christoph Quix
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	jährlich	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Information and Process Management
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN101: Data Engineering: konzeptuelle Datenmodellierung, relationale Daten, SQL.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, betriebliche Informations- und Prozesssysteme auf Grundlage konzeptueller Modelle und verfügbarer Prozessdaten systematisch zu analysieren und begründete Gestaltungsmaßnahmen für Informationsstrukturen, Datenverantwortlichkeiten und Geschäftsprozesse abzuleiten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- Typen betrieblicher Informationssysteme hinsichtlich Aufgaben, Datenbeständen und Prozesseinbindung einordnen,
- Aufgaben des Informationsmanagements auf strategischer, administrativer und operativer Ebene analysieren,
- Informationsobjekte und semantische Zusammenhänge mit konzeptuellen Modellen, Knowledge Graphs und Semantic-Web-Konzepten beschreiben,
- Ziele einer Datenstrategie und Strukturen der Data Governance für Fallbeispiele entwickeln,
- Geschäftsprozesse identifizieren, abgrenzen und mit BPMN modellieren,
- Prozessmodelle anhand qualitativer Kriterien und quantitativer Prozesskennzahlen untersuchen,
- einfache Simulationsmodelle für Geschäftsprozesse erstellen und Ergebnisse interpretieren,
- Prozessdaten analysieren, grundlegende Verfahren des Process Discovery anwenden und Ergebnisse fachlich begründen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um an der fachlich fundierten Analyse und Gestaltung daten- und prozessorientierter Informationssysteme in Unternehmen und öffentlichen Organisationen mitzuwirken. Sie bilden eine Grundlage für Business Analysis, Informationsmanagement, Data Governance, Prozessmanagement und IT-Beratung.

Inhalte

- Unternehmensführung, Management und betriebliche Wertschöpfung
- Funktionale und prozessorientierte Sicht auf Unternehmen
- ERP-, CRM-, SCM-, BI- und Workflow-Systeme
- Unternehmensarchitekturen und Business-IT-Alignment
- Konzeptuelle Informationsmodellierung und fachliche Begriffsmodelle
- Knowledge Graphs, Semantic-Web-Technologien, Data Governance, Datenqualitätsmanagement und Stammdatenmanagement
- Geschäftsprozessmodellierung, Prozesslandkarten und Prozessverantwortung
- BPMN-Modellierung einschließlich ausgewählter Ereignistypen
- Zusammenhang zwischen Geschäftsprozessen, Informationssystemen und Prozessausführung
- Qualitative und quantitative Prozessanalyse und Prozesssimulation
- Process Data Science und Process Discovery
- Soziotechnische und organisatorische Aspekte der Gestaltung von Informations- und Prozesssystemen

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Portfolio aus mehreren semesterbegleitenden Modellierungs- und Analyseaufgaben zu einem durchgängigen betrieblichen Fallbeispiel einschließlich Präsentation der Ergebnisse und Fachgespräch ausgestaltet.

Literatur

- Helmut Krcmar: *Einführung in das Informationsmanagement*. 2. Auflage, Springer Gabler, 2015.
- Marlon Dumas, Marcello La Rosa, Jan Mendling, Hajo A. Reijers: *Fundamentals of Business Process Management*. 2. Auflage, Springer, 2018.
- Wil M. P. van der Aalst: *Process Mining*. 2. Auflage, Springer, 2016.
- Mathias Weske: *Business Process Management*. 3. Auflage, Springer, 2019.

Kurzporträt BIN451: Informations- und Prozessmanagement

Worum geht es? Das Modul betrachtet Informationssysteme, Daten und Geschäftsprozesse als zusammenhängende Bestandteile betrieblicher Organisationen. Studierende beschreiben Informationsstrukturen und Geschäftsprozesse mit konzeptuellen Modellen und analysieren sie hinsichtlich ihrer Gestaltung. Informationsmanagement, Data Governance, Geschäftsprozessmodellierung und Prozessanalyse werden verbunden.
Wofür braucht man es? Unternehmen und öffentliche Organisationen müssen Daten, Informationssysteme und Prozesse gemeinsam gestalten. Die Kompetenzen helfen bei Business Analysis, IT-Beratung, Prozessmanagement und der Einführung oder Weiterentwicklung betrieblicher Informationssysteme.
Wieso ist es interessant? Das Modul zeigt, warum technische Informationssysteme immer auch organisatorische Strukturen abbilden. Wer Prozesse, Daten und Verantwortlichkeiten gemeinsam modelliert, erkennt Gestaltungsspielräume und kann Verbesserungen nachvollziehbar begründen.

BIN452: Fortgeschrittene Java-Programmierung

Modulverantwortung	Prof. Dr. Thomas Nitsche	**Lehrperson(en)**	Prof. Dr. Thomas Nitsche
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch
Angebot	jährlich	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	2 V | 2 Ü | - P | - S	**Engl. Titel**	Advanced Java Programming
Workload	• Präsenz: 45 Std. • Selbst: 90 Std.		

