hochschule mannheim

Fakultät für Informatik

Modulhandbuch

Bachelor Informatik

Letzte Änderung: 2023-04-12 17:53:49

Überblick

Sem.	Vorlesung	SWS	ECTS
1	Einführung in die Informatik (EI)	4 SWS	5 ECTS
1	Mathematik für die Informatik 1 (MA1)	4 SWS	5 ECTS
1	Nebenfach - Grundlagen der BWL (NF)	4 SWS	5 ECTS
1	Programmierung 1 (PR1)	8 SWS	10 ECTS
1	Technische Informatik 1 (TEI1)	4 SWS	5 ECTS
2	Mathematik für die Informatik 2 (MA2)	4 SWS	5 ECTS
2	Programmierung 2 (PR2)	8 SWS	10 ECTS
2	Software Engineering 1 (SE1)	4 SWS	5 ECTS
2	Technische Informatik 2 (TEI2)	4 SWS	5 ECTS
2	Theoretische Informatik (THI)	4 SWS	5 ECTS
3	Datenmanagement (DM)	4 SWS	5 ECTS
3	Mathematik für die Informatik 3 (MA3)	4 SWS	5 ECTS
3	Parallele Programmierung (PP)	4 SWS	5 ECTS
3	Programmierung 3 (PR3)	4 SWS	5 ECTS
3	Software Engineering 2 (SE2)	4 SWS	5 ECTS
3	Webbasierte Systeme (WEB)	4 SWS	5 ECTS
4	Projekte in der Informatik (PI-IB)	8 SWS	10 ECTS
	Ausgewählte Probleme des Software Engineerings (APS)	3 SWS	
	Projektmanagement (PM)	3 SWS	
	Virtualisierung (VIR)	2 SWS	
4	Softwareprojekt (SP)	4 SWS	20 ECTS
	Fachenglisch (FEN)	2 SWS	
	Software-Entwicklungsprojekt (SEP)		
	Teamentwicklungs-Workshop (TEW)	2 SWS	
5	Praktisches Studiensemester (PS)	2 SWS	30 ECTS
	Kolloquium zum Praktischen Studiensemester (KPS)		
	Praktisches Studiensemester (PSS)		
	Überfachliche Kompetenzen (UK)	2 SWS	
6/7	3D-Modellierung und Spieleentwicklung (3MS)	4 SWS	5 ECTS
6/7	Anwendungscontainer und Docker (ACD)	4 SWS	5 ECTS
6/7	Automatentheorie und formale Sprachen (AFS)	4 SWS	5 ECTS
6/7	Agile Softwareentwicklung (AGI)	4 SWS	5 ECTS
6/7	Algorithmen für moderne Rechnerarchitekturen (ALR)	4 SWS	5 ECTS
6/7	Angular und Node.js (ANO)	4 SWS	5 ECTS
6/7	Angewandte Projektarbeit: Visualisierung (APV)	4 SWS	5 ECTS
6/7	Bachelorarbeit (BA)		12 ECTS
6/7	Big Data Engineering and Analysis (BDEA)	4 SWS	5 ECTS
6/7	Bioinformatik (BIM)	4 SWS	5 ECTS
6/7	Bildverarbeitung (BIV)	4 SWS	5 ECTS
6/7	Campusmanagement als Anwendungskontext für Webanwendungen (CAW)	4 SWS	5 ECTS
6/7	Challenge-Based Making (CBM)	4 SWS	5 ECTS
6/7	Cross-Plattform-Development mit dem Flutter-Framework (CPD)	4 SWS	5 ECTS
6/7	Cybersicherheit in der Prozessindustrie (CPI)	4 SWS	5 ECTS
6/7	Cyber-Security-Consulting (CSC)	4 SWS	5 ECTS
6/7	Computer Vision (CVIS)	4 SWS	5 ECTS
6/7	Digitale Forensik (DIF)	4 SWS	5 ECTS
6/7	Digitale Transformation (DTF)		5 ECTS

6/7	Entwurfsmuster für funktionale Programmierung (EFP)	4 SWS	5 ECTS
6/7	Ethik, Recht und Datenschutz (ERD)	4 SWS	5 ECTS
6/7	Game Engineering (GAE)	4 SWS	5 ECTS
6/7	Global Digital Innovation Program (GDIP)	4 SWS	5 ECTS
6/7	Grundlagen der Datenvisualisierung (GDV)	4 SWS	5 ECTS
6/7	Grundlagen Neuronaler Netze (GNN)	4 SWS	5 ECTS
6/7	Geschäftsprozessmanagement (GPM)	4 SWS	5 ECTS
6/7	Graphentheorie (GRA)	4 SWS	5 ECTS
6/7	Interaction Design (IAD)	4 SWS	5 ECTS
6/7	Image Generation and Rendering (IGR)	4 SWS	5 ECTS
6/7	iPhone App Development mit SwiftUI (IOS)	4 SWS	5 ECTS
6/7	Internet of Things (IOT)	4 SWS	5 ECTS
6/7	International Product Development Project (IPDP)	4 SWS	10 ECTS
6/7	Kodierungstheorie (KDT)	4 SWS	5 ECTS
6/7	Künstliche Intelligenz für autonome Systeme (KIS)	4 SWS	5 ECTS
6/7	Kommunikationssysteme (KOS)	4 SWS	5 ECTS
6/7	Kommerzialisierung von Open-Source-Software (KOSS)	4 SWS	5 ECTS
6/7	Kryptographische Verfahren (KRV)	4 SWS	5 ECTS
6/7	Large-Scale (Software) Development (LSD)	4 SWS	5 ECTS
6/7	Mathematische Biologie (MBI)	4 SWS	5 ECTS
6/7	Microcomputing and Embedded Development (MCP)	4 SWS	5 ECTS
6/7	Mobile und drahtlose Informationsverarbeitung (MDI)	4 SWS	5 ECTS
6/7	Maschinelles Lernen (MLE)	4 SWS	5 ECTS
6/7	Mobile Programmierung mit Fuchsia (MOP)	4 SWS	5 ECTS
6/7	Natural Language Processing (NLP)	4 SWS	5 ECTS
6/7	Einführung in die Netzwerkforensik (NWF)	4 SWS	5 ECTS
6/7	Netzwerksicherheit (NWS)	4 SWS	5 ECTS
6/7	Penetration Testing (PET)	4 SWS	5 ECTS
6/7	Prototype It Yourself (PIY)	4 SWS	5 ECTS
6/7	Reverse Engineering (REE)	4 SWS	5 ECTS
6/7	Robotik (ROB)	4 SWS	5 ECTS
6/7	Skalierbare Systemarchitekturen für IoT und Cloud-Computing (SAC)	4 SWS	5 ECTS
6/7	Scientific Computing (SCO)	4 SWS	5 ECTS
6/7	Suchmaschinenoptimierung und Social Media Marketing (SEO)	4 SWS	5 ECTS
6/7	Software-Ergonomie und Usability (SEU)	4 SWS	5 ECTS
6/7	Sichere Internet-Dienste (SID)	4 SWS	5 ECTS
6/7	Sichere Java Entwicklung (SJE)	4 SWS	5 ECTS
6/7	Security Management (SMA)	4 SWS	5 ECTS
6/7	Softwareentwicklung für Medizinprodukte (SMP)	4 SWS	5 ECTS
6/7	SQUAD (SQUAD)	8 SWS	10 ECTS
6/7	Sichere Softwarentwicklung (SSE)	4 SWS	5 ECTS
6/7	Studienarbeit (STA)		10 ECTS
6/7	Tutorium (TUT)	2 SWS	3 ECTS
6/7	Urban Hacking (UHA)	4 SWS	5 ECTS
6/7	UX Research und Design (UXD)	4 SWS	5 ECTS
6/7	Verteilte Systeme (VS)	4 SWS	5 ECTS
6/7	Webarchitekturen und -frameworks (WAF)	4 SWS	5 ECTS
6/7	Wissenschaftliches Arbeiten (WIA)	4 SWS	5 ECTS

Pflichtmodule

Ausgewählte Probleme des Software Engineerings (APS)

Name	Ausgewählte Probleme des Software Engineerings (APS)
Kürzel	APS
Semester	4
Unterrichtssprache	Deutsch
Kreditpunkte	4 ECTS
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	IB

Teil des Moduls

- Projekte in der Informatik (PI-IB)

Studienleistung

Keine

Inhalte

- Techniken und Methoden, die im Software-Entwicklungsprojekt benötigt werden; die Inhalte werden bei der Vorbereitung des Projekts festgelegt

Semesterwochenstunden

Vorlesung	1 SWS
Labor	2 SWS
Summe	**3 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	15
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	15
Selbständiges Bearbeiten der Übungen	60
Summe	**120**

Dozentinnen / Dozenten

- Prof. Dr. Peter Kaiser

- Prof. Dr. Sven Klaus
- Prof. Dr. Peter Knauber
- Prof. Dr. Wolfgang Schramm
- externe Dozenten des Projektpartners

Literatur

- abhängig vom Inhalt

Bachelorarbeit (BA)

Name	Bachelorarbeit (BA)
Kürzel	BA
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	12 ECTS
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Mündliche Prüfung (M)
Prüfungsleistung	Bachelorarbeit (BA)

Empfohlene Vorkenntnisse

- Voraussetzungen für die Ausgabe der Bachelorarbeit sind der erfolgreiche Nachweis der Studien- und Prüfungsleistungen der ersten vier Studiensemester und die Anerkennung des praktischen Studiensemesters

Inhalte

- Verständnis und Analyse eines gestellten Problems, einer Aufgabenstellung
- Ermittlung des Stands der Technik
- Entwicklung eines Konzeptes zur Lösung des Problems
- Durchführung und Validierung der Lösung
- Selbstständiges wissenschaftliches Arbeiten
- Erkennen sich ergebender zukünftiger Ansätze
- Verfassen einer wissenschaftlichen Abschlussarbeit
- Ergebnispräsentation

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- innerhalb einer vorgesehenen Frist eine wissenschaftliche Fragestellung aus dem gewählten Fachgebiet der Informatik selbstständig zu recherchieren und nach wissenschaftlichen Methoden zu bearbeiten,
- einen wissenschaftlichen Text größeren Umfangs zu strukturieren und zu schreiben und
- im Kolloquium zur Bachelorarbeit die Ergebnisse sachgerecht darzustellen und sich in einer fachlichen Diskussion zu behaupten.

Arbeitsaufwand (work load)

Projektarbeit	360
Summe	**360**

Dozentinnen / Dozenten

- alle Dozenten der Fakultät für Informatik

Literatur

- Abhängig vom Inhalt der Arbeit

Datenmanagement (DM)

Name	Datenmanagement (DM)
Kürzel	DM
Semester	3
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Software Engineering 1 (SE1)

Inhalte

- Relationale Datenbanken
- Datenmodelle (z. B. E/R-Diagramme) und Datenmodellierung
- Relationenalgebra
- Datenbankentwurf
- Structured Query Language (SQL) und deren Grundlagen
- NoSQL und BigData
- Anbindung von Datenbanken an moderne (web-basierte) Anwendungen über JDBC und JSON

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Prinzipien von Datenmodellen, Datenbanksprachen und Datenbankmanagementsystemen zu benennen,
- abzuwägen, ob ein NoSQL oder relationaler Ansatz für eine Datenmodellierung anzuwenden ist,
- vorgegebene Szenarien unter Anwendung der Datenmodellierung in einen Datenbankentwurf zu übertragen,
- die verschiedenen Paradigmen der Anfragesprachen anzuwenden und
- Datenbankanwendungen aus Hochsprachen heraus zu programmieren.

Semesterwochenstunden

Vorlesung	3 SWS
Labor	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Präsenzübungen und Testate	15
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Markus Gumbel
- Prof. Dr. Oliver Hummel
- Prof. Dr. Sven Klaus

Literatur

- L. Piepmeyer: Grundkurs Datenbanksysteme. Von den Konzepten bis zur Anwendungsentwicklung. Hanser; 2011
- G. Saake, K.U. Sattler, A. Heuer: Datenbanken – Konzepte und Sprachen: MITP-Verlag; 3. Aufl. 2007; ISBN 978-3-8266-1644-8
- A. Kemper, A. Eickler: Datenbanksysteme – eine Einführung, Oldenbourg Verlag; 6. Aufl. 2006; ISBN 978-3486576900
- P. Kleinschmidt, Ch. Rank: Relationale DB-Systeme – Eine praktische Einführung: Springer; 3. Aufl. 2005; ISBN 978-3540224969

Einführung in die Informatik (EI)

Name	Einführung in die Informatik (EI)
Kürzel	EI
Semester	1
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Miriam Föller-Nord
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Überblick über das Gebiet der Informatik (Aufgaben, Teilbereiche, Tätigkeiten. . .)
- Geschichte der Informatik
- Grundlagen zu Rechnerarchitekturen (Komponenten des Rechners: RAM, ROM, CPU, . . .)
- Zahlensysteme und Kodierung
- Betriebssysteme (Einführung), Kommandozeile (Shell), Graphische Oberflächen
- Rechnernetze und Internet (Einführung)
- Prinzipien IT-Sicherheit (Einführung)
- Aktuelle und zukünftigen Herausforderungen der IT (Mobile Systeme, IOT, Cloud Computing, Big Data Analysis, . . .)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die wichtigsten Begriffe der Informatik wiederzugeben und einzuordnen
- die Methoden der Informatik zu nennen
- informatikspezifisches Denken auf einfache gegebene Probleme anzuwenden

Semesterwochenstunden

Vorlesung	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Alle Dozenten der Fakultät für Informatik

Literatur

- Grundlagen der Informatik, Herold, Lurz, Wohlrab, 2., aktualisierte Auflage 2012, Pearson
- Einführung in die Informatik, Gumm, Sommer, Oldenburg-Verlag
- Computerarchitektur, Tanenbaum, Pearson-Studium

Fachenglisch (FEN)

Name	Fachenglisch (FEN)
Kürzel	FEN
Semester	4
Unterrichtssprache	Deutsch und Englisch
Kreditpunkte	3 ECTS
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Teil des Moduls

- Cyber Security Projekt (CP)
- Medizinisches Softwareprojekt (MP)
- Softwareprojekt (SP)
- Unternehmensinformatikprojekt (UP)

Studienleistung

Keine

Inhalte

- English for meetings and negotiation
- Business English
- Writing technical reports and project documentation in English
- Giving presentations and presenting project results in English
- Small talk in English

Semesterwochenstunden

Labor	2 SWS
Summe	**2 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	60
Summe	**90**

Dozentinnen / Dozenten

- John Clear
- externe Dozenten

Literatur

- Skript

Kolloquium zum Praktischen Studiensemester (KPS)

Name	Kolloquium zum Praktischen Studiensemester (KPS)
Kürzel	KPS
Semester	5
Unterrichtssprache	Deutsch
Kreditpunkte	2 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Sandro Leuchter
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Teil des Moduls

- Praktisches Studiensemester (PS)

Studienleistung

Referat (R)

Inhalte

- Die Studierenden stellen die Arbeit vor, die sie in ihrem Projekt durchgeführt haben.

Arbeitsaufwand (work load)

Präsenzstudium	4
Eigenstudium zur Vor- und Nachbereitung	56
Summe	**60**

Dozentinnen / Dozenten

- alle Dozentinnen und Dozenten der Fakultät für Informatik

Literatur

- Abhängig vom Projekt

Mathematik für die Informatik 1 (MA1)

Name	Mathematik für die Informatik 1 (MA1)
Kürzel	MA1
Semester	1
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Elena Fimmel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Mengenlehre
- Relationen
- Funktionen
- Aussagenlogik
- Beweismethoden
- Grundlagen der Graphentheorie

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- mathematische Symbolik zu verstehen und zu verwenden
- das mathematische Argumentieren zu verstehen und anwenden zu können
- einfache mathematische Fragestellungen selbständig zu bearbeiten

Semesterwochenstunden

Vorlesung	3 SWS
Übung	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Eigenstudium zur Vor- und Nachbereitung	60
Selbständiges Bearbeiten der Übungen	15
Prüfungsvorbereitung	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Elena Fimmel
- Prof. Dr. Miriam Föller-Nord
- Prof. Dr. Lutz Strüngmann
- Dr. Yordan Todorov

Literatur

- A. Beutelspacher, M.-A. Zschiegner; Diskrete Mathematik für Einsteiger; Vieweg 2002
- U. Knauer; Diskrete Strukturen – kurz gefasst; Spektrum Verlag 2001
- P. Hartmann, Mathematik für Informatiker: Ein praxisbezogenes Lehrbuch, Vieweg Teubner, 2012

Mathematik für die Informatik 2 (MA2)

Name	Mathematik für die Informatik 2 (MA2)
Kürzel	MA2
Semester	2
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Lutz Strüngmann
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Mathematik für die Informatik 1 (MA1)

Inhalte

- Zahlentheorie (Primzahlen, Primzahlfaktorisierung, Eulersche Phi-Funktion, Stellenwert-systeme, Zahlensysteme mit beliebigen Basen, duale Basis, Zahlendarstellung im Computer)
- Mathematische Grundlagen der Kryptographie (erweiterter Euklidischer Algorithmus, modulare Arithmetik, Restklassenringe, Invertierbarkeit bzgl. der Multiplika-tion in Restklassenringen, Satz von Fermat-Euler)
- Grundlagen der linearen Algebra und Anwendungen in der Informatik (Lineare Gleichungssysteme, Gauß-Algorithmus, der Rn als Vektorraum, Vektorrechnung, Skalarprodukt, Kreuzprodukt, Geraden, Ebenen, Matrizen, Drehmatrizen, Spiegelungen, symmetrische Matrizen, inverse Matrix, Determinanten, Eigenwerte)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- mathematische Denkweisen und Beweistechniken zu verstehen
- mathematische Problemlösungstrategien auf Probleme der Informatik anzuwenden
- elementare kryptografische Fragestellungen zu verstehen und zu analysieren
- Datenfragestellungen einzuordnen
- Methoden der linearen Algebra anzuwenden

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Eigenstudium zur Vor- und Nachbereitung	60
Selbständiges Bearbeiten der Übungen	15
Prüfungsvorbereitung	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Elena Fimmel
- Prof. Dr. Miriam Föller-Nord
- Prof. Dr. Lutz Strüngmann
- Dr. Yordan Todorov
- Prof. Dr. Ivo Wolf