Empfohlene Voraussetzungen

- BIN201: Weiterführende Programmentwicklung: objektorientierte Programmierung, Tests, Softwareentwurf.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, fortgeschrittene objektorientierte Java-Programme, persistente Datenbankanbindungen und verteilte mehrschichtige Unternehmensanwendungen zu erläutern, zu analysieren, zu bewerten und praktisch umzusetzen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- allgemeine Entwurfs- und Implementierungsansätze objektorientierter Programme mit Java anwenden,
- Eigenschaften objektorientierter Typsysteme wie Subtyping, Generics und Typsicherheit analysieren,
- Persistenzlösungen mit JDBC und JPA erläutern, auswählen, entwickeln und bewerten,
- Probleme beim objektrelationalen Mapping erläutern und Lösungsansätze anwenden,
- Konzepte von Jakarta EE für komplexe, verteilte Unternehmensanwendungen einordnen,
- komplexe, heterogene und plattformübergreifende mehrschichtige Anwendungen mit Java-basiertem Backend entwickeln,
- moderne Entwicklungswerkzeuge sowie Test-, Persistenz- und Framework-Technologien anwenden,
- Entwurfsentscheidungen analysieren, Implementierungen testen und die Qualität ihrer Lösungen bewerten.

WOZU. Die Studierenden nutzen diese Kompetenzen, um robuste, wartbare und datenbankgestützte Java-Unternehmensanwendungen zu entwerfen, zu implementieren und weiterzuentwickeln. Sie können fortgeschrittene Konzepte der Softwareentwicklung zielgerichtet einsetzen und sich sicher in professionelle Java-Softwareprojekte einarbeiten.

Inhalte

- Virtuelle Maschinen und plattformunabhängige Programmierung
- Nebenläufigkeit, Threads und Serialisierung
- Reflection, Annotationen und dynamische Laufzeitinformationen
- Objektorientierte Typsysteme, Subtyping, Liskovsches Substitutionsprinzip, Varianz und Generics
- Testen mit JUnit, Build-Prozesse mit Maven oder Gradle, Logging und Javadoc
- Datenbankanbindung mit JDBC
- Persistenz mit JPA
- Objektrelationales Mapping
- Jakarta EE, Anwendungsserver, Servlets, Enterprise Beans, Transaktionen und Contexts and Dependency Injection
- Entwicklung komplexer verteilter Anwendungen mit unterschiedlichen Client-Technologien und Java-basiertem Backend

Prüfungsvorleistung

Bearbeitung ausgewählter Übungsaufgaben.

Prüfungsleistung

Mündliche Prüfung.

Literatur

- Christian Ullenboom: *Java ist auch eine Insel*. Rheinwerk Computing, aktuelle Auflage.
- Alexander Salvanos: *Professionell entwickeln mit Java EE 8*. Rheinwerk Computing, 2018.
- Michael Inden: *Der Weg zum Java-Profi*. 4. Auflage, dpunkt.verlag, 2017.
- Joshua Bloch: *Effective Java*. 3. Auflage, Addison-Wesley, 2018.

Kurzporträt BIN452: Fortgeschrittene Java-Programmierung

Worum geht es? Das Modul vertieft Java über grundlegende Programmierung hinaus. Studierende beschäftigen sich mit Typsystemen, Reflection, Persistenz, Datenbankanbindung, Jakarta EE und mehrschichtigen Unternehmensanwendungen. Dabei verbinden sie Sprachkonzepte mit praktischer Softwareentwicklung.
Wofür braucht man es? Java ist in vielen professionellen Softwareprojekten verbreitet, insbesondere bei Backend- und Unternehmensanwendungen. Die Kompetenzen helfen, bestehende Anwendungen zu verstehen, robuste Komponenten zu entwickeln, Datenbanken anzubinden und technische Entscheidungen begründet zu treffen.
Wieso ist es interessant? Fortgeschrittene Java-Programmierung zeigt, wie aus objektorientierten Grundlagen größere, verteilte und datenbankgestützte Systeme entstehen. Studierende sehen, wie Typsicherheit, Frameworks, Persistenz und Architekturentscheidungen zusammen die Qualität professioneller Software beeinflussen.

BIN499: Ausgewählte Fragen der Informatik

Modulverantwortung	Studiendekan:in des Studiengangs Informatik	**Lehrperson(en)**	Alle Lehrenden des Studiengangs
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Wahl	**Sprache**	Deutsch oder Englisch
Angebot	nach Anerkennung oder gemäß Wahlpflichtangebot	**Dauer**	1 Semester
Credits	5 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	nach anerkannter Lehrveranstaltung	**Engl. Titel**	Selected Topics in Computer Science
Workload	Präsenzstudium: nach anerkannter Lehrveranstaltung Selbststudium: nach anerkannter Lehrveranstaltung		

Empfohlene Voraussetzungen

Keine angegeben.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, ausgewählte informatische Fragestellungen ergänzend zum regulären Wahlpflichtangebot fachlich fundiert zu bearbeiten und die dabei erworbenen Kompetenzen in ihr individuelles Studienprofil einzuordnen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- fachliche Inhalte aus einem anerkannten oder besonders ausgewiesenen Informatikangebot erschließen,
- geeignete informatische Methoden, Werkzeuge oder Technologien anwenden,
- Aufgabenstellungen, Lösungswege und Ergebnisse nachvollziehbar dokumentieren,
- die erworbenen Kompetenzen in Bezug auf das eigene Studium und mögliche Berufsfelder reflektieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um ihr Informatikstudium durch anerkannte externe, internationale oder wechselnde interne Lehrangebote sinnvoll zu ergänzen und fachliche Schwerpunkte individuell auszubauen.

Inhalte

- Anerkannte oder besonders ausgewiesene informatische Lehrangebote
- Wechselnde fachliche Vertiefungen aus aktuellen Bereichen der Informatik
- Themen, Methoden und Werkzeuge abhängig von der konkreten Lehrveranstaltung
- Fachliche Einordnung der erworbenen Kompetenzen in das Informatikstudium

Prüfungsvorleistung

Nach anerkannter Lehrveranstaltung.

Prüfungsleistung

Nach anerkannter Lehrveranstaltung oder gemäß Anerkennungsentscheidung.

Literatur

- Literatur, Dokumentation, Standards und Spezifikationen werden abhängig von den konkreten Inhalten durch die Lehrperson oder im Rahmen der Anerkennung bekanntgegeben.

Kurzporträt BIN499: Ausgewählte Fragen der Informatik

Worum geht es? Das Modul dient als flexibles Wahlpflichtmodul für anerkannte oder besonders ausgewiesene informatische Lehrangebote. Die konkreten Inhalte hängen von der jeweiligen Lehrveranstaltung oder Anerkennungsentscheidung ab.
Wofür braucht man es? Es ermöglicht Studierenden, externe, internationale oder wechselnde interne Angebote in ihr Studium einzubringen und ihr fachliches Profil individuell zu ergänzen.
Wieso ist es interessant? Informatik entwickelt sich schnell und wird an vielen Orten mit unterschiedlichen Schwerpunkten gelehrt. Das Modul schafft Raum, passende zusätzliche Kompetenzen anzuerkennen und sinnvoll in das Studium einzubinden.

Projekt, Praxisphase und Bachelorarbeit

Der Modulbereich Projekt, Praxisphase und Bachelorarbeit führt die im Studium erworbenen fachlichen, methodischen und überfachlichen Kompetenzen in größeren anwendungsbezogenen Aufgaben zusammen. Die Studierenden bearbeiten informatische Projekte im Team, wenden ihre Kenntnisse in professionellen Praxisumgebungen an und schließen das Studium mit einer eigenständigen Bachelorarbeit samt Kolloquium ab.

Zusammenfassung der Lernziele / Kompetenzen

- **WAS**: Die Studierenden können nach Abschluss des Modulbereichs anwendungsbezogene Informatikaufgaben im Team bearbeiten, berufspraktische Aufgabenstellungen in professionellen Kontexten reflektiert umsetzen und eine eigenständige informatische Fragestellung wissenschaftlich oder anwendungsorientiert ausarbeiten. Sie entwickeln dabei insbesondere Kompetenzen in Projektarbeit, Kommunikation, Dokumentation, Selbstorganisation, fachlicher Vertretung und beruflicher Orientierung.
- **WOMIT**: Die Studierenden erreichen dieses Lernergebnis, indem sie im Modulbereich
  - eine anwendungsbezogene Projektaufgabe analysieren, planen, technisch bearbeiten, präsentieren und im Fachgespräch vertreten,
  - Projektmanagement, betriebswirtschaftliche Grundlagen und soziale Verantwortung auf informatische Projektarbeit beziehen,
  - in einer Praxisstelle eine dem Ausbildungsstand entsprechende informatische Aufgabe bearbeiten und dokumentieren,
  - berufspraktische Erfahrungen im begleitenden Seminar darstellen und reflektieren,
  - eine Bachelorarbeit eigenständig planen, durchführen, auswerten und schriftlich ausarbeiten,
  - Ergebnisse der Bachelorarbeit im Kolloquium präsentieren, einordnen und fachlich verteidigen.
- **WOZU**: Damit sie den Übergang vom Studium in berufliche oder weiterführende wissenschaftliche Kontexte vorbereitet und nachvollziehbar gestalten können. Der Modulbereich bündelt technische, organisatorische, soziale und kommunikative Kompetenzen und zeigt, dass informatische Arbeit immer auch Planung, Abstimmung, Verantwortung und verständliche Ergebnisdarstellung erfordert.