Literatur

- Howard Anton: Lineare Algebra (Einführung, Grundkurs, Übungen)
- Matthias Plaue , Mike Scherfner: Mathematik für das Bachelorstudium I
- Gilbert Strang: Lineare Algebra
- Günter Gramlich: Lineare Algebra
- Günter Gramlich: Anwendungen der Linearen Algebra
- Fetzer, Fränkel: Mathematik, Band 1
- Peter Stingl: Mathematik für Fachhochschulen, Technik und Informatik
- Lothar Papula: Mathematik für Ingenieure und Naturwissenschaftler, Band 2
- Albrecht Beutelspacher: Lineare Algebra

Mathematik für die Informatik 3 (MA3)

Name	Mathematik für die Informatik 3 (MA3)
Kürzel	MA3
Semester	3
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Elena Fimmel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Mathematik für die Informatik 1 (MA1)
- Mathematik für die Informatik 2 (MA2)

Inhalte

- Kombinatorik
- klassische Wahrscheinlichkeit, Zufallsexperiment
- Ereignisraum, Elementarereignis, Ereignis, Axiome der Wahrscheinlichkeitstheorie, Rechenregeln, Zusammenhang mit der klassischen Definition
- Bedingte Wahrscheinlichkeit, Satz von der totalen Wahrscheinlichkeit, Satz von Bayes
- Zufallsgrößen, Wahrscheinlichkeitsfunktionen, Verteilungsfunktionen,Erwartungswert und Varianz, Standardabweichung, Rechenregeln
- Zentraler Grenzwertsatz,Grenzwertsatz von Moivre-Laplace,
- Kovarianz und Korrelation, Rechenregeln
- Grundbegriffe der Statistik
- Parameterschätzungen
- Regression
- Maximum-Likelihood-Methode,
- Konfidenzintervalle
- Hypothesentests: Student's- und Chi-Quadrat-Verteilung, ROCKurven

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Probleme der Informatik mit Hilfe der Graphentheorie zu lösen
- Statistische Verfahren auf gegebene Probleme anzuwenden

Semesterwochenstunden

Vorlesung	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Elena Fimmel
- Prof. Dr. Miriam Föller-Nord
- Prof. Dr. Lutz Strüngmann
- Dr. Yordan Todorov
- Prof. Dr. Ivo Wolf

Literatur

- Lothar Papula: Mathematik für Ingeneure und Naturwissenschaftler, Band 3, Viewegs Fachbücher für Technik, 1999
- P.Hartmann: Mathematik für Informatiker: Ein praxisbezogenes Lehrbuch, Vieweg Teubner, 2012

Nebenfach - Grundlagen der BWL (NF)

Name	Nebenfach - Grundlagen der BWL (NF)
Kürzel	NF
Semester	1
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Gabriele Roth-Dietrich
Dauer	1 Semester
Studiengänge	IB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Theoriethemen
- Übungen zur Konzeption eines mittelständischen Unternehmens (KMU) als Teil eines Konzerns
- Übungen zum IT-Transfer: Abbildung von Geschäftsprozessen von KMUs in einer Business Suite (z. B. SAP Business ByDesign)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- den Aufbau und die Funktionsweise von Unternehmen zu beschreiben,
- Steuerungswerkzeuge der Unternehmensführung in verschiedenen Unternehmenbereichen zu unterscheiden und zu bewerten,
- grundlegende betriebswirtschaftliche Prozesse darzustellen und
- betriebswirtschaftliche Software für KMUs einzusetzen.

Semesterwochenstunden

Vorlesung	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Präsenzübungen und Testate	15
Eigenstudium zur Vor- und Nachbereitung	75
Summe	**150**

Dozentinnen / Dozenten

- Thomas Borckholder

Literatur

- Amely, Tobias und Thomas Krickhahn, BWL für Dummies, Weinheim 2009
- Daum, Andreas u.a., BWL für Ingenieure und Ingenieurinnen, Wiesbaden 2010
- Griga, Michael und Raymund Krauleidis, Buchführung und Bilanzierung für Dummies, 3. Aufl., Weinheim 2012
- Junge, Philip, BWL für Ingenieure, Weinheim 2010
- Oehlrich, Marcus, Betriebswirtschaftslehre, 2. Aufl., München 2010
- Pepels, Werner (Hrsg.), BWL im Nebenfach, 2. Aufl., Herne 2010
- Voss, Rödiger, BWL kompakt, 6. Aufl., Rinteln 2012

Projekte in der Informatik (PI-IB)

Name	Projekte in der Informatik (PI-IB)
Kürzel	PI-IB
Häufigkeit	Jedes Semester
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	IB

Das Modul besteht aus den Veranstaltungen

- Ausgewählte Probleme des Software Engineerings (APS)
- Projektmanagement (PM)
- Virtualisierung (VIR)

Prüfungsleistung

Referat (R)

Empfohlene Vorkenntnisse

- Abgeschlossenes Grundstudium
- Software Engineering 2 (SE2) (Studienleistung)
- Data Management (DM) (Studienleistung)
- Webbasierte Systeme (WEB) (Studienleistung)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- ein kleines Projekt selbstständig zu managen,
- eine projektspezifische virtuelle Rechnerinfrastruktur zu entwerfen, zu implementieren und zu warten,
- fachliche und Anwendungsdomänen-spezifische Techiken und Methoden aus dem Projektumfeld anzuwenden und damit Software-Lösungen zu entwickeln

Projektmanagement (PM)

Name	Projektmanagement (PM)
Kürzel	PM
Semester	4
Unterrichtssprache	Deutsch
Kreditpunkte	3 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Teil des Moduls

- Projekte in der Cyber Security (PCS)
- Projekte in der Informatik (PI-IB)
- Projekte in der Medizinischen Informatik (PMI)
- Projekte in der Wirtschaftsinformatik (PWI)

Studienleistung

Keine

Inhalte

- Was ist ein Projekt?
- Aufgaben- vs. Terminplanung
- Werkzeuge für Projektmanagement
- Qualitätsmanagement und Projektkontrolle
- Rollen im Projekt
- Klassisches Projektmanagement
- Agiles Projektmanagement (z. B. Scrum und KANBAN)

Semesterwochenstunden

Vorlesung	2 SWS
Labor	1 SWS
Summe	**3 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	15
Eigenstudium zur Vor- und Nachbereitung	45
Summe	**90**

Dozentinnen / Dozenten

- Prof. Dr. Rainer Gerten
- Prof. Dr. Michael Gröschel
- Prof. Dr. Peter Kaiser
- Prof. Dr. Sachar Paulus
- Prof. Dr. Gabriele Roth-Dietrich
- Prof. Thomas Smits

Literatur

- deMarco, Tom: Der Termin. Ein Roman über Projektmanagement. Hanser 1998.
- Preußig, Jörg: Agiles Projektmanagement. Haufe-Lexware, Freiburg, 2015.

Parallele Programmierung (PP)

Name	Parallele Programmierung (PP)
Kürzel	PP
Semester	3
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Sandro Leuchter
Dauer	1 Semester
Studiengänge	IB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Software Engineering 1 (SE1)

Inhalte

- Parallelität und Nebenläufigkeit, Prozesse, Threads, Co-Routinen/Fibers
- Thread-Konzepte in Java
- kritische Abschnitte aufgrund konkurrierenden Zugriffs auf Daten in Java, gegenseitiger Ausschluss mit synchronized (intrinsic lock)
- Deadlocks, Signalisieren (wait/notify) und Bedingungsvariablen (await/signal), Lebenszyklus von Threads, Interrupts
- Thread Pools, Callable, Future und FutureTask
- Java Speichermodell bei Nebenläufigkeit, Memory Barriers
- Locks (ReentrantLock, ReadWriteLock, StampedLock) und Semaphore
- synchronisierte, unmodifiable und concurrent Collections in Java
- atomare Variablen und ihre Verwendung
- weitere Architekturen und Frameworks für nebenläufige und parallele Programmierung in Java und ggf. anderen Programmiersprachen: z. B. Actor Model, Communicating Sequential Processes, Fork-Join, Map-Reduce, Parallel Streams, Software Transactional Memory, RxJava, Datenparallelismus: GPGPU (OpenCL oder CUDA)
- Performanzbetrachtungen
- Entwurfsmuster (für nebenläufige und parallele Programme)
- Praktische Übungen zur Implementierung asynchroner, nebenläufiger und paralleler Algorithmen mit Java und ggf. anderen Programmiersprachen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Probleme und Chancen der parallelen Programmierung zu nennen,
- Probleme in Java-Programmen zu erkennen, die aus Nebenläufigkeit resultieren,
- nebenläufige Programme zu erstellen,
- vorhandene Programme bezüglich ihres Verhalten bei Nebenläufigkeit zu analysieren

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	50
Selbständiges Bearbeiten der Übungen	40
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr.-Ing. Sandro Leuchter

Literatur

- Jamie Allen (2013): Effective Akka. Patterns and Best Practices. O'Reilly.
- Paul Butcher (2014): Seven Concurrency Models in Seven Weeks. When Threads Unravel. The Pragmatic Programmers.
- Jörg Hettel und Manh Tien Tran (2016): Nebenläufige Programmierung mit Java. Konzepte und Programmiermodelle für Multicore-Systeme. dpunkt Verlag.
- Andrew S. Tanenbaum und Herbert Bos (2016): Moderne Betriebssysteme. 4., aktualisierte Auflage. Pearson Studium.
- Vaughn Vernon (2016): Reactive Messaging Patterns With The Actor Model. Applications and Integration in Scala and Akka. Pearson Education.

Programmierung 1 (PR1)

Name	Programmierung 1 (PR1)
Kürzel	PR1
Semester	1
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	10 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Benutzung einer integrierten Entwicklungsumgebung (IDE), inklusive Debugger, Versionsverwaltung, automatisierte Tests
- Konstrukte der imperativen Programmierung: Zuweisung, Bedingung, Schleife, Array, (statische) Methode und Parameter, Rekursion, RuntimeException
- Speicherverwaltung (Stack und Heap)
- Konstrukte der objektorientierten Programmierung: Klasse, Objekt, Attribut, (dynamisch gebundene) Methode
- Interfaces und ihre Implementierung
- Vererbung, Polymorphismus

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- grundlegende Entwicklungswerkzeuge (Editor, Compiler, Interpreter, integrierten Umgebung, Versionsverwaltung, Shell) anzuwenden,
- Grundbegriffe der objektorientierten Programmierung zu benennen und zu erklären,
- vorgegebene Algorithmen in objektorientierte Java-Programme zu übertragen,
- in kleinen Teams zu arbeiten und ihre Arbeit vorzustellen,
- einfache Algorithmen zu verstehen, selbst zu entwerfen und (in Java) zu implementieren.

Semesterwochenstunden

Vorlesung	4 SWS
Labor	4 SWS
Summe	**8 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Präsenzübungen und Testate	60
Eigenstudium zur Vor- und Nachbereitung	80
Selbständiges Bearbeiten der Übungen	80
Prüfungsvorbereitung	20
Summe	**300**

Dozentinnen / Dozenten

- Prof. Dr. Jörn Fischer
- Prof. Dr. Oliver Hummel
- Prof. Dr. Sven Klaus
- Prof. Dr. Peter Knauber
- Prof. Dr.-Ing. Sandro Leuchter
- Prof. Dr. Sachar Paulus
- Prof. Dr. Wolfgang Schramm
- Prof. Thomas Smits
- Prof. Dr. Thomas Specht
- Prof. Dr. Frank

Literatur

- Christian Ullenboom, Java ist auch eine Insel, 12. Auflage, 2016
- Michael Kofler, Java: Der kompakte Grundkurs mit Aufgaben und Lösungen, 1. Auflage, 2014

Programmierung 2 (PR2)

Name	Programmierung 2 (PR2)
Kürzel	PR2
Semester	2
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	10 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Einführung in die Informatik (EI)
- Programmierung 1 (PR1)

Inhalte

- Objektorientierung / Klasse Object (Wiederholung) / Polymorphismus (Vertiefung)
- Innere Klassen und Lambdas
- Input/Output (Grundlagen)
- Exception Handling
- Grundlagen der Nebenläufigkeit (Threads)
- Generics (Grundlagen)
- Collection Framework
- Wichtige Datenstrukturen (Hash, Baum, Tries, Graphen)
- Sortieren

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- verschiedene Konzepte in Java zu beurteilen,
- alle wichtigen Konzepte von Java anzuwenden,
- nichttriviale objektorientierte Programme in Java zu entwickeln,
- in kleinen Teams zu arbeiten und ihre Arbeit vorzustellen.
- den Aufwand für Algorithmen abzuschätzen,

- unterschiedliche Algorithmen und dynamische Datenstrukturen in Hinblick auf ihre Anwendung zu beurteilen und zu implementieren

Semesterwochenstunden

Vorlesung	4 SWS
Labor	4 SWS
Summe	**8 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Präsenzübungen und Testate	60
Eigenstudium zur Vor- und Nachbereitung	100
Selbständiges Bearbeiten der Übungen	80
Summe	**300**

Dozentinnen / Dozenten

- Prof. Dr. Oliver Hummel
- Prof. Dr. Peter Kaiser
- Prof. Dr.-Ing. Sandro Leuchter
- Prof. Dr. Wolfgang Schramm
- Prof. Thomas Smits
- Prof. Dr. Markus Gumbel
- Prof. Dr. Frank Dopatka

Literatur

- Christian Ullenboom, Java ist auch eine Insel, 16. Auflage, 2022
- Michael Kofler, Java: Der kompakte Grundkurs mit Aufgaben und Lösungen, 4. Auflage, 2022
- Joshua Bloch, Effective Java: A Programming Language Guide, 2008
- Maurice Naftalin, Philip Wadler, Java Generics and Collections, 2006
- Gunter Saake und Kai-Uwe Sattler, Algorithmen und Datenstrukturen, 6. Auflage, 2021

Programmierung 3 (PR3)

Name	Programmierung 3 (PR3)
Kürzel	PR3
Semester	3
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	IB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)

Inhalte

- Multi-Paradigmen-Sprachen (z. B. Scala)
- Funkionale Sprachen (z. B. Elixir, Lisp, Clojure)
- Sprachen mit explizitem Speichermanagement (C)
- Systemnahe Sprachen (z. B. Go)
- Skriptsprachen (Python, Ruby)
- Domainspezifische Sprachen
- Weiterführende Konzepte von Java (z. B. Streams, Reflection)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- verschiedene Programmierparadigmen zu erläutern
- Programme mit einer Skript-Sprache zu erstellen
- Programme mit einer funktionalen Sprache zu erstellen
- Sprachkonstrukte zu bewerten und zu beurteilen

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Oliver Hummel
- Prof. Dr.-Ing. Sandro Leuchter
- Prof. Dr. Wolfgang Schramm
- Prof. Thomas Smits
- Prof. Dr. Jessica Steinberger

Literatur

- abhängig von den gewählten Programmiersprachen

Praktisches Studiensemester (PS)

Name	Praktisches Studiensemester (PS)
Kürzel	PS
Häufigkeit	Jedes Semester
Modulverantwortlich	Prof. Dr.-Ing. Sandro Leuchter
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Das Modul besteht aus den Veranstaltungen

- Kolloquium zum Praktischen Studiensemester (KPS)
- Praktisches Studiensemester (PSS)
- Überfachliche Kompetenzen (UK)

Prüfungsleistung

Keine

Empfohlene Vorkenntnisse

- Abgeschlossenes Grundstudium
- Softwareprojekt (SP) bzw. Medizinisches Softwareprojekt (MP) bzw. Unternehmensinformatik-Projekt (UP)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Arbeitsbedingungen und Arbeitsmethoden eines/r Informatikers/in in der Praxis zu beschreiben,
- ihr Wissen durch Mitarbeit in einem aktuellen Projekt aus dem Informatiksektor unter Anleitung eines/r erfahrenen Praktikers/in (Informatiker/in, Ingenieur/in, Naturwissenschaftler/in oder Mathematiker/in) anzuwenden,
- Schlüsselqualifikationen wie Teamfähigkeit, Kommunikationsfähigkeit, Kritikfähigkeit etc. einzusetzen und weiterzuentwickeln,
- einen technischen Bericht zu erstellen.

Praktisches Studiensemester (PSS)

Name	Praktisches Studiensemester (PSS)
Kürzel	PSS
Semester	5
Unterrichtssprache	Deutsch
Kreditpunkte	25 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Sandro Leuchter
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Teil des Moduls

- Praktisches Studiensemester (PS)

Studienleistung

Praktische Arbeit/Projektarbeit (PA)

Inhalte

- Tätigkeiten in Projekten, die im Zusammenhang mit der Ausbildung in Informatik, bzw. zur medizinischen Informatik, bzw. zur Unternehmens- und Wirtschaftsinformatik stehen.
- Möglichst selbstständiges Arbeiten innerhalb eines Projektteams.
- Die Studierenden erstellen für jede Woche eine Tätigkeitsbeschreibung.
- Am Ende des PS fertigen die Studierenden einen Projektbericht an.
- Im betrieblichen Teil des PS müssen mindestens 100 Arbeitstage Vollzeit nachgewiesen werden.

Arbeitsaufwand (work load)

100 * 7,5h = 750h

Dozentinnen / Dozenten

- alle Dozentinnen und Dozenten der Fakultät für Informatik

Literatur

- Oswald, A., Köhler, J. & Schmitt, R. (2016). Projektmanagement am Rande des Chaos. Sozialtechniken für komplexe Systeme. Berlin: Springer-Verlag.

Software Engineering 1 (SE1)

Name	Software Engineering 1 (SE1)
Kürzel	SE1
Semester	2
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Einführung in die Informatik (EI) (Studienleistung)
- Programmierung 1 (PR1) (Studienleistung)

Inhalte

- Einleitung: V-Modell als Strukturvorlage für SE1 und SE2
- Begriffe der Objektorientierung: Wiederholung, Prüfung, Ergänzung
- Objektorientierte Analyse
- Objektorientiertes Design
- Komponententests
- Integrationstests
- Zusammenarbeit in Teams von 8 bis 12 Personen, Managen dieser Teams
- Konfigurationsmanagement
- Entwickeln einer GUI mit Observer-Pattern-Anwendung
- Einführung in das Projektmanagement

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- das Zusammenspiel von objektorientierter Analyse und Design bis zur Programmierung zu erklären,
- den Zusammenhang zwischen statischer und dynamischer Modellierung zu beschreiben und mittels verschiedener UML-Diagrammtypen darzustellen,
- diese objektorientierten Konzepte in die Sprache Java zu übertragen und

- Beispiele für typische Probleme, die beim arbeitsteiligen Entwickeln von Software auftreten können, anzuführen.
- zu verstehen, was mit Tests erreicht werden kann, den Zusammenhang von BB-, WB- und GB-Tests zu erklären und Komponententests mit JUnit zu implementieren.