Kurzporträt Projekt, Praxisphase und Bachelorarbeit

Worum geht es? Dieser Modulbereich bildet den anwendungs- und abschlussorientierten Teil des Studiums. Die Studierenden bearbeiten ein größeres Informatikprojekt, absolvieren eine Praxisphase und erstellen ihre Bachelorarbeit mit Kolloquium. Dabei werden fachliche Inhalte aus dem Studium in realitätsnahe, berufspraktische und eigenständige Aufgaben überführt.
Wofür braucht man es? Die Module bereiten auf den Berufseinstieg und auf weiterführende Studienwege vor. Studierende üben, Aufgaben selbstständig und im Team zu strukturieren, Ergebnisse zu dokumentieren, mit Auftraggebenden oder Praxisstellen zu kommunizieren und fachliche Entscheidungen überzeugend zu vertreten. Die Bachelorarbeit zeigt die Fähigkeit zu eigenständigem informatischem Arbeiten.
Wieso ist es interessant? In diesem Bereich wird sichtbar, wie das bisherige Studium in größere Zusammenhänge übergeht. Projekte, Praxisphase und Bachelorarbeit eröffnen die Möglichkeit, eigene Interessen zu vertiefen, reale Arbeitsfelder kennenzulernen und fachliche Kompetenzen an konkreten Aufgaben zu zeigen. Dadurch erhält das Studium einen klaren Transfer in Praxis und Abschluss.

BIN501: Anwendungsbezogenes Informatik-Projekt

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	Alle Professor:innen des Studiengangs
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	10 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | 3 P | 1 S | 2 SL	**Engl. Titel**	Application-Oriented Computer Science Project
Workload	Anwendungsbezogenes Informatik-Projekt: 35 Stunden Präsenzstudium und 165 Stunden Selbststudium Projektmanagement und betriebswirtschaftliche Grundlagen: 25 Stunden Präsenzstudium und 30 Stunden Selbststudium Sozial.Punkt!: 15 Stunden Präsenzstudium Gesamtworkload: 270 Stunden		

Empfohlene Voraussetzungen

- Erfolgreicher Abschluss der Module der ersten vier Fachsemester, insbesondere aus den Bereichen Programmierung, Softwareentwicklung, Datenbanken und Datenmodellierung, Künstliche Intelligenz und Daten, IT-Systeme, Rechnernetze, IT-Sicherheit sowie Software Engineering.
- Fähigkeit, informatische Aufgabenstellungen zu analysieren, Softwarelösungen zu entwerfen und umzusetzen, geeignete Methoden und Werkzeuge auszuwählen sowie Arbeitsergebnisse nachvollziehbar darzustellen.

Zugehörige Lehrveranstaltungsteile

- *BIN501a: Anwendungsbezogenes Informatik-Projekt*: 7 ECTS, 3 P
- *BIN501b: Projektmanagement und betriebswirtschaftliche Grundlagen*: 2 ECTS, 2 SL
- *BIN501c: Sozial.Punkt!*: 1 ECTS, 1 S

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine offene informatische Aufgabenstellung mit realem oder realitätsnahem Anwendungskontext im Team als Projekt zu strukturieren, mit fachlichen Ansprechpersonen abzustimmen und zu einer präsentierbaren Softwarelösung zu bearbeiten. Sie können Anforderungen, Methoden, Technologien, Architekturentscheidungen sowie technische, organisatorische und wirtschaftliche Rahmenbedingungen begründen und fachlich vertreten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine informatische Projektaufgabe analysieren, Anforderungen abstimmen sowie Projektziele, Erfolgskriterien und bearbeitbare Teilaufgaben ableiten,
- alternative Lösungsansätze entwickeln und geeignete Methoden, Modelle, Technologien und Werkzeuge begründet auswählen,
- eine prototypische oder produktnahe Softwarelösung implementieren, testen, evaluieren und weiterentwickeln,
- die Zusammenarbeit im Team organisieren, den Projektfortschritt vorstellen und Ergebnisse im Fachgespräch begründen,
- einen Projektauftrag mit Meilensteinen, Arbeitspaketen und Verantwortlichkeiten formulieren und geeignete Vorgehensweisen projektbezogen anwenden,
- Projektaufwände, Fortschritt und Risiken einschätzen sowie geeignete Steuerungs- und Gegenmaßnahmen planen,
- Anwendungskontext, Anspruchsgruppen, Kostenstrukturen, Nutzenpotenziale und Wirtschaftlichkeit anhand eines Business Case analysieren,
- technische und wirtschaftliche Lösungsalternativen, insbesondere Make-or-Buy-, Lizenzierungs- und Betriebsentscheidungen, strukturiert vergleichen,
- sich im Bestandteil Sozial.Punkt! in einem sozialen Kontext engagieren oder die soziale Verantwortung informatischen Handelns reflektieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um offene informatische Aufgabenstellungen in praxisnahen Softwareentwicklungs- und IT-Projektkontexten eigenständig und kooperativ zu bearbeiten. Sie sind darauf vorbereitet, Anforderungen mit fachlichen Ansprechpersonen zu klären, technische und wirtschaftliche Entscheidungen nachvollziehbar zu treffen, Projektverläufe unter Unsicherheit zu steuern und Verantwortung für ein gemeinsames Projektergebnis zu übernehmen.