Semesterwochenstunden

Vorlesung	3 SWS
Labor	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Präsenzübungen und Testate	15
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Oliver Hummel
- Prof. Dr. Peter Kaiser
- Prof. Dr. Sven Klaus
- Prof. Dr. Peter Knauber
- Prof. Kirstin Kohler
- Prof. Dr. Thomas Specht
- Prof. Dr. Wolfgang Schramm

Literatur

- B. McLaughlin, G. Pollice: Head First Object-Oriented Analysis and Design: A Brain Friendly Guide to OOA&D. O'Reilly; 2006
- F. Witte: Testmanagement und Softwaretest: Theoretische Grundlagen und praktische Umsetzung. Springer; 2015
- E. Freeman, E. Robson: Head First Design Patterns. O'Reilly; 2014
- I. Sommerville: Software Engineering. Pearson; 2012

Software Engineering 2 (SE2)

Name	Software Engineering 2 (SE2)
Kürzel	SE2
Semester	3
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Erfahrungen im objektorientierten Entwurf und Teamarbeit an einem (kleinen) Softwareentwicklungsprojekt (z. B. erfolgreiche Studienleistung in Software Engineering 1 (SE1))
- Fundierte Kenntnisse in Java (z. B. erfolgreiche Studienleistung in Programmieren 1 (PR1))

Inhalte

- Ethik der Informatik
- Anforderungsanalyse (Personas, Use Cases, User Stories, Szenarien, UI-Prototypen)
- Usability (Dialogprinzipien, UI-Elemente)
- Security-Grundlagen
- Architektur-Grundlagen (Arc42)
- Qualitätssicherung und Testmanagement
- Softwareentwicklungsprozesse (Überblick, Scrum)
- Durchgängiges Beispiel (Anforderungsanalyse, Architektur, QS) aufbauend auf SE1-Beispiel

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Bedeutung der verschiedenen Phasen der Software-Entwicklung zu beschreiben, deren Anordnung in Entwicklungsprozessen zu benennen,
- den Nutzen der Mensch-Maschine-Interaktion und der Security für ein Projekt zu erkennen und Lösungsansätze zu entwickeln,
- die Herausforderungen der Anforderungsanalyse, des Entwurfs und der Qualitätssicherung zu erkennen und Lösungsansätze zu entwickeln,

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Oliver Hummel
- Prof. Dr. Peter Kaiser
- Prof. Dr. Peter Knauber
- Prof. Kirstin Kohler
- Prof. Dr. Till Nagel
- Prof. Dr. Wolfgang Schramm

Literatur

- Dahm, M.: Grundlagen der Mensch-Computer-Interaktion. München, Pearson, 2005.
- Richter, M. & Flückiger, M.: Usability Engineering kompakt, Spektrum Akademischer Verlag, 2007
- The Encyclopedia of Human-Computer Interaction, http://www.interaction-design.org/encyclopedia/
- Sommerville, Ian: Software Engineering. Pearson Studium 8. Auflage, 2007.
- Ludewig, Jochen; Jan Lichter: Software Engineering. dpunkt-Verlag, 2007.
- Pohl, Klaus; Chris Rupp: Basiswissen Requirements Engineering. dpunkt-Verlag, 2011.
- Starke, Gernot: Effektive Software-Architekturen. Hanser, 2011.
- Erich Gamma, R. Helm, R. Johnson, J. Vlissides: Entwurfsmuster - Elemente wiederverwendbarer objektorientierter Software, Addison-Wesley, 1995.
- Spillner, Andreas; Tilo Linz: Basiswissen Softwaretest. dpunkt-Verlag 2012.
- Tom DeMarco: Der Termin, Carl Hanser Verlag, 1998.
- SCRUM Guide http://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum_Guide.pdf

Software-Entwicklungsprojekt (SEP)

Name	Software-Entwicklungsprojekt (SEP)
Kürzel	SEP
Semester	4
Unterrichtssprache	Deutsch
Kreditpunkte	15 ECTS
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	IB

Teil des Moduls

- Softwareprojekt (SP)

Studienleistung

Keine

Inhalte

- In kleinen Teams von ca. 5 Personen arbeiten die Studierenden selbständig an der Realisierung eines Produktes
- Es werden alle Phasen von der Produktidee bis zur Einführung beim Kunden durchgeführt
- Es gibt einen Hochschul-externen Kunden, der die Produkt- und ggf. auch Projektanforderungen stellt
- Die Entwicklung läuft nach Scrum ab
- Das technische Wissen stammt aus Vorlesungen der ersten drei Semester, in Workshops werden soziale Kompetenzen (Teamentwicklungs-Workshop) vermittelt.

Arbeitsaufwand (work load)

Projektarbeit	450
Summe	**450**

Dozentinnen / Dozenten

- Prof. Dr. Peter Kaiser
- Prof. Dr. Sven Klaus
- Prof. Dr. Peter Knauber
- Prof. Kirstin Kohler
- Prof. Thomas Smits
- Prof. Dr. Wolfgang Schramm

Literatur

- Preußig, Jörg: Agiles Projektmanagement. Haufe-Lexware, Freiburg, 2015.
- Schatten, Alexander; Markus Demolsky; Dietmar Winkler; Stefan Biffl; Erik Gostischa-Franta; Thomas Östreicher: Best Practice Software Engineering. Spektrum Akademischer Verlag, 2010.

Softwareprojekt (SP)

Name	Softwareprojekt (SP)
Kürzel	SP
Häufigkeit	Jedes Semester
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	IB

Das Modul besteht aus den Veranstaltungen

- Fachenglisch (FEN)
- Software-Entwicklungsprojekt (SEP)
- Teamentwicklungs-Workshop (TEW)

Prüfungsleistung

Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Abgeschlossenes Grundstudium
- Software Engineering 2 (SE2) (Studienleistung)
- Datenmanagement (DM) (Studienleistung)
- Webbasierte Systeme (WEB) (Studienleistung)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- ein Software-Entwicklungsprojekt von ersten Anforderungen bis zur Produkteinführung aus der Sicht von Projektleiter und Entwickler zu beschreiben,
- mit unklaren Anforderungen und Zeitdruck konstruktiv umzugehen,
- ein kleines Projekt selbstständig durchzuführen,
- angemessene Methoden und Werkzeuge auszuwählen und anzuwenden,
- den Nutzen der Projektplanung und der Entwicklungsdokumentation zu kennen.
- englische Entwicklungsdokumentation zu verstehen und – für interne Zwecke – zu schreiben,
- an englisch geführten Fachdiskussionen teilzunehmen

Technische Informatik 1 (TEI1)

Name	Technische Informatik 1 (TEI1)
Kürzel	TEI1
Semester	1
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Ihme
Dauer	1 Semester
Studiengänge	IB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Schaltungstechnische Grundlagen
- Kombinatorische Schaltnetze (Gatter, einfache Schaltungen mit Gattern)
- Boolesche Schaltalgebra (Gesetze, Minimierung), Karnaugh-Pläne
- Sequenzielle Schaltungen (Flip-Flops, Anwendungen mit Flip-Flops)
- Computerarithmetik (Zahlendarstellungen, Grundrechenarten, Realisierung in Hardware)
- Codes (Zeichencodes, fehlererkennende und fehlerkorrigierende Codes, optimale Codes)
- Adressierung und Befehlsfolgen
- Struktur der CPU (Adress- und Datenpfad, mikroprogammierte/verdrahtete Kontrolleinheit)
- RISC-Prozessoren (Eigenschaften, Registerfenster, Pipelining)
- Speicher (verschiedene Ausführungen)
- Speicherarchitektur (Lokalitätsprinzip, Cache, virtueller Speicher)
- Parallelrechner (Kommunikationsmodelle, Verbindungsnetzwerke, Metriken)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- kombinatorische Schaltnetze mit Hilfe von Wahrheitstabellen zu beschreiben und Boolescher Algebra zu minimieren und Schaltnetze zu synthetisieren

- zu erläutern, wie grundlegende arithmetische Operationen realisiert und bezüglich Realisierungsaufwand und Rechengeschwindigkeit zu unterscheiden
- die Funktionsweise von Registern, Speicher und Zählern zu beschreiben,
- die rechnerinterne Darstellung von Zahlen und Zeichen zu erläutern
- grundlegende Methoden der fehlererkennenden und -korrigierenden Codierung sowie optimaler Codes zu erläutern und diese in einfachen Beispielen anzuwenden
- Programmausführung auf Hardwareebene zu verstehen (CPU) und einfache Assemblerprogramme zu entwickeln
- Fortgeschrittene Rechnerkonzepte wie RISC/CISC, Speicherarchitekturen und Parallelrechner zu verstehen und resultierende Effekte bezüglich Programmausführung und Rechenzeit einzuschätzen

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jörn Fischer
- Prof. Dr. Thomas Ihme

Literatur

- Tanenbaum: „Structured Computer Organization“, Prentice Hall
- Patterson, Hennessy: „Computer Organization & Design. The Hardware/Software Interface“, Morgan Kaufmann
- Clements: „The Principles of Computer Hardware“, Oxford University Press
- Hamacher, Vranesic, Zaky: „Computer Organization“, McGraw-Hill

Technische Informatik 2 (TEI2)

Name	Technische Informatik 2 (TEI2)
Kürzel	TEI2
Semester	2
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Jörn Fischer
Dauer	1 Semester
Studiengänge	IB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Hardwarenahe C/C++ konstrukte (Shift, Bitoperationen), Zeigerarithmetik
- Begrenzte Rechengenauigkeit und numerische Probleme
- Prozessortypen (Microcontroller, Grafikprozessoren)
- Betriebssysteme
- Virtuelle Maschinen
- Profiler und Debugger

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- C/C++ Programme zu schreiben
- Entwicklungswerkzeuge Profiler und Debugger anzuwenden
- einfache numerische Probleme zu lösen
- Microcontroller und Grafikkarten zu programmieren
- den Aufbau von Betriebssystemen zu kennen
- den Aufbau von Virtuellen Maschinen zu kennen

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jörn Fischer
- Prof. Dr. Thomas Ihme

Literatur

- Taschenbuch der Informatik, Hanser Verlag, ISBN:978-3-446-42638-2, 2012

Teamentwicklungs-Workshop (TEW)

Name	Teamentwicklungs-Workshop (TEW)
Kürzel	TEW
Semester	4
Unterrichtssprache	Deutsch
Kreditpunkte	2 ECTS
Modulverantwortlich	Prof. Dr. Sven Klaus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Teil des Moduls

- Cyber Security Projekt (CP)
- Medizinisches Softwareprojekt (MP)
- Softwareprojekt (SP)
- Unternehmensinformatikprojekt (UP)

Studienleistung

Keine

Inhalte

- Es gibt einen Einführungs-, sowie i.d.R. zwei Supervisionsworkshops.
- Einführungsworkshop: Was ist ein Team? Was sind Ziele? Wie werden Sie formuliert? Wie entsteht ein Team (Phasen)? Welche Rollen gibt es im Team? Wie läuft Kommunikation ab? Wie gibt man Feedback im Team? Wie geht man mit Konflikten im Team um? Wie koordiniert man ein Team?
- Supervisionsworkshop: Besprechen aktueller Themen nach den Wünschen/Erfordernissen der Teams

Semesterwochenstunden

Workshop	2SWS
Summe	**2 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	30
Summe	**60**

Dozentinnen / Dozenten

- Marco D'Angelo

Literatur

- Cohn, Ruth C., Christina Terfurth (Hrsg.): Lebendiges Lehren und Lernen, TZI macht Schule. Stuttgart, Klett-Cotta, 2001.
- Fengler, Jörg: Feedback geben - Strategien und Übungen. Beltz, Weinheim, 1998.
- Gellert, Manfred; Claus Novak: Ein Praxisbuch für die Arbeit in und mit Teams. Limmer Verlag, Meezen, 2004.
- Gilsdorf, Rüdiger; Günther Kistner: Kooperative Abenteuerspiele, eine Praxishilfe für die Schule. Kallmeyer Verlag, Seelze-Velber, 2000.
- Haeske, Udo: Team- und Konfliktmanangement. Cornelsen, Berlin, 2002.
- Langmaak, Barbara: Einführung in die Themenzentrierte Interaktion TZI. Beltz Verlag, Weinheim, 2001.
- Langmaak, Barbara; Michael Braune-Krickau: Wie die Gruppe laufen lernt. Beltz Verlag, Weinheim, 2000.
- Prior, Michael: MiniMax- Interventionen, 15 minimale Interventionen mit maximaler Wirkung. Carl-Auer Verlag, Heidelberg, 2004.
- Schulz von Thun, Friedemann: Miteinander Reden, Band 1: Störungen und Klärungen. Rowohlth, Hamburg, 1999.
- Walter, Simon: Gabals großer Methodenkoffer - Grundlagen der Kommunikation. Gabal Verlag, Offenbach, 2004.
- Wolf, Manfred: Fachlexikon der Sozialen Arbeit. Deutscher Verein für öffentliche und private Fürsorge, Frankfurt/Main, 2002.

Theoretische Informatik (THI)

Name	Theoretische Informatik (THI)
Kürzel	THI
Semester	2
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Lutz Strüngmann
Dauer	1 Semester
Studiengänge	IB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Grundlagen der Logik (Prädikatenlogik 1.Stufe inkl. Resolutionsverfahren, Hornklausellogik, SLD-Resolution, Negation as Failure und regelbasierte Systeme)
- Formale Sprachen (Chomsky-Hierarchie, Reguläre Sprachen, Kontextfreie Sprachen)
- Automatentheorie (endliche Automaten, Kellerautomaten)
- Grundlagen der Berechenbarkeit
- Grundlagen der Komplexitätstheorie

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- den Prozess, von einer Problemstellung über strukturierte logische Konzepte zu einer automatisch durchführbaren Problemlösung zu gelangen, zu erläutern und zu verfolgen,
- ein gedanklichen Lösungskonzepts in eine vom Rechner verarbeitbare formale Darstellung umzusetzen,
- die syntaktischen Grundlagen von Programmiersprachen und ihre Verarbeitung auf dem Rechner zu beschreiben und
- analytisch und strukturiert zu denken mit dem Ziel, systematisch Problemlösungen entwickeln zu können.

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Astrid Schmücker-Schend

Literatur

- Socher, Rolf: „Theoretische Grundlagen der Informatik“, Hanser-Verlag
- Vossen, G., Witt, K.-U.: „Grundkurs Theoretische Informatik“, Vieweg+Teubner
- Hopcroft, John E., Motwani, Rajeev, Ullman, Jeffrey D.: „Einführung in Automatentheorie, Formale Sprachen und Berechenbarkeit“, Verlag: Addison Wesley / Pearson Studium
- Ertel, Wolfgang:„Grundkurs Künstliche Intelligenz : Eine praxisorientierte Einführung“, Springer Vieweg

Tutorium (TUT)

Name	Tutorium (TUT)
Kürzel	TUT
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	3 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Referat (R)

Empfohlene Vorkenntnisse

- abgeschlossenes Grundstudium

Inhalte

- Unterstützen von Übungen (Konzeption einer Übungen mit Musterlösung; Vorstellen der Übungen; aktive Betreuung der Übungen vor Ort im Labor; Testieren der Studierenden)
- Unterstützen von Vorlesungen (Konzeption einer Unterrichtseinheit mit klar abgegrenztem Thema; Vorstellen des Themas)
- Unterstützen von Projekten (Konzeption eines zum Inhalt des Projekts passenden Vortrags (15-25 Minuten); Halten des Vortrags; aktive Betreuung der Studierenden)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Lehrinhalte selbstständig zu erstellen,
- Übungs- oder Unterrichtseinheiten, abgestimmt auf den jeweiligen Kenntnisgrad der Teilnehmer, auszuarbeiten,
- Lösungen von anderen Studierenden zu bewerten und abzunehmen,
- Studierende zu unterstützen und fördern.

Semesterwochenstunden

Projekt	2 SWS
Summe	**2 SWS**

Arbeitsaufwand (work load)

Eigenstudium zur Vor- und Nachbereitung	60
Projektarbeit	30
Summe	**90**

Dozentinnen / Dozenten

- alle Dozenten der Fakultät für Informatik

Literatur

- J. Böhringer, P. Bühler, P. Schlaich - Präsentieren in Schule, Studium und Beruf: Springer; 1. Aufl. 2008; ISBN 978-3540457046

Überfachliche Kompetenzen (UK)

Name	Überfachliche Kompetenzen (UK)
Kürzel	UK
Semester	5
Unterrichtssprache	Deutsch
Kreditpunkte	3 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Teil des Moduls

- Praktisches Studiensemester (PS)

Studienleistung

Referat (R)

Inhalte

- Die Studierenden engagieren oder beschäftigen sich außerhalb ihrer Lehrveranstaltungen mit/für soziale oder andere nicht informatische Themen, die jedoch für den Informatik-Beruf relevant sind. Die Veranstaltung folgt dem Konzept des Service Learning.
- Beispiele: Teilnahme an Kompass (als Teilnehmer, als Tutor); für UK freigegebene Workshops der Summer School; Kurse des Carreer Centers der Hochschule Mannheim; Engagement außerhalb der Hochschule in sozialen Einrichtungen.