Inhalte

Inhalte des anwendungsbezogenen Informatik-Projekts

- Analyse und Konkretisierung einer offenen informatischen Projektaufgabe
- Anforderungsbeschreibung und Abstimmung mit fachlichen Ansprechpersonen
- Festlegung von Projektzielen, Projektumfang und Erfolgskriterien
- Entwicklung und Bewertung alternativer Lösungsansätze
- Auswahl geeigneter Methoden, Technologien, Werkzeuge und Softwarearchitekturen
- Implementierung, Test, Qualitätssicherung und Evaluation einer Softwarelösung
- Teamorganisation, Aufgabenkoordination und regelmäßige Vorstellung des Projektfortschritts
- Präsentation, Demonstration und Dokumentation der Projektergebnisse und individuellen Beiträge

Inhalte des Bestandteils Projektmanagement und betriebswirtschaftliche Grundlagen

- Projektauftrag, Projektziele, Arbeitspakete, Rollen und Meilensteine
- Klassische, agile und hybride Vorgehensweisen in IT-Projekten
- Aufwandsschätzung, Zeitplanung, Fortschrittskontrolle und Änderungsmanagement
- Risiko-, Qualitäts- und Entscheidungsmanagement
- Betrieblicher Anwendungskontext, Stakeholdernutzen und Wertbeitrag von IT-Lösungen
- Kostenstrukturen und Lebenszykluskosten von Software- und IT-Systemen
- Nutzenbewertung, Wirtschaftlichkeitsrechnung und Business Case
- Make-or-Buy-, Lizenzierungs-, Beschaffungs- und Betriebsentscheidungen

Inhalte des Bestandteils Sozial.Punkt!

- Soziales Engagement beziehungsweise Reflexion sozialer Verantwortung im Rahmen von Sozial.Punkt!
- Einordnung sozialer Verantwortung in informatische Aufgaben- und Projektkontexte
- Dokumentation und Reflexion der im sozialen Kontext gewonnenen Erfahrung

Prüfungsvorleistung

Prüfungsvorleistung ist der Nachweis zum Bestandteil Sozial.Punkt!

Prüfungsleistung

Studien- oder Projektarbeit. Das Prüfungskonzept ist als Projektarbeit mit Präsentation und Fachgespräch ausgestaltet. Die Projektarbeit umfasst die entwickelte Softwarelösung, die für die Durchführung und Bewertung wesentlichen Projektartefakte sowie eine Projektausarbeitung. Präsentation, Demonstration und Fachgespräch bilden den Schwerpunkt der Bewertung. Die Projektausarbeitung dient der nachvollziehbaren Darstellung der Aufgabenstellung, des Vorgehens, der wesentlichen technischen und wirtschaftlichen Entscheidungen, der erzielten Ergebnisse sowie der individuellen Beiträge. Bei Teamleistungen müssen die individuellen Beiträge der Studierenden erkennbar und bewertbar sein.

Literatur

- Ian Sommerville: *Software Engineering*. Pearson.
- Bernd Brügge; Allen H. Dutoit: *Objektorientierte Softwaretechnik mit UML, Entwurfsmustern und Java*. Pearson; Gernot Starke: *Effektive Softwarearchitekturen*. Hanser.
- Chris Rupp: *Requirements-Engineering und -Management*. Hanser; Robert C. Martin: *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall.
- Ken Schwaber; Jeff Sutherland: *The Scrum Guide*.
- Kathy Schwalbe: *Information Technology Project Management*. Cengage; Philip Junge: *BWL für Ingenieure*. Springer Gabler; Jens Kaufmann; Wilhelm Mülder: *Grundkurs Wirtschaftsinformatik*. Springer Vieweg.
- Weitere projektbezogene Literatur, technische Dokumentationen, Standards und Spezifikationen werden abhängig von der konkreten Aufgabenstellung semesterbezogen ergänzt.

Kurzporträt BIN501: Anwendungsbezogenes Informatik-Projekt

Worum geht es? In diesem Modul bearbeiten Studierende eine offene informatische Aufgabenstellung mit realem oder realitätsnahem Anwendungskontext im Team. Sie klären Anforderungen, planen das Projekt, entwickeln eine Softwarelösung und beziehen technische, organisatorische, wirtschaftliche und soziale Aspekte ein.
Wofür braucht man es? Die Kompetenzen werden für praxisnahe Softwareentwicklungs- und IT-Projekte benötigt, in denen Anforderungen unvollständig sind, Entscheidungen begründet werden müssen und Teamarbeit, Projektsteuerung und Wirtschaftlichkeit eine wichtige Rolle spielen.
Wieso ist es interessant? Das Modul zeigt, wie aus einer zunächst offenen Aufgabe ein tragfähiges Projektergebnis entsteht. Studierende übernehmen Verantwortung für ein gemeinsames Ergebnis und erleben, wie technische Lösungen, Projektmanagement, Business Case und soziale Verantwortung zusammenwirken.