Semesterwochenstunden

Vorlesung	2 SWS
Summe	**2 SWS**

Arbeitsaufwand (work load)

Eigenstudium zur Vor- und Nachbereitung	60
Workshop	30
Summe	**90**

Dozentinnen / Dozenten

- alle Dozenten der Fakultät für Informatik

Virtualisierung (VIR)

Name	Virtualisierung (VIR)
Kürzel	VIR
Semester	4
Unterrichtssprache	Deutsch
Kreditpunkte	3 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	IB

Teil des Moduls

- Projekte in der Informatik (PI-IB)

Studienleistung

Keine

Inhalte

- Modellierung von Entwicklungs- und Qualitätssicherungsumgebungen
- Virtualisierung
- Modellierung von virtuellen Maschinen, Snapshots, Datensicherung
- Serverbetriebssysteme: Windows Server und Linux Server
- Desktopbetriebssysteme: Windows und Linux
- Planung, Installation und Administration von Server- und Desktop-Betriebssystemen in virtualisierten Umgebungen
- Planung, Modellierung und Administration von virtuellen Netzen
- Services in virtuellen Maschinen
- Web-Services: WAMP / LAMP

Semesterwochenstunden

Vorlesung	1 SWS
Übung	1 SWS
Summe	**2 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	15
Eigenstudium zur Vor- und Nachbereitung	60
Selbständiges Bearbeiten der Übungen	15
Summe	**90**

Dozentinnen / Dozenten

- Dipl.-Inf. Thorsten Gerweck

Verteilte Systeme (VS)

Name	Verteilte Systeme (VS)
Kürzel	VS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Specht
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 60 Minuten (K60)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Software Engineering 1 (SE1)
- Webbasierte Systeme (WEB)

Inhalte

- Rechnernetze und Netzwerkdienste
- Virtualisierung von Rechnern, Rechnernetzen und Speichersystemen
- Verteilte Architekturen (Client-Server, SOA, Microservices, komponentenbasiert, webbasiert, Anwendungscontainer, Cluster, Cloud, XaaS)
- Middleware
- Entfernte Methodenaufrufe
- Message Oriented Middleware

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Anforderungen an ein verteiltes Informationssystem zu analysieren
- geeignete Softwarearchitekturen für verteilte Informationssysteme auszuwählen und zu bewerten
- eine geeignete Middleware und Ablaufumgebung auszuwählen
- ein verteiltes Softwaresystem zu entwerfen, zu implementieren und in Betrieb zu nehmen

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	75
Selbständiges Bearbeiten der Übungen	30
Prüfungsvorbereitung	15
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Oliver Hummel
- Prof. Dr.-Ing. Sandro Leuchter
- Prof. Thomas Smits
- Prof. Dr. Thomas Specht

Literatur

- Günther Bengel: Grundkurs Verteilte Systeme, Vieweg, 2014
- Andrew Tanenbaum, Maarten van Steen: Verteilte Systeme, Pearson Studium, 2017
- Oracle: Java Platform, Enterprise Edition (Java EE) 8 The Java EE Tutorial, http://www.oracle.com/technetwork/java/javaee/dow
  2018

Webbasierte Systeme (WEB)

Name	Webbasierte Systeme (WEB)
Kürzel	WEB
Semester	3
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Einführung in die Informatik (EI)
- Programmierung 1 (PR1)
- Programmierung 2 (PR2)

Inhalte

- Client/Server-Modell
- XML-Grundlagen (XML als Markup, XML Schema, DTD)
- HTML/CSS Grundlagen
- JavaScript, JSON
- Web Frameworks (z. B. Rails, JSF, Django)
- Einfache Backend-Integration mit REST
- Oberflächentests (Selenium)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Anforderungen an eine Webanwendung zu analysieren,
- die Programmierparadigmen von Web-basierten Applikationen verstehen,
- geeignete Webtechnologien für die Umsetzung konkreter Anforderungen auszuwählen,
- Webanwendungen mit den Standardtechnolgien HTML, XML, CSS, JavaScript, Webserver u.ä. zu entwickeln und
- einfache Backends für Webanwendungen zu realisieren und per REST anzubinden.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Frank Dopatka
- Prof. Dr. Sachar Paulus
- Prof. Thomas Smits
- Prof. Dr. Thomas Specht

Literatur

- Helmut Vonhoegen, Einstieg in XML: Grundlagen, Praxis, Referenz, Rheinwerk Computing, 2015
- Jürgen Wolf, HTML5 und CSS3, Rheinwerk Computing, 2016
- Klaus Franz, Handbuch zum Testen von Web-Applikationen, Springer, 2007

Wissenschaftliches Arbeiten (WIA)

Name	Wissenschaftliches Arbeiten (WIA)
Kürzel	WIA
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Hausarbeit (HA)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Methoden wissenschaftlichen Arbeitens
- Literaturrecherche
- kritischer Umgang mit Quellen und Literatur
- Aufbau wissenschaftlicher Arbeiten
- Wissenschaftliche Argumentation
- Zitierweisen
- Vermeidung von Plagiaten
- Schreiben von wissenschaftlichen Texten

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- den Unterschied zwischen nichtwissenschaftlichen und wissenschaftlichen Denk- und Arbeitsweisen zu erklären
- bei der eigenen Arbeit einem wissenschaftlichen Ansatz zu folgen;
- die Bedeutung und Grenzen des Einsatzes von wissenschaftlichen Modellen für die Erkenntnis und für den Einsatz bei Managementaufgaben zu verstehen;
- wissenschaftliche Thesen aufzustellen und zu formulieren;
- Fachliteratur systematisch zu recherchieren, mit Quellen wissenschaftlich umzugehen, deren Qualität zu bewerten und in die eigene Arbeit sachgerecht einzubeziehen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- alle Dozenten der Fakultät für Informatik

Literatur

- Wissenschaftliches Arbeiten: Erfolgreich bei Bachelor- und Masterarbeit. Manuel René Theisen, Martin Theisen. Vahlen, 2017.

Wahlpflichtmodule

3D-Modellierung und Spieleentwicklung (3MS)

Name	3D-Modellierung und Spieleentwicklung (3MS)
Kürzel	3MS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Ihme
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Lineare Algebra für die mathematischen Grundlagen und Bildverarbeitung für die Anwendung von Algorithmen zur Bildverbesserung.
- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Programmierung 3 (PR3)

Inhalte

- Modellierung, Blender-GUI
- 3D-Koordinaten, Koordinatentransformationen und Geometrische Grundlagen
- Physically-Based-Materials (PBR) und Shader-Programmierung
- Texturen, UV-Koordinaten und Texturabbildung
- Beleuchtung & photorealistisches Rendering
- Animation fester Körper & Humanoide
- Node-Editor, Visuelle Programmierung, Compositing von Renderebenen
- Spieleentwicklung mit Kollisionserkennung, Interaktionen, Scripting,
- Model-Import
- Management von Kreativprojekten, Rollen in Kreativteams, Rapid Prototyping

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Kreativprojekte zu planen und durchzuführen,
- Frameworks (z. B. Spiele-Entwicklungsframeworks) auf eigene Projekte zu adaptieren
- Konzepte zu visualisieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Tobias Günther

Literatur

- John Vince, Handbook of computer animation
- Carsten Wartmann, Das Blender-Buch
- Marius Apetri, 3D-Grafikprogrammierung
- Jeremy Birn, Lighting & Rendering
- James Chronister, Blender Basics Classroom Tutorial Book

Anwendungscontainer und Docker (ACD)

Name	Anwendungscontainer und Docker (ACD)
Kürzel	ACD
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Sommersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Specht
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Klausur 60 Minuten (K60)

Empfohlene Vorkenntnisse

- Datemanagement (DM)
- Webbasierte Systeme (WEB)
- Verteilte Systeme (VS), kann parallel gehört werden

Inhalte

- Komponenten, Services, Micro Services
- Virtualisierungskonzepte: Hypervisor Typ 1, Hypervisor Typ 2, Anwendungscontainer
- Technische Basis von Anwendungscontainern: Linux Namespaces, Overlay-Dateisysteme, Overlay-Netzwerke
- Docker-Grundlagen: Architektur, Registry, Hub, Images, Container
- Erstellen eigener Images
- Docker Service-Komposition
- Kubernetes-Einführung

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Docker Desktop auf dem eigenen Rechner installieren und administrieren
- Einfache Micro Service basierte Softwaresysteme entwerfen und für Betrieb in Anwendungscontainern vorbereiten
- Eigene Images erstellen und in Containern zum Laufen bringen
- Anwendungen aus (Micro) Services komponieren und orchestrieren
- Services ausfallsicher betreiben und skalieren

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	75
Selbständiges Bearbeiten der Übungen	30
Prüfungsvorbereitung	15
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Thomas Specht

Literatur

- Docker: Get Started, https://www.docker.com/get-started
- Docker: Download and Install, https://docs.docker.com/get-docker/
- Docker Engine: https://docs.docker.com/engine/
- Docker Compose: https://docs.docker.com/compose/
- Kubernetes: https://kubernetes.io/de/docs/concepts/overview/what-is-kubernetes/

Automatentheorie und formale Sprachen (AFS)

Name	Automatentheorie und formale Sprachen (AFS)
Kürzel	AFS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Lutz Strüngmann
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

Es ist von Vorteil, wenn Sie Grundlagen iin der theoretischen Informatik und Mathematik mitbringen. Die Veranstaltung kann jedoch auch ohne diese Grundlagen besucht werden, da alle wichtigen Voraussetzungen wiederholt werden.

Inhalte

- In der Veranstaltung soll eine Einführung in die Automatentheorie und die Theorie der formalen Sprachen gegeben werden. Mit dem Begriff Automatentheorie ist das Studium abstrakter Rechengeräte gemeint. Als es noch keine Computer gab entwickelte Alan Turing eine abstrakte Maschine, die über sämtliche Fähigkeiten heutiger Computer verfügte. Ziel war es zu differenzieren zwischen dem, was eine Maschine berechnen kann und dem, was sie nicht berechnen kann. In den 40er und 50er Jahren wurden dann einfachere Maschinen untersucht, die heute als endliche Automaten bezeichnet werden und sich für verschiedene Zwecke als außerordentlich nützlich erwiesen haben. Zudem begann der Linguist Chomsky, formale Grammatiken zu untersuchen, die eine enge Verwandschaft zu abstrakten Automaten aufweisen. Einige dieser Konzepte wie endliche Automaten und bestimmte Arten formaler Grammatiken werden heute im Design und im Aufbau wichtiger Arten von Software verwendet. Neben Beweistechniken werden wir die Grundlagen der Automatentheorie und der formalen Sprachen legen und an Beispielen erörtern. Unter anderem werden deterministische und nicht-deterministische endliche Automaten, Push down Automaten, reguläre Ausdrücke und Sprachen, kontextfreie und kontextsensitive sowie Turing erkennbare und Turing entscheidbare Sprachen besprochen.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Grundlagen der Automatentheorie und der formalen Sprachen zu beschreiben.
- zu gegebenen Automaten/Grammatiken die passende Sprache zu erkennen sowie zu einer Sprache einen Automaten/eine Grammatik zu definieren.
- die verschiedenen Arten von endlichen Automaten sowie formalen Sprachen zu beschreiben und sie beherrschen deren Eigenschaften sowie entsprechende Algorithmen zur Umwandlung.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	20
Präsenzübungen und Testate	40
Eigenstudium zur Vor- und Nachbereitung	60
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Lutz Strüngmann

Literatur

- Folgende Bücher können ergänzend zur Veranstaltugn gelesen werden:
- Schöning: Theoretische Informatik – kurzgefaßt. Spektrum, 2008. (5. Auflage)
- Hollas, Boris: Grundkurs Theoretische Informatik, Spektrum Akademischer Verlag 2007
- Rich: Automata, Computability, and Complexity, Pearson 2008
- Asteroth, Baier: Theoretische Informatik, Pearson, 2003.
- Vossen, Witt: Grundkurs Theoretische Informatik, vieweg, 2006.

Agile Softwareentwicklung (AGI)

Name	Agile Softwareentwicklung (AGI)
Kürzel	AGI
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Bestandene Prüfung in Programmieren 1 (PR1) und Programmieren 2 (PR2)
- Gute Programmierfähigkeiten
- Bestandene Prüfung in Software Engineering 1 (SE1)
- Erfahrungen in der Software-Entwicklung, zum Beispiel aus dem Softwareprojekt (SP) und nach Möglichkeit dem Praktischen Studiensemester (PS)

Inhalte

- Methoden der agilen Software-Entwicklung, z. B. Extreme Programming, Scrum, Kanban
- Techniken wie paarweises Programmieren, einfaches Design, testgetriebene Entwicklung, Refactoring, fortlaufende Integration, kleine Releases
- Werkzeuge für die Umsetzung der genannten Techniken und verschiedener Methoden (Entwicklungstools, etc.)
- Einsatz der erlernten Kenntnisse in einem zusammenhängen Mini-Projekt

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Prinzipien und Methoden der agilen Software-Entwicklung zu erläutern,
- je nach Kontext geeignete Elemente der agilen Entwicklung auszuwählen und einzusetzen,
- einzelne Techniken für die Aufwandsschätzung und die Entwicklung anzuwenden.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Selbständiges Bearbeiten der Übungen	30
Projektarbeit	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Peter Knauber
- Roland Schotte

Literatur

- K. Beck. Exterme Programming Explained: Embrace Change. Addison-Wesley, 2000
- J. Benson & T. DeM. Barry: Personal Kanban. CreateSpace Independent Publishing Platform, 2011
- W. Bleek, H. Wolf: Agile Softwareentwicklung. dpunkt-Verlag, 2008
- M. Fowler Refactoring: Improving the Design of Existing Code. Addison-Wesley, 1999
- M. Fowler, J. Highsmith: The Agile Manifesto. Software Development Online, 2001
- P. Hruschka, C. Rupp: Agile Softwareentwicklung für Embedded Real-Time Systems mit der UML. Hanser, 2002
- K. Leopold, S. Kaltenecker: Kanban in der IT. Hanser, 2012
- R. Pichler: Scrum. dpunkt-Verlag, 2008
- H. Wolf, S. Roock, M. Lippert: eXtreme Programming. 2. Auflage, dpunkt-Verlag, 2005

Algorithmen für moderne Rechnerarchitekturen (ALR)

Name	Algorithmen für moderne Rechnerarchitekturen (ALR)
Kürzel	ALR
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Ihme
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Mündliche Prüfung (M)

Empfohlene Vorkenntnisse

- Technische Informatik 1 (TEI1)
- Technische Informatik 2 (TEI2)

Inhalte

- Unterschiedlichen Rechnerarchitekturen
- CPU, DSP, GPU
- Speicherhierarchieen
- Kommunikation
- Hardwarenahe Programmierung (Hardware-Software Zusammenspiel)
- Unterstützung für Multiprocessing in Programmiersprachen
- Virtuelle Maschinen
- Optimierung/Profiling
- GPU-Programmierung
- nVIDIA CUDA (Compute Unified Device Architecture)
- OpenCL

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- unterschiedliche Prozessortypen zu unterscheiden und einen für eine Aufgabenstellung geeigneten Prozessortyp auszuwählen,
- parallele Programmierkonzepte anwenden,

- die Leistung von Programmen zu bewerten und diese zu optimieren,
- Konzepte virtueller Maschinen zu erklären und
- parallele Programmiertechniken zum Programmieren auf Grafikkarten anzuwenden.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jörn Fischer

Literatur

- David Kirk, “Programming massively parallel processors”, ELSEVIER Verlag, ISBN-13: 978-0-12-381472-2
- Thomas Rauber, Gundula Rünger, "Parallele Programmierung", 2. Auflage, Springer Verlag, ISBN:978-3-540-46549-2
- T.Ottmann, P.Widmayer, "Algorithmen und Datenstrukturen", 3.Auflage, Spektrum Verlag, ISBN:3-8274-0110-0
- Thomas Beierlein, Olaf Hagenbruch, "TaschenbuchMikroprozessortechnik", 2. Auflage, Fachbuchverlag Leipzig, ISBN:3-446-21686-3
- M.Allen, B.Wilkinson, “Parallel Programming”, Prentice-Hall, ISBN-13:978-0131405639
- Seiler, L. et al: Larrabee: A Many-Core x86 Architecture for Visual Computing. ACM Transactions on Graphics, Vol. 27, No. 3, Article 18, August 2008, 15 pages
- OpenCL - The open standard for parallel programming of heterogeneous systems
- Khronos Group: OpenCL - The open standard for parallel programming of heterogeneous systems
- Parallel Computing Technologies. Proceedings of the 10th International Conference, PaCT 2009, Novosibirsk, Russia, August 31-September 4, 2009

Angular und Node.js (ANO)

Name	Angular und Node.js (ANO)
Kürzel	ANO
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Specht
Dauer	1 Semester
Studiengänge	IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Mündliche Prüfung (M)

Empfohlene Vorkenntnisse

Für dieses Wahlpflichtmodul werden fundierte Kenntnisse in HTML5 und JavaScript vorausgesetzt

- Datenmanagement (DM)
- Webbasierte Systeme (WEB)

Inhalte

- Fortgeschrittene Webarchitekturen
- JavaScript, ECMAScript, TypeScript
- TypeScript: Compiler, Statische Typisierung, Dekoratoren
- Angular: Architektur, Komponenten und Templates, Formulare, Router, Services, Module, RESTful Web Service Clients
- Node.js: Paketmanager npm, asynchrone Aufrufe, Callbacks, Event Loop, Module, Web Server, RESTful Web Services

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Anforderungen an eine komplexe Webanwendung analysieren und eine geeignete Architektur entwerfen
- Eigenschaften und Konzepte TypeScript-basierter Web-Programmiersprachen beschreiben und bewerten
- Eigenschaften und Konzepte browserseitiger Webframeworks beschreiben und bewerten
- Webanwedungen mit browserseitigen Webframeworks entwickeln

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Eigenstudium zur Vor- und Nachbereitung	75
Selbständiges Bearbeiten der Übungen	20
Prüfungsvorbereitung	10
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Thomas Specht

Literatur

- TypeScript Tutorial: https://www.tutorialspoint.com/typescript
- Angular Tutorial + Fundamentals: https://angular.io/docs
- Node.js Tutorial: https://www.tutorialspoint.com/nodejs

Angewandte Projektarbeit: Visualisierung (APV)

Name	Angewandte Projektarbeit: Visualisierung (APV)
Kürzel	APV
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Till Nagel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Grundlagen der Datenvisualisierung (GDV)

Inhalte

- Nutzerzentrierte Anforderungsanalyse
- Entwurfsprozess
- Iterative Implementierung
- Agile Methoden der Softwarentwicklung
- Creative Thinking Methoden
- Komplexe interaktive Visualisierungsysteme
- Visual Analytics

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- fortgeschrittene Konzepte, Techniken und Methoden der Informationsvisualisierung einzuschätzen und deren Vor- und Nachteile kritisch zu reflektieren.
- eigenständig zu recherchieren und akademische Artikel als Quelle zu nutzen.
- eigene nutzergerechte Visualisierungen in Zusammenarbeit mit den Partnern zu entwickeln.
- den Daten und Aufgaben angemessene Visualisierungsformen zu wählen.
- verschiedene Visualisierungs- und Interaktionstechniken prototypisch zu implementieren.
- die Umsetzung und ihre Anwendung des entwickelten Systems zu reflektieren.

Semesterwochenstunden

Übung	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Projektarbeit	120
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Till Nagel

Literatur

- Munzner, T.: Visualization Analysis and Design, CRC Press, 2014
- Spezifische, dem Projektthema angemessene Literatur.

Big Data Engineering and Analysis (BDEA)

Name	Big Data Engineering and Analysis (BDEA)
Kürzel	BDEA
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Oliver Hummel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Datenmanagement (DM)
- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Parallele Programmierung (PP)
- keine Angst vor Programmieren, Algorithmen und ein wenig Statistik

Inhalte

- Was ist Big Data (4 Vs) und was bedeutet Skalierbarkeit?
- Big-Data-Architekturen (wie Lambda, Micro Services u.ä.) sowie ihre Entwurfsprinzipien
- CAP-Theorem, BASE vs. ACID und ihre Auswirkungen auf die Architektur
- Big-Data-Frameworks (Batch vs. Streaming: Hadoop, Spark, Flink) und ihr praktischer Einsatz
- NoSQL-Datenbanken: Auswahlkriterien und Nutzung (z. B. Cassandra u. Kafka)
- Big-Data-Algorithmen (z. B. HyperLogLog, Bloom-Filter)
- Daten-Analyse (z. B. mit R)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Herausforderungen bei der Verarbeitung von großen Datenmengen zu erkennen
- und unter Anwendung der vorgestellten Methoden und Werkzeuge skalierbare Lösungen zu entwerfen und umzusetzen

Semesterwochenstunden

Vorlesung	2 SWS
Übung	1 SWS
Labor	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	25
Präsenzübungen und Testate	25
Eigenstudium zur Vor- und Nachbereitung	25
Selbständiges Bearbeiten der Übungen	25
Prüfungsvorbereitung	50
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Oliver Hummel
- Marcus Kessel

Literatur

- Jure Leskovec, Anand Rajaraman, Jeff Ullman: Mining Massive Data Sets, Cambridge University Press, 2014.
- Nathan Marz and James Warren: Big Data - Principles and best practices of scalable realtime data systems, Manning, 2015.
- Ricardo Baeza-Yates, Berthier Ribeiro-Neto: Modern Information Retrieval, ACM Press, 2010.
- Martin Kleppmann: Designing Data-Intensive Applications, O'Reilly, 2017.
- Robert Kabacoff: R in Action (2nd Edition), Manning, 2015.
- Oliver Hummel: Blog Big Data Engineering: https://bigdataengineering.blogspot.de

Bioinformatik (BIM)

Name	Bioinformatik (BIM)
Kürzel	BIM
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Wintersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Markus Gumbel
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Grundlagen der Molekularbiologie
- Grundlagen der Biotechnologie (v.a. Sequenzierung)
- Informationssysteme in der Biologie und Wirkstoffforschung
- Datenbanken für Gene und Proteine
- Datenbanken für Moleküle in der Wirkstoffforschung (Compounds)
- Einführung in String- und Tree-Algorithmen und dynamische Programmierung
- Dotplot, paarweises Alignment von Sequenzen
- Score-Matrizen
- Multiples Alignment von Sequenzen
- Analyse von Sequenzen und Genomen
- Vorhersage von Molekülstrukturen und Faltung
- Evolutionsmodelle; Phylogenetische Inferenz
- System-Biologie, Simulationsmethoden zur Entscheidungsunterstützung
- Anwendung der Genomik in der personalisierten Medizin

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- grundlegende molekularbiologische Vorgänge zu beschreiben,

- biologische Fragestellungen der Anwender (Biologen, Mediziner) einzuschätzen,
- die verschiedenen Bioinformatik-Algorithmen anzuwenden und zu erklären, wo diese ihren Ursprung in der Informatik haben,
- die Einsatzgebiete und Grenzen der Verfahren zu verstehen,
- zu hinterfragen, welche Bioinformatik-Methode am besten zur Lösung eines medizinisch-biologischen Problems herangezogen werden kann.

Semesterwochenstunden

Vorlesung	3 SWS
Übung	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	15
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Markus Gumbel

Literatur

- H.J. Böckenhauer, D. Bongartz: Algorithmische Grundlagen der Bioinformatik
- D. M. Mount: Bioinformatics. Cold Spring Harbor Laboratory Press 2004, 2. Aufl., ISBN 978-0879697129
- R. Merkl, St. Waack: Bioinformatik Interaktiv: Algorithmen und Praxis. Wiley-Vch, 2009, 2. Aufl., ISBN 978-3527325948
- M. Th. Hütt, M. Dehnert: Methoden der Bioinformatik: Eine Einführung. Springer 2006, ISBN 978-3540256878
- R. Durbin, S. Eddy, A. Krogh, G. Mitchison: Biological Sequence Analysis. Cambridge University Press, Cambridge, UK, 1998.
- D. Gusfield: Algorithms on Strings, Trees and Sequences. Cambridge University Press, Cambridge, UK, 1997.

Bildverarbeitung (BIV)

Name	Bildverarbeitung (BIV)
Kürzel	BIV
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Ivo Wolf
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Mathematik für die Informatik 1 (MA1)
- Mathematik für die Informatik 2 (MA2)
- Mathematik für die Informatik 3 (MA3)

Inhalte

- Repräsentation von digitalen Bildern und Bildvolumen, Farbräume
- Punkt-Operatoren
- Lokale Opteratoren
- Feature Engineering und Objekterkennung
- Segmentierung
- Anwendungen von Deep Learning in der Bildverarbeitung
- Bildtransformationen
- Registrierung
- Anwendungen in der Medizin

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- zentrale Konzepte und Methoden der Bildverarbeitung anzuwenden,
- Einsatzgebiete und Grenzen der Verfahren zu erläutern,
- erlernte Verfahren auf neue Anwendungsbereiche zu übertragen,
- einschlägige, aktuelle Forschungsthemen einzuordnen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	30
Prüfungsvorbereitung	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Ivo Wolf

Literatur

- A. Nischwitz, M. Fischer, P. Haberäcker, G. Socher: Bildverarbeitung, Springer, 2020
- D. Sundararajan: Digital Image Processing, Springer, 2017
- R.C. Gonzalez, R.E. Woods: Digital image processing, Pearson Prentice Hall, 2017
- B. Jähne: Digitale Bildverarbeitung, Springer, 2012
- K.D. Toennies: Grundlagen der Bildverarbeitung, Pearson-Verlag, 2005
- M. Sonka, V. Hlavac, R. Boyle: Image Processing, Analysis, and Machine Vision, Addison-Wesley, 2007
- T. Lehmann, W. Oberschelp, E. Pelikan: Bildverarbeitung für die Medizin, Springer, 1997
- Handbücher der eingesetzten Programmiersprache

Campusmanagement als Anwendungskontext für Webanwendungen (CAW)

Name	Campusmanagement als Anwendungskontext für Webanwendungen (CAW)
Kürzel	CAW
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- Dieser Kurs betrachtet die kommende Campusmanagement-Lösung der Hochschule als Webanwendung. Im Rahmen dieser Anwendung beschäftigen wir uns mit der Gestaltung und dem Customizing der Webanwendung mit den vom Framework bereitgestellten Möglichkeiten.
- In einem ersten Schritt werden die Möglichkeiten der Plattform betrachtet, und im zweiten Schritt werden in kleinen Projekten Veränderungen an der Webanwendung vorgenommen.
- Die Lehrveranstaltung ist stark selbstgesteuert und praktisch. Die Anzahl der Mini-Projekte und Themen richtet sich nach der Anzahl der teilnehmenden Studierenden.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Anforderungen an Webanwendungen aufzunehmen, zu strukturieren und zu dokumentieren,
- Umsetzungsoptionen im Rahmen eines ausgewählten Frameworks zu ermitteln, abzuwägen und auszuwählen,
- und die Anforderungen im Rahmen des Customizings umzusetzen.

Semesterwochenstunden

Projekt	3 SWS
Workshop	1SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Eigenstudium zur Vor- und Nachbereitung	30
Projektarbeit	90
Workshop	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Sachar Paulus
- Tatiana Derevyankina

Literatur

- wird in der Lehrveranstaltung bekannt gegeben

Challenge-Based Making (CBM)

Name	Challenge-Based Making (CBM)
Kürzel	CBM
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Kirstin Kohler
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Schwerpunkt

- Software Engineering

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- Design Thinking Methoden: Ideation, divergent and convergent thinking
- Die Bedeutung von Prototyping im Innovationsprozess
- Evaluation von Prototypen
- Hardware und Software "Making" Technologien
- Physical Computing und Elektronik
- Verschiedene Sensoren und Aktuatoren als Input und Output
- Mikrocontroller (Arduino)
- Design von Schaltplänen mit Fritzing
- Exploration und Implementierung intelligenter Kontrollmechanismen und Parameter für dynamische und innovative interaktive Systeme

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Ideen im Kontext zu erklären und Entwurfsentscheidungen zu begründen,
- Konvergentes und divergentes Denken anzuwenden,
- Design-Alternativen zu evaluieren und auszuwählen und
- Technologien des Physical Computing zur Umsetzung ihrer Ideen auszuwählen und anzuwenden.

Semesterwochenstunden

Vorlesung	2 SWS
Projekt	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Kirstin Kohler
- Prof. Thomas Smits

Literatur

- John Boxall, Volkmar Gronau; Arduino-Workshops: Eine praktische Einführung mit 65 Projekten; September 2013, dpunkt.verlag
- Michael Lewrick, Patrick Link und Larry Leifer ; Design Thinking Playbook; 2017, Vahlen Verlag

Cross-Plattform-Development mit dem Flutter-Framework (CPD)

Name	Cross-Plattform-Development mit dem Flutter-Framework (CPD)
Kürzel	CPD
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Oliver Hummel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Webbasierte Systeme (WEB)
- Software Engineering 2 (SE2)

Inhalte

- Entwicklung von mobilen, Web- und nativen Applikationen mit Flutter
- Cross-Plattform und Cross-Device Development
- Dart (Programmiersprache)
- Flutter (Framework)
- Testen von Cross-Plattform-Applikationen
- Techniken für Leichtgewichtige Software-Entwicklung bis hin zum Minimal Viable Product

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Konzepte, Gemeinsamkeiten und Unterschiede von mobilen Apps, Web-Applikationen und nativen Applikationen zu beschreiben
- eine Applikation für unterschiedliche Plattformen mit einer Code-Basis zu konzipieren
- eine Applikationen mit dem Flutter-Framework zu implementieren
- Programme in der Programmiersprache Dart zu erstellen und eine passende Entwicklungsumgebung zu verwenden

- das Projekt selbstständig und leichtgewichtig zu planen und im zeitlich vorgegebenen Rahmen umzusetzen
- Methoden und Tools zur Qualitätssicherung derartiger Applikationen anzuwenden

Semesterwochenstunden

Vorlesung	3 SWS
Übung	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Präsenzübungen und Testate	15
Eigenstudium zur Vor- und Nachbereitung	50
Selbständiges Bearbeiten der Übungen	40
Summe	**150**

Dozentinnen / Dozenten

- Dr. Christoph Giess
- Philipp Bletzer

Literatur

- Biessek: "Flutter for Beginners", Packt Publishing, 2019
- Clow: "Learn Google Flutter Fast", https://github.com/markclow/flutter_book_examples
- Mainkar, Giordano: "Google Flutter Mobile Development Quick Start Guide", Packt Publishing, 2019
- Paine: "Beginning App Development with Flutter", APress, 2019
- Windmill: "Flutter in Action", Manning, 2020
- Zaccagnino: "Programming Flutter - Native, Cross-Platform Apps the Easy Way", The Pragmatic Programmers, 2020
- Zammetti: "Practical Flutter", APress, 2019

Cybersicherheit in der Prozessindustrie (CPI)

Name	Cybersicherheit in der Prozessindustrie (CPI)
Kürzel	CPI
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Grundlagen der IT-Sicherheit
- Grundlagen der Automatisierungstechnik

Inhalte

- Moderne Produktionsanlagen sind durch ihren hohen Vernetzungsgrad und die Einbindung von Standard-IT-Komponenten zunehmend Cybersicherheitsrisiken ausgesetzt. Gleichzeitig lassen sich etablierte Maßnahmen der IT-Sicherheit in diesem Umfeld nicht uneingeschränkt umsetzen.
- Die Vorlesung vermittelt das erforderliche Wissen und die Kompetenzen, um Cybersicherheit im Kontext industrieller Produktion bewerten und verbessern zu können. Der Fokus der Vorlesung liegt dabei auf Produktionsanlagen der Prozessindustrie (Chemie, Pharmazie, Lebensmittelherstellung, etc.). Die vermittelten Inhalte sind aber nicht nur in der Prozessindustrie, sondern auch in anderen Branchen (z. B. Automobilindustrie, Energie- und Wasserversorgung) und Kontexten (z. B. Gebäudeautomatisierung) anwendbar.
- Systeme, Komponenten und Vernetzung in der Prozessautomatisierung
- Herausforderungen der Cybersicherheit in der Prozessindustrie
- Spezifische Bedrohungen und Schwachstellen
- Etablierte Standards und Best Practices
- Geeignete Methoden und Werkzeuge für Sicherheitsanalysen
- Auswahl und Umsetzung von Sicherheitsmaßnahmen (Netzwerksicherheit, System-/ Anwendungssicherheit, organisatorische Maßnahmen)
- Cybersicherheit von Sicherheitssystemen (Safety Instrumented Systems)
- Maßnahmen zur Reduktion der Auswirkungen von Sicherheitsvorfällen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Bedrohungen angemessen einzuordnen,
- Auswirkungen von Sicherheitsvorfällen zu verstehen,
- Sicherheitsschwachstellen zu identifizieren,
- Sicherheitsrisiken zu analysieren und zu bewerten,
- Maßnahmenempfehlungen und Sicherheitskonzepte zu erarbeiten und
- die Ergebnisse von Sicherheitsanalysen zielgruppengerecht zu kommunizieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	60
Summe	**150**

Dozentinnen / Dozenten

- Dr. Felix Kahrau

Literatur

- Ackerman, P. (2017). Industrial Cybersecurity: Efficiently secure critical infrastructure systems. Packt Publishing Ltd.
- Bochman, A. A., & Freeman, S. (2021). Countering Cyber Sabotage: Introducing Consequence-driven, Cyber-informed Engineering (CCE). CRC Press.
- Knapp, E. D., & Langill, J. (2014). Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems. Syngress.
- Marszal, E., & McGlone, J. (2019). Security PHA review for consequence-based cybersecurity. International Society of Automation (ISA).
- Niemann, K. H. (2021). Abgrenzung der IT-Sicherheitsnormenreihen ISO 27000 und IEC 62443: eine Sicht auf automatisierungstechnische Anlagen der Fertigungs-und Prozessindustrie. ABB Automation Products GmbH, Heidelberg. [Verfügbar unter: https://serwiss.bib.hs-hannover.de/files/1973/Abgrenzung_ISO_27001_IEC_62443_V13.pdf]

Cyber-Security-Consulting (CSC)

Name	Cyber-Security-Consulting (CSC)
Kürzel	CSC
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Mündliche Prüfung (M)

Empfohlene Vorkenntnisse

- Praktische Erfahrungen als Werksstudent oder Praktikant im Cyber Security Consulting

Inhalte

- Überblick über das Beratungsportfolio im Bereich Cyber Security Consulting,
- Beispielhafte Darstellungen von Projektbearbeitungsmethoden,
- Strategien für Kundengespräche,
- Praktische Bearbeitung eines Kundenprojekts,
- Präsentation der Ergebnisse.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- sich selbst in ein neues Themengebiet einzuarbeiten, um sich einen Überblick darüber zu verschaffen,
- sich auf ein Kernthema eines realen Kundenprojekts zu fokussieren,
- Herausforderungen des Kunden zu verstehen,
- Lösungsansätze für den Kunden zu erarbeiten,
- den selbst erarbeiteten Lösungsansatz entsprechend vorzustellen,
- einen Bericht über das Kundenprojekt zu formulieren.

Semesterwochenstunden

Projekt	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Eigenstudium zur Vor- und Nachbereitung	20
Projektarbeit	130
Summe	**150**

Dozentinnen / Dozenten

- Gerrit Aufderheide, Antje Anger

Computer Vision (CVIS)

Name	Computer Vision (CVIS)
Kürzel	CVIS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Oliver Hummel
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

Grundlegende Kenntnisse aus den Modulen ...

- Einführung in die Höhere Mathematik
- Lineare Algebra
- ... sowie gängige Grundlagen der Informatik

Inhalte

- Kamera-Modell
- Kamera-Kalibrierung
- Epipolargeometrie
- Pixel Erkennung und Vergleich
- 3D Rekonstruktion
- Struktur und Bewegung (SfM)
- Optischer Fluss, Szenen Fluss
- Tiefenkameras
- Anwendungen: Autonomes Fahren, Industrielle Bilderverarbeitung, usw.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- zentrale Konzepte und Methoden der 3D Computer Vision zu verstehen
- Algorithmen zur visuellen Wahrnehmung zu entwickeln
- Einsatzgebiete und Grenzen der Verfahren zu erläutern

- die erlernten Prinzipien in Anwendung (Autonomes Fahren, Industrielle Bilderverarbeitung, usw.) zu bringen

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	50
Eigenstudium zur Vor- und Nachbereitung	50
Selbständiges Bearbeiten der Übungen	25
Prüfungsvorbereitung	25
Summe	**150**

Dozentinnen / Dozenten

- Dr. Oliver Wasenmüller (DFKI, Kaiserslautern)

Literatur

- Richard Hartley and Andrew Zisserman: Multiple View Geometry in Computer Vision
- Richard Szeliski: Computer Vision: Algorithms and Applications
- Gary Bradski and Adrian Kaehler: Learning OpenCV

Digitale Forensik (DIF)

Name	Digitale Forensik (DIF)
Kürzel	DIF
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Sommersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Jessica Steinberger
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Hausarbeit (HA)

Empfohlene Vorkenntnisse

- Kenntnisse über digitale Zahlendarstellungen und Kodierungen (z. B. ASCII)
- Grundlegende Programmierkenntnisse

Inhalte

- Klassische forensische Wissenschaften und digitale Forensik
- Grundlagen der digitalen Forensik
- Digitale Spuren (Entstehung, Manipulier- und Kopierbarkeit, Personenbezogenheit)
- Datenquellen für forensische Untersuchungen (z. B. Desktop, Tablet, Smartphone, Network, Car, IoT)
- Analyse mit forensischen Tools (Sleuthkit, Autopsy)
- Hashfunktionen in der digitalen Forensik
- Vorgehensmodelle und Gutachtenerstellung

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Grundlagen der digitalen Forensik und der Teilgebiete zu beschreiben und können diese anwenden.
- die Entstehung, der Manipulier- und Kopierbarkeit sowie der Personenbezogenheit von digitalen Spuren zu beschreiben.
- das grundlegende Konzept sowie die Eigenschaften der gängigen Dateisysteme beschreiben und können mit diesem Wissen eine Dateisystemanalyse durchführen.
- grundlegenden Schritte eines IT-Forensikers beschreiben und können mit allgemeinen und speziellen forensischen Werkzeugen sicher umgehen.