BIN601: Praxisphase

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	Alle Lehrenden des Studiengangs
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	15 ECTS	**Benotung**	unbenotet
SWS	- V | - Ü | - P | 1 S	**Engl. Titel**	Practical Phase
Workload	Praktische Tätigkeit in der Praxisstelle: 440 Stunden Seminar zur Praxisphase: 10 Stunden Gesamtworkload: 450 Stunden		

Empfohlene Voraussetzungen

- Prüfungsordnung: formale Zulassungsvoraussetzungen zur Praxisphase.
- Vorhergehende Module im Studienplan: informatische Methoden, Projektarbeit, Dokumentation, professionelle Anwendungskontexte.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine realistische berufspraktische Aufgabenstellung der Informatik in einem betrieblichen oder vergleichbaren professionellen Kontext fachlich begründet, kooperativ und reflektiert zu bearbeiten.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- sich in fachliche, technische, organisatorische und soziale Rahmenbedingungen einer Praxisstelle einarbeiten,
- eine ihrem Ausbildungsstand entsprechende informatische Aufgabenstellung analysieren, strukturieren und unter Berücksichtigung gegebener Anforderungen bearbeiten,
- geeignete informatische Methoden, Modelle, Werkzeuge und Vorgehensweisen aufgabenangemessen auswählen und anwenden,
- selbstständig oder im Team in professionellen Arbeits- und Projektstrukturen mitarbeiten,
- Arbeitsergebnisse, Entscheidungen und Vorgehensweisen nachvollziehbar dokumentieren,
- Erfahrungen, Arbeitsinhalte und Ergebnisse im begleitenden Seminar darstellen und fachlich reflektieren.

WOZU. Die Studierenden nutzen diese Kompetenzen, um den Übergang von der hochschulischen Ausbildung in informatische Berufsfelder vorzubereiten und eine tragfähige Grundlage für die anschließende Bachelorarbeit sowie den Berufseinstieg zu entwickeln. Die Praxisphase umfasst in der Regel elf Wochen; für Studierende des Teilzeitstudiengangs kann sie auf Antrag bis zu 18 Wochen umfassen.

Inhalte

- Einarbeitung in Aufgaben, Arbeitsweisen und organisatorische Strukturen einer Praxisstelle
- Bearbeitung einer informatischen Aufgabenstellung in einem Unternehmen, einer öffentlichen Einrichtung, einer Forschungseinrichtung, einer Hochschule oder einer vergleichbaren professionellen Praxisumgebung
- Anwendung geeigneter informatischer Methoden, Modelle, Werkzeuge und Vorgehensweisen
- Mitarbeit in Arbeits-, Entwicklungs-, Betriebs-, Analyse- oder Projektkontexten der Informatik
- Einbindung in fachliche, organisatorische und gegebenenfalls abteilungsübergreifende Arbeitszusammenhänge
- Aufarbeitung berufspraktischer Erfahrungen im begleitenden Seminar
- Dokumentation von Aufgabenstellung, Problemumfeld, Vorgehen, Lösungswegen und erzielten Ergebnissen
- Reflexion der Erfahrungen und Eindrücke aus der praktischen Tätigkeit
- Erstellung eines Praxisphasenberichts

Prüfungsvorleistung

Teilnahme am Seminar zur Praxisphase, insbesondere an fünf Praxisphasenvortragsblöcken. Als Nachweis der ordnungsgemäßen Durchführung der Praxisphase ist eine qualifizierte Bescheinigung bzw. ein qualifiziertes Arbeitszeugnis der Praxisstelle vorzulegen.

Prüfungsleistung

Testat. Das Prüfungskonzept besteht aus einem Praxisphasenbericht als Nachweis der ordnungsgemäßen Durchführung der Praxisphase. Der Bericht beschreibt insbesondere die Aufgabenstellung, das Problemumfeld, die gewählten Lösungswege, gegebenenfalls erzielte Ergebnisse sowie Erfahrungen und Eindrücke aus der praktischen Tätigkeit. Der Umfang richtet sich nach der Praxisphasenordnung. Die Bewertung erfolgt unbenotet mit bestanden bzw. nicht bestanden.