- die grundlegenden Funktionsweise kryptographischer Hashfunktionen zu benennen und sind mit deren Rolle in der digitalen Forensik vertraut.
- gängige forensischen Tools zu benennen und können wichtige Ergebnisse daraus eigenständig entnehmen.
- Grundprinzipien der IT-Forensik bei der Bearbeitung einer forensischen Aufgabenstellung einzuhalten und können diese bei einer forensischen Untersuchung anwenden.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jessica Steinberger

Literatur

- Bundesamt für Sicherheit in der Informationstechnik, "Leitfaden IT-Forensik", Version 1.0.1, https://www.bsi.bund.de/DE/Them Verwaltung/Sicherheitspruefungen/IT-Forensik/forensik_node.html
- Alexander Geschonneck: Computer-Forensik. Computerstraftaten erkennen, ermitteln, aufklären. 6. aktualisierte und erweiterte Auflage. dpunkt Verlag, Heidelberg 2014,ISBN 978-3-86490-133-1.
- Eoghan Casey (Hrsg.): Handbook of computer crime investigation. Forensic tools and technology. 6th Printing. Elsevier Academic Press, Amsterdam u. a. 2007, ISBN 978-0-12- 163103-1.

Digitale Transformation (DTF)

Name	Digitale Transformation (DTF)
Kürzel	DTF
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Referat (R)

Empfohlene Vorkenntnisse

Inhalte

- Grundlagen von E-Business und digitaler Transformation
- Digital Business
- Digitale Medien
- Digitales Marketing
- Business Model
- Trends & Innovation
- Organisationale Transformation
- Design Thinking: Fallstudie

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Digitalisierung und Industrie 4.0 im Kontext gesellschaftlicher, technologischer, wirtschaftlicher und politischer Entwicklungen einordnen;
- die Potenziale digitaler Wertschöpfung erkennen;
- digitale und analoge Geschäftsmodelle ausarbeiten und integrieren;
- die Customer Experience und das User-Centered-Design (Design Thinking) herausarbeiten;
- Vorgehensmodelle der digitalen Transformation anwenden und Treiber identifizieren;
- den Einfluss des Menschen (Digital Leadership, New Work und Unternehmenskultur) in der Transformation erkennen;
- konkrete Fälle digitaler Transformation analysieren und Maßnahmen für den Change Prozess ableiten;
- Megatrends und Innovationspotenziale der Zukunft erschließen.

Arbeitsaufwand (work load)

Präsenzstudium	45
Präsenzübungen und Testate	15
Eigenstudium zur Vor- und Nachbereitung	45
Selbständiges Bearbeiten der Übungen	45
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Christoph Sandbrink

Literatur

- Gassmann, O. & Sutter, P. (2019): Digitale Transformation gestalten, 2. Aufl., München: Carl Hanser.
- Kollmann, T. (2019). E-Business – Grundlagen elektronischer Geschäftsprozesse in der Digitalen Wirtschaft, 7. Aufl., Wiesbaden: Springer Gabler.
- Osterwalder, A. & Pigneur, Y. (2011): Business Model Generation – Ein Handbuch für Visionäre, Spielveränderer und Herausforderer, Frankfurt/New York: Campus.
- Wirtz, B. W. (2018). Electronic Business. Wiesbaden: Springer Gabler.

Entwurfsmuster für funktionale Programmierung (EFP)

Name	Entwurfsmuster für funktionale Programmierung (EFP)
Kürzel	EFP
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Sandro Leuchter
Dauer	1 Semester
Studiengänge	IB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Datenmanagement (DM)
- Software Engineering 1 (SE1)
- Software Engineering 2 (SE2)
- Webbasierte Systeme (WEB)
- Softwareprojekt (SP)

Inhalte

- Viele eingeführte Entwurfsmuster, die im Wesentlichen aus einem objektorientierten Ansatz heraus entwickelt worden waren, müssen im Kontext der funktionalen Programmierung neu bewertet werden. Es zeigt sich, dass sie im Rahmen der Anwendung einer konkreten funktionalen Programmiersprache teilweise nicht mehr nützlich sind oder in der Implementierung grundsätzlich abweichen. Zudem wurden im Zusammenhang mit der Programmierung in funktionalen Sprachen weitere Entwurfsmuster sowohl auf der Ebene des Systemarchitektur als auch auf der eher idiomatischen Ebene der Programmierung neue Muster identifiziert.
- Diese Lehrveranstaltung gliedert sich in drei Teile: ein einführender Programmierkurs, der theoretische Teil und ein Projekt.
- Im Programmierkurs (ca. 20% des Umfangs im Präsenzstudium) wird am Beispiel einer konkreten funktionalen Programmiersprache, die für die praktische Anwendung relevant ist (z. B. Clojure, Scala oder Elixir), die Grundlagen der funktionalen Programmierung und der idiomatische Gebrauch der verwendeten Programmiersprache anhand von kleinen Beispielen vermittelt. Dieser Teil ist sehr kurz und steht nicht im Vordergrund dieser Lehrveranstaltung.
- Im theoretischen Teil (ca. 40% des Umfangs im Präsenzstudium) wird eine Reihe gebräuchlicher Entwurfsmuster auf die Anwendbarkeit im funktionalen Paradigma untersucht. Das geschieht anhand von Referaten der Teilnehmer. Abhängig von der Anzahl der Teilnehmer auch durch den Dozenten. Außerdem werden Entwurfsmuster, die speziell im Kontext der funktionalen Programmierung entstanden sind, analysiert.
- Im Projektteil (ca. 40% des Umfangs im Präsenzstudium) werden passende Entwurfsmuster auf ein komplexeres Problem angewendet. Im Projekt wird eine verteilte Anwendung mit funktionalen Mitteln programmiert. Das

System, das im Projekt in Kleingruppen entwickelt wird, soll web-basierte Schnittstellen haben und eine im Rahmen der Lehrveranstaltung zu konkretisierende E-Learning Anwendung umsetzen.

- Objekt-orientierte Entwurfsmuster und idiomatische Konstrukte, die bei funktionaler Programmierung ersetzt werden müssen (beispielhafte Auswahl): Functional Interface, State Carrying Functional Interface, Command, Builder For Immutable Object, Iterator, Template Method, Strategy, Null Object, Decorator, Visitor, Dependency Injection
- Entwurfsmuster und idiomatische Verwendung von Konstrukten bei funktionaler Programmierung (beispielhafte Auswahl): Tail Recursion, Mutual Recursion, Filter-Map-Reduce, Chain of Operations, Function Builder, Memoization, Lazy Sequence, Focused Mutability, Customized Control Flow, Domain-Specific Language, Railway Oriented Programming (Monaden für Error Handling)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Entwurfsmuster für Softwarearchitekturen, die im objektorientierten und funktionalen Paradigma umgesetzt werden sollen, zu beschreiben und zu erläutern.
- die funktionale Programmiersprache, die im Rahmen der Lehrveranstaltung benutzt wird, professionell einzusetzen und komplexe Softwareprojekte unter Anwendung von spezifischen Entwurfsmustern, die im Umfeld der funktionalen Programmierung entstanden sind, umzusetzen.
- Entwurfsmuster auf ihre Anwendbarkeit im Rahmen des Software Engineerings im funktionalen Programmierparadigma zu analysieren und auf ihre Anwendbarkeit hin beurteilen.

Semesterwochenstunden

Vorlesung	1 SWS
Übung	3 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	15
Präsenzübungen und Testate	45
Eigenstudium zur Vor- und Nachbereitung	45
Projektarbeit	45
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr.-Ing. Sandro Leuchter

Literatur

- Allen, J. (2013). Effective Akka. Sebastopol: O Reilly.
- Backus, J. (1978). Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs. Communications of the ACM 21, (8) 613-641. DOI: 10.1145/359576.359579.

- Bevilacqua-Linn, M. (2013). Functional Programming Patterns in Scala and Clojure. Write Lean Programs for the JVM. Dallas: The Pragmatic Programmers.
- Butcher, P. (2014). Seven Concurrency Models in Seven Weeks. When Threads Unravel. Dallas: The Pragmatic Programmers.
- Emerick, C., Carper, B., & Grand, C. (2012). Clojure Programming. Sebastopol: O Reilly.
- Ford, N. (2014). Functional Thinking. Paradigm Over Syntax. Sebastopol: O Reilly.
- Fowler, M. (2003). Patterns für Enterprise Application-Architekturen. Bonn: mitp-Verlag.
- Hinze, R. (1992). Einführung in die funktionale Programmierung mit Miranda. Stuttgart: Teubner.
- Meier, C. (2015). Living Clojure. Sebastopol: O Reilly.
- Vernon, V. (2016). Reactive Messaging Patterns with the Actor Model. Applications and Integration in Scala and Akka. New York: Addison-Wesley.

Ethik, Recht und Datenschutz (ERD)

Name	Ethik, Recht und Datenschutz (ERD)
Kürzel	ERD
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Klausur 120 Minuten (K120)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Recht, Rechtsgeschäfte
- Datenschutzgrundverordnung
- Informationssicherheit als Führungsaufgabe
- Compliance und Good Corporate Governance

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Rechtsfälle, deren Sachverhalte IT-Probleme beinhalten, zu bewerten,
- Datenschutzanforderungen zu kennen und in Anwendungsfällen umzusetzen,
- Ethische Aspekte beim Einsatz von Informationstechnologie zu berücksichtigen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Sachar Paulus
- Gerrit Aufderheide

Literatur

- Helmut Köhler, Bürgerliches Gesetzbuch, dtv Verlagsgesellschaft, 81. Auflage, 2018
- Gesetz über Urheberrecht und verwandte Schutzrechte (Urheberrechtsgesetz), https://www.gesetze-im-internet.de/urhg/BJNR01

Game Engineering (GAE)

Name	Game Engineering (GAE)
Kürzel	GAE
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Frank Dopatka
Dauer	1 Semester
Studiengänge	IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Voraussetzungen gemäß RGS zum Absolvieren eines Wahlfachs. Vertiefte Kenntnisse in den folgenden Bereichen sind von Vorteil:

- Einführung in die Informatik (EI)
- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Software Engineering 1 (SE1)

Inhalte

- Interdisziplinäres Arbeiten an neuen und vorhandenen Game-Projekten mit Schwerpunkt Gamification und/oder Game Engineering, u.a. Entwicklung von Stories, Game-Balancing, verteilte Software-Architekturen bis hin zur Frontend-Entwicklung.
- Beginn mit einem kurzen Vorlesungsteil in der Schnupperwoche über das Vorgehen und die Regeln, über die grundlegenden Definitionen wie Game Design, GDD, Game Engineering, Gamification sowie Vorstellung von bereits existierenden Game-Projekten.
- Möglichkeit zum pitchen eigener Projektideen in der Folgewoche, anschließend verbindliche Zuordnung der Teams.
- Danach Definition des ersten Sprints für jedes Team, insgesamt sind 3 Sprints vorgesehen.
- Zentrale Vorstellung der Sprint-Ergebnisse zu 3 festen Zwischenterminen vor der gesamten Gruppe.
- Abschlußpräsentation des Gesamtergebnisses bei der iExpo-Messe.
- Abschließende individuelle Notenvergabe im Rahmen eines Personalgespräches.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- in einem kleinen interdisziplinären Team mit verschiedenen Rollen zusammenzuarbeiten.
- sich agil selbst in Teams zu organisieren, Arbeitsaufwände abzuschätzen, Deadlines zu definieren und einhalten.
- sich in der weiten Fachdomäne der Spieleentwicklung zurechtzufinden und lösungsorientiert zu arbeiten.

Semesterwochenstunden

Vorlesung	2 SWS
Projekt	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Projektarbeit	105
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Frank Dopatka

Literatur

- Gunther Rehfeld: Game Design und Produktion; Hanser-Verlag; 2014; Print-ISBN: 978-3-446-43163-8; eBook-ISBN: 978-3-446-43621-3
- Jashan Chittesh: Das Unity-Buch; Dpunkt.Verlag; 2015; ISBN 978-3-864-90232-1
- Susanne Strahringer, Christian Leyh (Hrsg.): Gamification und Serious Games - Grundlagen, Vorgehen und Anwendungen; Springer-Verlag; 2017; Print-ISBN: 978-3-658-16741-7; eBook-ISBN: 978-3-658-16742-4
- Stefan Stieglitz: Gamification – Vorgehen und Anwendung; HMD Praxis der Wirtschaftsinformatik; Ausgabe 6/2015

Global Digital Innovation Program (GDIP)

Name	Global Digital Innovation Program (GDIP)
Kürzel	GDIP
Semester	6/7
Unterrichtssprache	Englisch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Die Auswahl der Teilnehmer erfolgt durch Interviews

- Gute Englischkenntnisse

Inhalte

- Grundlagen des Design Thinking
- Entwickeln einer Produktidee mit den Methoden des Design Thinking, ausgehend von der Challenge eines Unternehmens
- Selbständige Projektarbeit im (nationalen Informatik-)Teilteam und im internationalen, interdisziplinären Team

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Methoden des Design Thinking unter Anleitung anzuwenden
- Nutzer-zentriert und Prototyp-basiert zu arbeiten
- als Informatiker in einem internationalen, interdisziplnären Team innovative Produktideen zu entwickeln

Semesterwochenstunden

Projekt	2 SWS
Workshop	2SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Projektarbeit	120
Workshop	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Thomas Smits
- Catarina Batista

Literatur

- Michael Lewrich, Patrick Link, Larry Leifer: Das Design Thinking Playbook, 2. Auflage, Vahlen, 2018

Grundlagen der Datenvisualisierung (GDV)

Name	Grundlagen der Datenvisualisierung (GDV)
Kürzel	GDV
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Jedes Semester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Till Nagel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Grundlagen der Mensch-Maschine-Kommunikation oder

- Software Engineering 2 (SE2)

Inhalte

- Einführung und Geschichte
- Begriffe und Definitionen
- Aufgaben und Ziele der InfoVis
- Entwurfsprozess und Visualisierungspipeline
- Daten und Informationen
- Datenquellen, -formate, und -typen
- Grundlegende Statistiken
- Kognitiv-psychologische Grundlagen der Informationsvisualisierung
- Semantik und Semiotik
- Visuelle Variablen und Encoding
- Visualisierungstechniken
- Visualisierungstaxonomien und Design Pattern
- Interaktionen und UI in der InfoVis
- Aktuelle Entwicklungen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- grundlegende Konzepte, Techniken und Methoden der Informationsvisualisierung zu verstehen.
- die relevanten kognitiv-psychologischen Grundlagen im Kontext der Visualisierung einzusetzen.
- verschiedene Visualisierungs- und Interaktionstechniken zu benennen und deren Vor- und Nachteile benennen zu können.
- existierende Visualisierungsysteme kritisch zu beurteilen.
- eigene nutzergerechte Visualisierungen zu entwickeln.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Till Nagel

Literatur

- Few, S.: Information Dashboard Design, Analytics Press, 2013
- Meirelles, I.: Design for Information, Rockport, 2013
- Cairo. A.: The Functional Art, New Riders, 2012
- Ware, C.: Information Visualization: Perception for Design, Morgan Kaufmann, 2012
- Tufte, E.: Envisioning Information, Graphics Press, 1990

Grundlagen Neuronaler Netze (GNN)

Name	Grundlagen Neuronaler Netze (GNN)
Kürzel	GNN
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Sommersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Jörn Fischer
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Praktische Arbeit/Projektarbeit (PA)
Prüfungsleistung	Klausur 60 Minuten (K60)

Empfohlene Vorkenntnisse

- Mathematik für die Informatik 1 (MA1)
- Mathematik für die Informatik 2 (MA2)
- Maschinelles Lernen (MLE) von Vorteil

Inhalte

- Neuronenmodelle
- Netzwerk Topologien
- Lernregeln
- Rekurrente Netze
- Dynamische Systeme

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Neuronale Modelle umzusetzen
- Netzwerktopologien einzuordnen
- Lernregeln zu verstehen und anzuwenden
- einfache rekurrente Neuronale Netzwerke zu verstehen
- ein Netzwerk als Dynamisches System zu analysieren

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jörn Fischer
- Prof. Dr. Ivo Wolf

Literatur

- I. Goodfellow, Y.Bengio, A. Courville : Deep Learning, MIT Press, ISBN: 978-0262035613, 2016
- Mitchell, Tom: Machine Learning. McGraw-Hill, 1997
- Zell, Andreas: Simulation Neuronaler Netze. Oldenbourg Verlag, München, 1997

Geschäftsprozessmanagement (GPM)

Name	Geschäftsprozessmanagement (GPM)
Kürzel	GPM
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Michael Gröschel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Nebenfach - Grundlagen der BWL (NF)

Inhalte

- Introduction to Business Process Management
- Business Process Model and Notation (BPMN) - Modeling Foundations
- Analyzing Process Behaviour
- BPMN - Advanced Business Process Modeling
- Data in Business Processes
- Process Choreographies
- Decision Modeling with the Decision Model and Notation (DMN) - optional
- Student projects

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- prozessorientiert zu denken.
- Geschäftsprozesse zu modellieren, zu analysieren, und zu optimieren.
- Bezüge zwischen Geschäftsprozessmanagement und den Methoden des Software Engineerings sowie geeigneten IT-Architekturen herzustellen.
- ausgewählte Themen des Geschäftsprozessmanagements gemäß des Konzeptes "Lernen durch Lehren" im Plenum vorzustellen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	1 SWS
Projekt	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	15
Projektarbeit	60
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Michael Gröschel