Literatur

- themenabhängig

Kurzporträt BIN601: Praxisphase

Worum geht es? Die Praxisphase führt Studierende in eine professionelle Arbeitsumgebung der Informatik. Sie bearbeiten dort eine realistische Aufgabe, lernen betriebliche oder vergleichbare Strukturen kennen und reflektieren ihre Erfahrungen im begleitenden Seminar. Die Tätigkeit kann in Unternehmen, öffentlichen Einrichtungen, Forschungseinrichtungen, Hochschulen oder vergleichbaren Praxisstellen stattfinden.
Wofür braucht man es? Die Praxisphase bereitet den Übergang vom Studium in informatische Berufsfelder vor. Studierende wenden fachliche und methodische Kompetenzen unter realen organisatorischen Bedingungen an, dokumentieren ihre Arbeit und lernen, berufspraktische Erfahrungen fachlich einzuordnen. Dies unterstützt auch die Themenfindung und Vorbereitung der Bachelorarbeit.
Wieso ist es interessant? Studierende erleben Informatik außerhalb des Lehrveranstaltungsrahmens und sehen, wie fachliche Entscheidungen, Kommunikation, Verantwortung und Zusammenarbeit im Berufsalltag zusammenwirken. Die Praxisphase bietet Gelegenheit, eigene Interessen zu prüfen, Arbeitsfelder kennenzulernen und das bisherige Studium mit konkreter Berufserfahrung zu verbinden.

BIN602: Bachelorarbeit mit Kolloquium

Modulverantwortung	Studiendekan:in des Fachbereichs	**Lehrperson(en)**	Alle Lehrenden des Studiengangs
Verwendbarkeit	Bachelor Informatik	**Fächergruppe**	Informatik
Modultyp	Pflicht	**Sprache**	Deutsch
Angebot	jedes Semester	**Dauer**	1 Semester
Credits	15 ECTS	**Benotung**	Deutsche Notenskala 1-5
SWS	- V | - Ü | - P | 1 S	**Engl. Titel**	Bachelor's Thesis with Colloquium
Workload	Eigenständige Bearbeitung der Bachelorarbeit einschließlich Literatur- und Quellenrecherche, Konzeption, Durchführung, Auswertung und schriftlicher Ausarbeitung: 420 Stunden Vorbereitung und Durchführung des Kolloquiums: 30 Stunden Gesamtworkload: 450 Stunden		

Empfohlene Voraussetzungen

- Prüfungsordnung: formale Zulassungsvoraussetzungen zur Bachelorarbeit mit Kolloquium.
- Übrige Module des Studiengangs: fachliche Vertiefung, Methodenauswahl, Literaturrecherche, Dokumentation, Präsentation.

Lernergebnisse / Kompetenzen

WAS. Mit erfolgreichem Abschluss des Moduls sind die Studierenden in der Lage, eine anwendungsbezogene oder wissenschaftsorientierte Fragestellung der Informatik eigenständig, methodisch begründet und nachvollziehbar zu bearbeiten sowie die erzielten Ergebnisse schriftlich darzustellen, mündlich zu präsentieren und fachlich zu verteidigen.

WOMIT. Die Studierenden erreichen dieses Lernergebnis, indem sie

- eine informatische Aufgabenstellung erfassen, fachlich einordnen und in bearbeitbare Teilfragen strukturieren,
- den relevanten Stand von Wissenschaft, Technik und Anwendung durch eine aufgabenbezogene Literatur- und Quellenrecherche erschließen,
- geeignete informatische Methoden, Modelle, Verfahren und Werkzeuge auswählen, begründet anwenden und deren Eignung reflektieren,
- Lösungsansätze konzipieren, umsetzen, analysieren, bewerten oder vergleichend untersuchen,
- fachliche Entscheidungen, Vorgehensweisen, Annahmen, Ergebnisse und Grenzen der eigenen Arbeit nachvollziehbar dokumentieren,
- die Ergebnisse der Bachelorarbeit im Kolloquium adressatengerecht präsentieren, fachlich einordnen und in einer wissenschaftlich-fachlichen Diskussion verteidigen.

WOZU. Die Studierenden nutzen diese Kompetenzen, um komplexe informatische Aufgabenstellungen im späteren Berufsfeld oder in einem weiterführenden Studium selbstständig, methodisch fundiert und verantwortbar bearbeiten sowie ihre Ergebnisse fachlich überzeugend kommunizieren zu können.

Inhalte

- Eigenständige Bearbeitung einer informatischen Aufgabenstellung innerhalb eines festgelegten Bearbeitungszeitraums
- Fachliche Einordnung der Aufgabenstellung in den jeweiligen informatischen Kontext
- Literatur- und Quellenrecherche zum relevanten Stand von Wissenschaft, Technik und Anwendung
- Analyse der Problemstellung, Anforderungen, Rahmenbedingungen und Zielsetzungen
- Auswahl und Anwendung geeigneter informatischer Methoden, Modelle, Verfahren und Werkzeuge
- Konzeption, Umsetzung, Evaluation oder vergleichende Untersuchung eines Lösungsansatzes
- Kritische Reflexion der Vorgehensweise, Ergebnisse und Grenzen der eigenen Arbeit
- Schriftliche Ausarbeitung der Bachelorarbeit
- Vorbereitung der mündlichen Präsentation der Bachelorarbeit
- Präsentation und fachliche Verteidigung der Arbeit im Kolloquium

Prüfungsvorleistung

Keine angegeben.