Literatur

- Weilkiens, T./ Weiss, C./ Grass, A./ Duggen, K.N.: Basiswissen Geschäftsprozessmanagement, Aus- und Weiterbildung zum OMG Certified Expert in Business Process Management 2 (OCEB 2) - Fundamental Level, 2. Aufl, Heidelberg 2015
- Freund, J./ Rücker, B.: Praxishandbuch BPMN, 5. Aufl., München 2017
- Gadatsch, A.: Grundkurs Geschäftsprozess-Management: Methoden und Werkzeuge für die IT-Praxis: Eine Einführung für Studenten und Praktiker, 7. Aufl., Wiesbaden 2012
- Gadatsch, A.: Geschäftsprozesse analysieren und optimieren: Praxistools zur Analyse, Optimierung und Controlling von Arbeitsabläufen, Wiesbaden 2015
- Allweyer, T.: BPMN 2.0 - Business Process Model and Notation, 2. Aufl., Norderstedt 2009
- Staud, J.: Geschäftsprozessanalyse, 3. Aufl., Heidelberg 2006
- Silver, B.: BPMN Method and Style, 2. Aufl., Aptos 2012
- Dumas, M./ La Rosa, M./ Mendling, J. et al.: Fundamentals of Business Process Management, Heidelberg 2013
- Weske, M.: Business Process Management - Concepts, Languages, Architectures, 2. Aufl, Heidelberg 2012
- Becker, J./ Kugeler, M./ Rosemann, M. (Hrsg.): Prozessmanagement - Ein Leitfaden zur prozessorientierten Organisationsgestaltung, 7. Aufl., Springer: Heidelberg 2012
- Silver, B.: DMN Method and Style: The Practitioner
- Debevoise, T./ Taylor, J.: The MicroGuide to Process and Decision Modeling in BPMN/DMN, 2014

Graphentheorie (GRA)

Name	Graphentheorie (GRA)
Kürzel	GRA
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Elena Fimmel
Dauer	1 Semester
Studiengänge	IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Grundbegriffe der Graphentheorie,
- Wege, Eulersche Kreise und offene Linien,
- Hamiltonkreise, bewertete Graphen, das Problem des Handlungsreisenden, kürzeste Wege,
- Algorithmen zur Suche kürzester Wege (u.a. der Dijkstra-Algorithmus)
- Bäume, Gerüste, Minimalgerüste, Algorithmen zur Suche Minimalgerüste (u.a. die Kruskal- und Prim-Algorithmen)
- Planare Graphen
- Matchingtheorie, Heiratsproblem
- Färbungen, Färbungsalgorithmen, Ramsey-Theory
- Gerichtete Graphen, Turniere
- Netzwerke
- Maximale Flüsse, minimale Schnitte, der Ford-Fulkerson-Algorithmus

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Einsatzgebiete und Grenzen der erlernten Verfahren zu erkennen und zu erläutern,
- einige graphentheoretische Algorithmen zu verstehen und ggf. zu implementieren und
- das erhaltene Ergebnis einem nicht eingearbeiteten Fachkollegen zu erklären und zu präsentieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Elena Fimmel

Literatur

- A. Steger: Diskrete Strukturen, Band 1, Springer, 2002
- Peter Hartmann: Mathematik für Informatiker, Vieweg, 2004
- A. Beutelspacher, M.-A. Zschiegner: Diskrete Mathematik für Einsteiger, Vieweg, 2002
- J.Clark, D.A. Holton: Graphentheorie, Spektrum Verlag, 1991

Interaction Design (IAD)

Name	Interaction Design (IAD)
Kürzel	IAD
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Kirstin Kohler
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Projektarbeit (PA)

Empfohlene Vorkenntnisse

Inhalte

- Ausgehend von der Idee des Projektes wird eine detailierte Benutzerführung und Gestaltungskonzept erarbeitet und das Projekt weitestgehend umgesetzt. Die Bandbreite der Projekte soll hoch sein und z. B. von der Erweiterung bestehender Systeme bis zu experimentellen, visionären oder technisch (noch) nicht umsetzbarer Ideen reichen. Die Projekte können in Teams aus I- und D-Studierenden durchgeführt werden. Ein Kursziel besteht darin, sich über die verschiedenen Projekte auszutauschen. Die Projektarbeit wird von einer Vorlesung begleitet, die aktuelle Entwicklungen im Bereich der interaktiven Medien aufgreift und in Bezug zu den Projekten stellt.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- aktuelle Entwicklungen und Methoden im Interaction Design zu benennen und ausgewählte anzuwenden

Semesterwochenstunden

Vorlesung	2 SWS
Projekt	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	20
Projektarbeit	70
Summe	**150**

Dozentinnen / Dozenten

- Prof. Hartmut Wöhlbier

Image Generation and Rendering (IGR)

Name	Image Generation and Rendering (IGR)
Kürzel	IGR
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Sommersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Gerd Marmitt
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Programmierung 1 & 2
- Mathematik für die Informatik 1 & 2
- Erfahrungen mit C++ von Vorteil

Inhalte

- Projektion und Bildersterstellung im 3D-Raum
- Grundlagen zu Ray Tracing und OpenGL
- Fortgeschrittene Rendering-Techniken: Fraktale, Partikelsysteme, Isoflächen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Orthographische und perspektivische Projektionen und Transformationen im 3D-Raum anwenden
- Computergenerierte Grafiken mittels Rasterisierung und Ray-Tracing zu erstellen
- Grundlegende Renderingeffekte (Shader)zu erstellen

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Gerd Marmitt

Literatur

- Shirley, P. and Morley, R.K.: Realistic Ray Tracing, A.K. Peters, 2nd Edition
- Shirley, P. et al.: Fundamentals of Computer Graphics, A.K. Peters, 2nd Edition
- Hansen, C.D. and Johnsen C.R.: The Visualiuation Handbook, Elsevier

iPhone App Development mit SwiftUI (IOS)

Name	iPhone App Development mit SwiftUI (IOS)
Kürzel	IOS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Miriam Föller-Nord
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Projektarbeit (PA)

Empfohlene Vorkenntnisse

- Programmiererfahrung (PR1, PR2), insbesondere objektorientierte Programmierung
- Erfahrung (als Nutzer) mit den Betriebssystemen MacOS X und IOS
- Kenntnisse (als Nutzer) über iCloud, und AppleID.

Inhalte

- Grundlagen App-Entwicklung
- Besonderheiten bei der Entwicklung von nativen Apps für IOS/MacOS/AppleWatch
- Swift vs. Objective C
- SwiftUI vs. UIKit und Storyboard
- Entwicklungswerkzeug: XCode
- Programmiersprache Swift und der Swift Playground
- Entwickeln und Testen von Apps mit Swift und SwiftUI
- Veröffentlichung von Apps im Apple App Store

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- native Apps für IOS, MacOS und AppleWatch mit SwiftUI zu entwickeln und im Apple App Store zu Veröffentlichen.

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	30
Projektarbeit	60
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Miriam Föller

Literatur

- Beginning SwiftUI for iOS 15: Build iOS apps with Xcode 13 (English Edition), Greg Lim, independently published
- Einstieg in SwiftUI: User Interfaces erstellen für macOS, iOS, watchOS und tvOS, Thomas Sillmann, Carl Hanser Verlag GmbH & Co. KG; 1. Edition (12. Oktober 2020)
- OS 15 Programming for Beginners: Kickstart your mobile app development journey by building iOS apps with Swift 5.5 and Xcode 13, 6th Edition, Englisch Ausgabe von Ahmad Sahar (Autor), Craig Clayton (Autor), Packt Publishing; 6. Edition (24. Dezember 2021)
- iOS Development with SwiftUI: Acquire the Knowledge and Skills to Create iOS Applications Using SwiftUI, Xcode 13, and UIKit (English Edition) Mukesh Sharma, BPB Publications; 1. Edition (13. Januar 2022)
- Beginning iPhone Development with SwiftUI, Exploring the iOS SDK Sixth Edition , Wally Wang, Apress

Internet of Things (IOT)

Name	Internet of Things (IOT)
Kürzel	IOT
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Referat (R)

Empfohlene Vorkenntnisse

Inhalte

- IoT-relevante Technologiekonzepte der Digitalisierung
- Produktivitätszuwachs durch IoT und Information
- Internet of Things (IoT), Industrie 4.0 und Industrial IoT
- Use Cases und Anwendungsszenarien des IoT in Produkten, Prozessen und Geschäftsmodellen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- den Begriff der Industrie 4.0 einzuordnen
- datengetriebene Geschäftsmodelle der Industrie 4.0 zu bestimmen und zu entwickeln
- Technologien und Architekturen des IoT zu benennen und zu erklären
- Geschäftsmodelle und Use Cases des IoT zu beschreiben und einzuordnen

Semesterwochenstunden

Vorlesung	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	40
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	80
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Christoph Sandbrink

Literatur

- Andelfinger, Volker P. et al.: Internet der Dinge; Springer Gabler 2015.
- Anderl, R., u. a. (Industrie 4.0, 2016): Industrie 4.0 grenzenlos, Berlin: Springer Vieweg, 2016.
- Höller, Jan et al.: From Machine-to-Machine to the Internet of Things; Elvesier 2015.
- Huber, Walter: Industrie 4.0 kompakt – Wie Technologien unsere Wirtschaft und unsere Unternehmen verändern, Springer Vieweg 2018.
- Kaufmann, Timothy : Geschäftsmodelle in Industrie 4.0 und dem Internet der Dinge; Springer Vieweg 2015.
- Mohamed, Khaled Salah : The Era of Internet of Things; Springer 2019.
- Neugebauer, Reimund: Digitalisierung; Springer Vieweg 2018.
- Specht, Philip: Die 50 wichtigsten Themen der Digitalisierung; Redline 2018.
- Tesch, Jan F. : Business Model Innovation in the Era of the Internet of Things; Springer 2017.

International Product Development Project (IPDP)

Name	International Product Development Project (IPDP)
Kürzel	IPDP
Semester	6/7
Unterrichtssprache	Englisch
Häufigkeit	Sommersemester
Kreditpunkte	10 ECTS
Modulverantwortlich	Prof. Kirstin Kohler
Dauer	1 Semester
Studiengänge	CSB, IB, IM, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- Design Thinking Bootcamp
- Selbständiges Bearbeiten der Challenge im internationalem Team aus zwei Hochschulen/Universitäten
- Wöchentliche Vermittlung von Wissen über u.a. User Research, Prototyping, User Testing, erfolgreich Teamarbeit
- Wöchentliche Fortschrittsberichte einschl. Diskussion der (Zwischen-)Ergebnisse
- Wöchentlicher Austausch mit dem Partnerteam
- Final Presentation
- Besuch der Partnerhochschule/Universität

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- typische Herausforderungen eines Wicked Problems zu erkennen und Probleme im Sinne des Design Thinking anzugehen, d. h. Nutzer-zentriert, Prototypen-basiert und hoch iterativ zu bearbeiten,
- Methoden des Design Thinkings einzusetzen, sowie Kenntnisse aus dem eigenen Fachgebiet in die Lösungserarbeitung selbständig einzubringen und zu vertiefen,
- in internationalen Teams kooperativ und verantwortlich zu arbeiten, sowie gemeinsam eine Lösung in Form eines „Proof-of-Concept“ Demonstrators zu entwickeln,
- das Ergebnis vor internationalem Publikum zu präsentieren sowie eine Konzeptdokumentation zu erstellen
- die Bedeutung von Wünschbarkeit, Machbarkeit und Wirtschaftlichkeit einer Lösung zu benennen,
- die eigenen Handlungen kritisch zu reflektieren und aus Fehlern zu lernen sowie Vertrauen in die Eigene Kreativität und Lösungskompetenz zu entwickeln,

- sich schnell und selbständig in neue und komplexe Domänen einzuarbeiten und unvorhergesehene Hürden aus dem Weg zu räumen.

Semesterwochenstunden

Projekt	2 SWS
Workshop	2SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Projektarbeit	270
Workshop	30
Summe	**300**

Dozentinnen / Dozenten

- Prof. Kirstin Kohler
- Prof. Dr. Peter Kaiser

Literatur

- Kees Dorst: Frame Innovation: Create New Thinking by Design (Design Thinking, Design Theory), MIT Press Ltd, 2015
- Nigel Cross: Designerly Ways of Knowing, Springer, 2010
- Michael Lewrich, Patrick Link, Larry Leifer: Das Design Thinking Playbook, 2. Auflage, Vahlen, 2018
- Michael Lewrich, Patrick Link, Larry Leifer: Das Design Thinking Toolbook, Vahlen, 2020

Kodierungstheorie (KDT)

Name	Kodierungstheorie (KDT)
Kürzel	KDT
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Lutz Strüngmann
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Es ist sinnvoll, folgende Voraussetzungen mitzubringen:

- Grundlagen der Veranstaltung Mathematik 2

Inhalte

- Die Kodierungstheorie untersucht Verfahren zur Konstruktion und Verwendung von fehlertoleranten Codes. Die Urväter der Kodierungstheorie sind vor allem Hamming, Golay und Shannon, die in den 1940er und 1950er Jahren fundamentale Arbeiten hierzu verfasst haben. Inzwischen ist die Kodierungstheorie zu einem stark verzweigten und wichtigem Gebiet geworden, dessen Anwendungen sich vor allem in der Nachrichten- und Speichertechnologie finden, aber auch in der Bildverabeitung, der Mustererkennung und natürlich der Kryptographie. Sogar in der Biologie findet die Kodierungstheorie ihren Platz. Offensichtlich spielen algorithmische Probleme eine große Rolle in der Kodierungstheorie. Effiziente Algorithmen zur Kodierung oder Dekodierung von leistungsstarken Codes sind ebenso wichtig wie die Codes selbst. Aus diesem Grunde hat die Kodierungstheorie zur Entwicklung zahlreicher sehr interessanter Algorithmen geführt. Zudem führen kodierungstheoretische Resultate zu überraschenden Anwendungen in der Komplexitätstheorie und in anderen Bereichen der Informatik.
- In dieser Veranstaltung sollen die Grundzüge der Kodierungstheorie, also die elementare Kodierungstheorie besprochen werden. Wir wollen eine Einführung in die Grundfragen und Grundlagen der Theorie der linearen Codes geben. Dazu werden wir die gängigen Klassen von Codes samt zugehöriger Decodieralgorithmen vorstellen. Zu diesen gehören u. a. Reed-Solomon-Codes, Hamming-Codes, Golay-Codes, BCH-Codes, quadratische Reste-Codes, Reed-Muller-Codes sowie die klassischen Goppa-Codes. Daneben sollen einige der asymptotischen Schranken für die Informationsrate von Codes bewiesen werden.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- mathematische Grundlagen der Kodierungstheorie zu verstehen und anzuwenden,
- (lineare) Codes zu analysieren,

- Codes bezüglich Fehlertoleranz und Fehlerkorrektur zu entwickeln und
- Anwendungen der Kodierungstheorie in der Informatik zu verstehen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Eigenstudium zur Vor- und Nachbereitung	60
Selbständiges Bearbeiten der Übungen	15
Prüfungsvorbereitung	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Lutz Strüngmann

Literatur

- W.C. Huffman and V. Pless. Fundamentals of Error-Correcting Codes, Cambridge, 2003.
- J.H. van Lint. Introduction to Coding Theory. Springer Verlag, 1999.
- Werner Lütkebohmert. Codierungstheorie. Vieweg Studium Verlag, 1. edition, 2003.
- MacWilliams, F.J. and Sloane, N.J.A. The Theory of Error-Correcting Codes. North- Holland, Amsterdam, 1977.
- D. Jungnickel, Codierungstheorie. Spektrum Akademischer Verlag, 1995.
- R.-H. Schulz, Codierungstheorie. Vieweg, 2003.

Künstliche Intelligenz für autonome Systeme (KIS)

Name	Künstliche Intelligenz für autonome Systeme (KIS)
Kürzel	KIS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Jörn Fischer
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Praktische Arbeit/Projektarbeit (PA)
Prüfungsleistung	Referat (R)

Empfohlene Vorkenntnisse

- Lineare Algebra, Analysis, MLE von Vorteil

Inhalte

- Methoden der Künstlichen Intelligenz
- Planungsalgorithmen
- Evolutionäre Algorithmen in der Simulation
- Maschinelles Lernen in Real-Welt Szenarien
- Neuronale Netze für Bilderkennung, Sprachverarbeitung, zur Steuerung von Bewegungsabläufen etc.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Methoden der Künstlichen Intelligenz zu verstehen
- Planungsalgorithmen zu verstehen
- Evolutionäre Algorithmen zu implementieren
- Machine Learning Ansätze für die reale Welt zu entwickeln
- Neuronale Netze für unterschiedliche Anwendungsszenarien zu implementieren

Semesterwochenstunden

Vorlesung	2 SWS
Projekt	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Projektarbeit	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jörn Fischer
- Prof. Dr. Thomas Ihme

Literatur

- S. Russell, P. Norvig, Artificial Intelligence A modern approach, ISBN: 978-0132071482, 2010
- I. Goodfellow, Y.Bengio, A. Courville : Deep Learning, MIT Press, ISBN: 978-0262035613, 2016
- Mitchell, Tom: Machine Learning. McGraw-Hill, 1997
- Zell, Andreas: Simulation Neuronaler Netze. Oldenbourg Verlag, München, 1997

Kommunikationssysteme (KOS)

Name	Kommunikationssysteme (KOS)
Kürzel	KOS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Miriam Föller-Nord
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Grundlagen und Konzepte von Kommunikationssystemen
- Topologie
- Protokolle
- TCP/IP-Stack
- Internet
- LAN, WAN
- Sicherheit in Rechnernetzen
- Socketprogrammierung

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die wichtigsten Konzepte und Technologien im Umfeld von Computernetzen zu verstehen und zu erläutern,
- Einsatzgebiete und Grenzen der Techniken zu erläutern und
- erlernte Konzepte auf neue Anwendungsbereiche zu übertragen.