Prüfungsleistung

Bachelorarbeit mit Kolloquium. Die Bachelorarbeit ist eine schriftliche Ausarbeitung zu einer eigenständig bearbeiteten informatischen Aufgabenstellung. Das Kolloquium umfasst die mündliche Präsentation und fachliche Verteidigung der Inhalte, Vorgehensweise und Ergebnisse der Bachelorarbeit.

Literatur

- themenabhängig

Kurzporträt BIN602: Bachelorarbeit mit Kolloquium

Worum geht es? In der Bachelorarbeit bearbeiten Studierende eine informatische Fragestellung eigenständig und stellen ihre Ergebnisse schriftlich dar. Im Kolloquium präsentieren sie Vorgehen, Ergebnisse und fachliche Einordnung und verteidigen ihre Arbeit in einer wissenschaftlich-fachlichen Diskussion. Das Modul bündelt die im Studium erworbenen fachlichen und methodischen Kompetenzen.
Wofür braucht man es? Die Bachelorarbeit zeigt, dass Studierende komplexere informatische Aufgaben selbstständig strukturieren, passende Methoden auswählen und Ergebnisse nachvollziehbar kommunizieren können. Diese Kompetenzen sind für den Berufseinstieg ebenso wichtig wie für ein weiterführendes Studium, in dem eigenständiges wissenschaftliches oder anwendungsorientiertes Arbeiten erwartet wird.
Wieso ist es interessant? Das Modul bietet die Möglichkeit, ein eigenes Thema vertieft zu bearbeiten und fachliche Interessen sichtbar zu machen. Studierende können zeigen, wie sie Analyse, Konzeption, Umsetzung, Bewertung und Kommunikation in einer zusammenhängenden Arbeit verbinden. Das Kolloquium macht die Ergebnisse diskutierbar und schließt das Studium fachlich ab.

Glossar

Dieses Glossar erklärt Abkürzungen und wiederkehrende Begriffe, die in den Modulbeschreibungen verwendet werden.

ECTS European Credit Transfer System. Ein ECTS-Punkt (auch „Credit") entspricht etwa 30 Stunden studentischem Arbeitsaufwand (Workload) und ist die europaweit vergleichbare Maßeinheit für den Umfang eines Moduls.

SWS Semesterwochenstunden. Gibt an, wie viele Zeitstunden Lehrveranstaltung pro Woche während der Vorlesungszeit eines Semesters stattfinden, aufgeschlüsselt nach Veranstaltungsart (siehe V, Ü, P, S/SL).

V Vorlesung. Lehrveranstaltungsform, in der fachliche Inhalte überwiegend im Vortrag vermittelt werden.

Ü Übung. Lehrveranstaltungsform zur Vertiefung und Einübung der in der Vorlesung vermittelten Inhalte, häufig anhand von Aufgaben.

P Praktikum. Praktische Lehrveranstaltung, in der Inhalte im Labor oder an Versuchsaufbauten selbst durchgeführt werden.

S Seminar. Lehrveranstaltungsform mit stärker eigenständiger, oft projekt- oder präsentationsorientierter Arbeitsweise.

SL Seminaristischer Unterricht. Lehrveranstaltungsform, die das freie Vortragen von Inhalten (wie in einer Vorlesung) mit dem interaktiven Dialog kombiniert.

Workload Der geschätzte studentische Gesamtarbeitsaufwand für ein Modul, aufgeteilt in Präsenzstudium (Teilnahme an Lehrveranstaltungen) und Selbststudium (eigenständige Vor- und Nachbereitung, Prüfungsvorbereitung).

Modultyp Kennzeichnet, ob ein Modul verpflichtend zu belegen ist (**Pflicht**) oder aus einem Angebot ausgewählt werden kann (**Wahlpflicht**).

WAS / WOMIT / WOZU Format zur Beschreibung von Lernergebnissen: **WAS** die Studierenden nach dem Modul können, **WOMIT** (mit welchen Inhalten und Tätigkeiten) sie dieses Lernergebnis erreichen und **WOZU** sie diese Kompetenzen im weiteren Studium oder Beruf benötigen.

Ziele-Module-Matrix Übersicht, die zeigt, in welchen Modulen die übergreifenden Qualifikationsziele eines Studiengangs aufgebaut und vertieft werden.

Capstone-Modul Abschließendes Modul eines Lernpfads, in dem die zuvor erworbenen fachlichen und methodischen Kompetenzen in einer größeren, integrierenden Projektaufgabe zusammengeführt und angewendet werden.

Lernpfad Thematisch gebündelte Abfolge von Pflichtmodulen, die inhaltlich aufeinander aufbauen und in einem Capstone-Modul münden. Lernpfade orientieren sich an einem typischen Berufs- bzw. Tätigkeitsprofil des Studiengangs.

Kurzporträt Kurzbeschreibung eines Moduls oder Lernpfads entlang der Fragen „Worum geht es¿', „Wofür braucht man es?" und „Wieso ist es interessant?", gedacht als schneller Einstieg für Studierende.