Semesterwochenstunden

Vorlesung	3 SWS
Übung	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	15
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Miriam Föller-Nord

Literatur

- Kurose, Ross; Computernetze; Pearson Studium
- Peterson, Davie; Computernetze; dPunkt-Verlag
- Sikora; Technische Grundlagen der Rechnerkommunikation; Fachbuchverlag Leipzig

Kommerzialisierung von Open-Source-Software (KOSS)

Name	Kommerzialisierung von Open-Source-Software (KOSS)
Kürzel	KOSS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Oliver Hummel
Dauer	1 Semester
Studiengänge	CSB, IB, IM, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Empfohlene Vorkenntnisse

- Programmierung 1 & 2 (PR 1 & PR2)
- Software Engineering 2 (SE2)
- Webbasierte Systeme (WEB)

Inhalte

- Geschäftsmodelle im Umfeld von Open-Source-Software (OSS)
- Open-Source-Lizenzen
- Fallstudien erfolgreicher Unternehmen im OSS-Umfeld
- Code-Reading
- Installation, Konfiguration und Erweiterung eines Beispielsystems
- "Productizing" und Qualitätssicherung der Software
- Umgang mit der Community
- Markteintritt planen
- Ausschreibungen verstehen und gewinnen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- zu erklären, welche kommerziellen Nutzungen verschiedene Open-Source-Lizenzen erlauben
- Open-Source-Software hinsichtlich ihrer kommerziellen Anwendbarkeit zu analysieren und bewerten
- Geschäftsmodelle für unterschiedliche OS-Applikationen zu entwickeln

- Tools zum Bauen der Software aus Sourcecode zu verwenden
- die Software selbst zu installieren und zu konfigurieren
- grundlegende Sicherheitsaspekte zu berücksichtigen
- fehlende Dokumentation durch Code-Analyse zu kompensieren
- die Software gemeinsam mit der Community weiterzuentwickeln
- ein Geschäftsmodell prototypisch umzusetzen

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	50
Selbständiges Bearbeiten der Übungen	40
Summe	**150**

Dozentinnen / Dozenten

- Dr. Christoph Giess
- Philipp Bletzer

Literatur

- How Open Source Ate Software, Gordon Haff, Apress 2021
- Open Source Jahrbuch, http://www.opensourcejahrbuch.de
- Linux- und Open-Source-Strategien für CIOs, BerleconResearch, 2004
- Rechtsfragen der Open Source Software, Prof. Dr. Gerald Spindler, Universität Göttingen, 2003
- Ein Leitfaden für kleine und mittlere Unternehmen, BMWi, 2001
- The Cathedral and the Bazaar, Eric S. Raymond, 1999
- The Architecture of Open Source Applications, http://aosabook.org

Kryptographische Verfahren (KRV)

Name	Kryptographische Verfahren (KRV)
Kürzel	KRV
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Elena Fimmel
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Einführung in die Höhere Mathematik

Inhalte

- Geschichte und Ziele der Kryptographie
- Zahlentheoretische Grundlagen
- Klassische Kryptographie (symmetrische Verfahren):
- Transpositionschiffren,
- Substitutionschiffren,
- Monoalphabetische Verschlüsselung,
- Polyalphabetische Verschlüsselung, Entschlüsselung der Enigma
- Moderne Kryptographie:
- Funktionsweise des DES (Data Encryption Standard),
- Vorstellung vom AES (Advanced Encryption Standard)
- Asymmetrische (Public-Key) Kryptographie:
- Diffie-Hellman-Schlüsselaustauschprotokol, RSA
- Digitale Unterschriften
- Verschlüsselung eines Dokuments mit PGP

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Funktionsweise symmetrischer Kryptosysteme, u.a. DES und AES zu verstehen,

- die Funktionsweise asymmetrischer Kryptosysteme (Public-Key-Verfahren) zu verstehen und sie (wie z. B. bei digitalen Unterschriften) anzuwenden und
- das erhaltene Ergebnis einem nicht eingearbeiteten Fachkollegen zu erklären und zu präsentieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Elena Fimmel

Literatur

- Johannes Buchmann: Einführung in die Kryptographie. Springer, 2004
- Simon Singh: Codes. Hanser, 2002
- A. Beutelspacher, M.-A. Zschiegner: Diskrete Mathematik für Einsteiger. Vieweg, 2002
- Wolfgang Ertel: Angewandte Kryptographie. Fachbuchverlag Leipzig, 2001

Large-Scale (Software) Development (LSD)

Name	Large-Scale (Software) Development (LSD)
Kürzel	LSD
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Software Engineering 1 (SE1)
- Software Engineering 2 (SE2)

Inhalte

- Was sind große Systeme
- Umgang mit Legacy-Systemen und großen Codebasen
- Analyse
- Reeengineering
- Refactoring
- Wrapping
- Virtualisierung
- Configuration Management
- Code-Linien/Branches
- Merging
- Configuration as Code
- Produktion von Software
- Build-Werkzeuge (Ant, Maven, gnumake)
- Continuous Intergration
- Continuous Delivery

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Komplexität bei der Entwicklung großer Softwaresysteme zu verstehen und zu erklären.
- sich in großen Codebasen zurecht zu finden, diese zu strukturieren und zu verbessern,
- moderne Werkzeuge der Software-Entwicklung einzusetzen und
- große Systeme weiterzuentwickeln.

Semesterwochenstunden

Vorlesung	2 SWS
Projekt	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Thomas Smits

Literatur

- Vincent Maraia, The Build Master, Addison-Wesley Professional, 2005
- Eberhard Wolff, Continuous Delivery: Der pragmatische Einstieg, 2. Auflage, dpunkt.verlag, 2016
- Adrian Mouat, Docker: Software entwickeln und deployen mit Containern, dpunkt.verlag, 2016
- René Preißel, Git: Dezentrale Versionsverwaltung im Team, dpunkt.verlag, 2017

Mathematische Biologie (MBI)

Name	Mathematische Biologie (MBI)
Kürzel	MBI
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Lutz Strüngmann
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Es ist sinnvoll, wenn Sie die Basisveranstaltungen in Mathematik besucht haben bzw. äquivalentes Vorwissen in Mathematik besitzen. (It is reasonable if you have taken part in the basic courses on mathematics or if you have equivalent knowledge in mathemat

- Mathematik für die Informatik 1 (MA1)
- Mathematik für die Informatik 2 (MA2)
- Mathematik für die Informatik 3 (MA3)

Inhalte

- In der Mathematischen Biologie (oder vielleicht Biomathematik? Oder Mathematik für Biologen ?) beschäftigen wir uns mit der Modellierung biologischer Prozesse und Phänomene mittels formaler Methoden aus der Mathematik. Insbesondere werden Theorien erarbeitet, die lebende Systeme samt ihrer Dynamik und strukturellen Eigenschaften zu erfassen und abzubilden suchen. Beispiele sind etwa die Entwicklung von Populationen, die zum Beispiel im Falle von Hasen mit den sogenannten Fibonacci Zahlen beschrieben werden kann. Auch der genetische Code und die Evolution bieten zahlreiche Möglichkeiten zur Modellierung. Um solche Prozesse zu beschreiben bedient sich die Mathematische Biologie neben Methoden aus dem Gebiet der Dynamischen Systeme wie etwa Differentialgleichungen, auch der Gruppentheorie und Kombinatorik. In vielen Bereichen besteht dabei auch ein enger Zusamenhang zur Bioinformatik und es werden Werkzeuge aus der Diskreten Mathematik benutzt. In der Vorlesung werden einige der Grundlagen der Mathematischen Biologie sowie viele einfache Modelle biologischer Prozesse besprochen und in Beispielen selbstständig angewendet. Ein wesentlicher Aspekt ist dabei, wie die Parameter der entwickelten Modelle möglichst gut zu wählen sind, um den Prozess bestmöglich abzubilden.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- selbstständig die Grundlagen der mathematischen Modellierung von biologischen Prozessen anzuwenden.

Semesterwochenstunden

Vorlesung	3 SWS
Übung	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	50
Präsenzübungen und Testate	10
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Professor Dr. Lutz Strüngmann

Literatur

- Britton, N. F., Essential mathematical biology, Springer (2003)
- Edelstein-Keshet, L., Mathematical models in biology, reprint, SIAM (2005)
- Keener, J., Sneyd, J., Mathematical Physiology I & II, (2 Bde.), 2nd ed., Springer (2009)
- Murray, J. D., Mathematical biology I & II, (2 Bde.), Springer (2002/3)
- Shonkwiler, R. W., Herod, J., Mathematical biology (An introduction with Maple and Matlab, 2nd ed., Springer (2009)
- Fall, C. F., Computational cell biology, Springer (2002)
- Brauer, F., Castillo-Chavez, C., Mathematical models in population biology and epidemiology, 2nd ed., Springer (2012)
- Bohl, E., Mathematik in der Biologie, 4. Auflage, Springer (2006)

Microcomputing and Embedded Development (MCP)

Name	Microcomputing and Embedded Development (MCP)
Kürzel	MCP
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Miriam Föller-Nord
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Beherrschen der im Modul Hardwarenahe Programmierung vermittelten Inhalte

Inhalte

- Einführung Mikroprozessoren und Mikrocontroller
- Einführung eines speziellen Mikrocontrollers, z. B. C167
- Entwicklungsumgebung für Mikrocontroller
- Speicherorganisation eines MC
- On-Chip-Peripherieeinheiten
- A/D-Wandler
- I/O-Ports
- Interrupt-Handling
- Timer
- Pulsweitenmodulation
- Anschluss externer Peripherieeinheiten

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die wichtigsten Grundkonzepte beim Entwickeln von Software auf Microcontroller-Basis zu verstehen und zu erläutern,
- Software-Designentscheidungen für eingebettete Systeme zu treffen und
- einen geeigneten Microcontroller für eine gegebene Aufgabenstellung auszuwählen und zu programmieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	25
Eigenstudium zur Vor- und Nachbereitung	40
Selbständiges Bearbeiten der Übungen	30
Prüfungsvorbereitung	25
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Miriam Föller-Nord

Literatur

- Brinkschulte, Ungerer; Microcontroller und Mikroprozessoren; Springer-Verlag; 2002
- Bähring; Mikrorechner-Technik Band I und II; Springer-Verlag; 2002
- Simon; An Embedded Software Primer; Addison-Wesley

Mobile und drahtlose Informationsverarbeitung (MDI)

Name	Mobile und drahtlose Informationsverarbeitung (MDI)
Kürzel	MDI
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Miriam Föller-Nord
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Beherrschen der in den Modulen Grundlagen der Informatik, Algorithmen und Datenstrukturen, Objektorientierte Techniken und Techniken der Programmentwicklung vermittelten Inhalte
- Die Vorlesung IPhone-Programmierung kann gleichzeitig gehört werden

Inhalte

- Mobile Endgeräte, Plattformen und Anwendungen
- Protocol Stack
- Sicherheit in mobilen Szenarien
- Location Based Services
- Übertragungstechnische Grundlagen
- Programmiertechniken mit einer geeigneten Programmiersprache

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Technologien der drahtlosen Kommunikation zu verstehen und zu erläutern,
- aktuelle Plattformen zur Software-Entwicklung im Bereich Mobile Computing zu bewerten,
- passende Plattformen bei einer neu zu entwickelnden Applikation auszuwählen und
- erlernte Konzepte auf neue Anwendungsbereiche zu übertragen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Miriam Föller-Nord

Literatur

- J. F. Kurose/K. W. Ross, Computernetze
- J. Roth, Mobile Computing
- J. Schiller, Mobilkommunikation
- U. Karrenberg, Signale Prozesse Systeme

Maschinelles Lernen (MLE)

Name	Maschinelles Lernen (MLE)
Kürzel	MLE
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Wintersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Jörn Fischer
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Praktische Arbeit/Projektarbeit (PA)
Prüfungsleistung	Klausur 60 Minuten (K60)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Optimierung
- Evolutionäre Algorithmen
- Reinforcement Learning
- Neuronale Netze

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- grundlegende Optimierungsstrategien anzuwenden,
- Genetische Algorithmen zu erläutern,
- Genetische Programmierung zu implementieren und zu verstehen,
- Reinforcement Learning anzuwenden und
- künstliche Neuronale Netze zu erläutern.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jörn Fischer

Literatur

- I. Goodfellow, Y.Bengio, A. Courville : Deep Learning, MIT Press, ISBN: 978-0262035613, 2016
- Russel, Stuard; Norvig, Peter: Künstliche Intelligenz. Prentice Hall, New Jersey, 1995
- Mitchell, Tom: Machine Learning. McGraw-Hill, 1997
- Zell, Andreas: Simulation Neuronaler Netze. Oldenbourg Verlag, München, 1997
- Sutton, Richard; Barto, Andrew G.: Reinforcement Learning. MIT Press, 1998

Mobile Programmierung mit Fuchsia (MOP)

Name	Mobile Programmierung mit Fuchsia (MOP)
Kürzel	MOP
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Sandro Leuchter
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Software Engineering 1 (SE1)
- Software Engineering 2 (SE2)
- Webbasierte Systeme (WEB)

Inhalte

- Am Beispiel des neuen vollkommen modular aufgebauten Betriebssystems Fuchsia werden moderne Konzepte der mobilen Programmierung analysiert.
- Neue Betriebssystem-Konzepte von Zircon/Garnet/Peridot/Topaz: z. B. Noun, Verb, Module, Agent, Component, Package, Story, Ledger
- Sicherheitsarchitektur von Zircon/Garnet/Peridot/Topaz
- Interprozesskommunikation mit FIDL
- Systemnahe Programmierung von Kommandozeilenanwendungen mit Dart 2, z. B. für embedded systems
- Cross-Plattform-Entwicklung mit dem UI-Framework Flutter und Dart 2 (neben Fuchsia laufen solche Anwendungen auf auf IOS und Android)
- Entwicklung mit Fuchsia als VM und auf realer Hardware (nach Möglichkeit)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- neuartige Betriebssystemkonzepte von Fuchsia zu nennen

- zu beurteilen wie neuartige Betriebssystemkonzepte von Fuchsia die Software- und Systemarchitektur von Anwendungen beeinflussen
- mobile Anwendungen mit Flutter und Dart 2 für Fuchsia, Android und IOS zu entwickeln
- embedded Anwendungen mit Dart 2 und Fuchsia zu entwickeln

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	30
Projektarbeit	60
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr.-Ing. Sandro Leuchter

Literatur

- Google (2018): Fuchsia is not Linux. https://fuchsia.googlesource.com/fuchsia/+/master/docs

Natural Language Processing (NLP)

Name	Natural Language Processing (NLP)
Kürzel	NLP
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Kai Eckert
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- In diesem Kurs lernen Sie die NLP-Grundlagen kennen:
- Reguläre Ausdrücke, Finite State Automaten, N-Grams, Probabilistic Language Modeling, Vector Space Model, TF-IDF, Machine Learning for NLP, Sequence Labeling, Part of speech tagging, Information Extraction, Named Entity Recognition

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- grundlegende Techniken des Natural Language Processing zu verstehen und anzuwenden.
- Zusammen mit Grundlagen für Neuronale Netze und Deep Learning ist diese Veranstaltung die Grundlage für fortgeschrittene Techniken in NLP, die in darauf aufbauenden Veranstaltungen (bzw. im Master) gehört werden können.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	30
Prüfungsvorbereitung	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Kai Eckert

Einführung in die Netzwerkforensik (NWF)

Name	Einführung in die Netzwerkforensik (NWF)
Kürzel	NWF
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Wintersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Jessica Steinberger
Dauer	1 Semester
Studiengänge	CSB, IB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Hausarbeit (HA)

Empfohlene Vorkenntnisse

- Kenntnisse über digitale Zahlendarstellungen und Kodierungen (z. B. ASCII)
- Grundlegende Programmierkenntnisse

Inhalte

- Vorgehensmodell des „Network Security Monitoring“
- Hashfunktionen in der digitalen Forensik
- Grundlagen der Netzwerkprotokolle und deren Angriffsvektoren
- Arten von Netzwerkangriffen und deren Auswirkungen im Netzwerkdatenverkehr
- Einblicke in Datengewinnung aus verschiedenen Netzwerkkomponenten
- IT-forensische Analyse von Netzwerkdatenverkehr
- Umgang mit Werkzeugen der Netzwerkforensik
- Visualisierung von Netzwerkdaten
- Gutachtenerstellung

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- das Vorgehensmodell des „Network Security Monitoring“ zu erklären und können die beschriebene Vorgehensweise anwenden.
- die grundlegenden Funktionsweise kryptographischer Hashfunktionen zu benennen und sind mit deren Bedeutung in der Netzwerkforensik vertraut.
- verschiedene Netzwerkmitschnittdateitypen zu erzeugen und relevante Daten zu extrahieren.

- verschiedene Netzwerkangriffe zu benennen, deren Funktionsweise zu erklären und im Netzwerkdatenverkehr zu erkennen.
- Datenströme zu analysieren sowie nachzuverfolgen.
- wichtige Dienste wie E-Mail, HTTP, HTTPS, DNS zu beschreiben und deren Nutzung zu analysieren.
- gängige forensischen Tools zu benennen und wichtige Ergebnisse mit deren Hilfe eigenständig zu entnehmen.
- Visualisierung von Netzwerkdaten für ein forensisches Gutachten anzufertigen, kennen die unterschiedlichen Diagrammtypen und können Informationen aus den Grafiken für ein forensisches Gutachten extrahieren.
- die Herausforderungen der Netzwerkforensik zu benennen und können diese erklären.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Jessica Steinberger

Netzwerksicherheit (NWS)

Name	Netzwerksicherheit (NWS)
Kürzel	NWS
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Sommersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Referat (R)

Empfohlene Vorkenntnisse

Inhalte

- Grundlagen der Netzwerk-Sicherheit basierend auf der selbständigen Vorbereitung durch ausgewählte Kapitel der vorlesungsbegleitenden Literatur (Zisler)
- Firewalls, Netzwerksegmentierung
- Netzwerkbasierte Authentifizierung
- Sicherheitsprotokolle in Netzwerken (TLS, IPSec, DNSSEC,...)
- Deep Packet Inspection
- Endpoint Security
- Vulnerability Management
- Referat: IT-Security Audit eines mittelständischen Unternehmens mit ausführlichem Ergebnisbericht

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- eine ordentliche bzw. auf dem „Stand der Technik“ befindliche IT-Security Infrastruktur zu erkennen und zu bewerten,
- technische Elemente der IT-Security zu verstehen und anzupassen,
- Sicherheitsprotokolle zu bewerten,
- Firewalls einzurichten und anzupassen,
- Geräte im Netzwerk zu schützen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	80
Selbständiges Bearbeiten der Übungen	10
Summe	**150**

Dozentinnen / Dozenten

- Marco Rossi

Literatur

- Zisler, Harald: Computer-Netzwerke – Grundlagen, Funktionsweise, Anwendung. Rheinwerk Computing, 7. Auflage
- Schäfer, Roßberg: Netzsicherheit: - Grundlagen & Protokolle - Mobile & drahtlose Kommunikation - Schutz von Kommunikationsinfrastrukturen. dpunkt.verlag

Penetration Testing (PET)

Name	Penetration Testing (PET)
Kürzel	PET
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Die Studierenden sollten schon eine der folgenden Lehrveranstaltungen gehört haben:

- Security Management
- Sichere Internet-Dienste
- Sichere Software-Entwicklung
- Hacking-Workshop

Inhalte

- Linux und die Konsole für Hacker
- Informationsermittlung zur Vorbereitung eines Angriffs (aktive und passive Werkzeuge)
- Verschiedene Typen von Ausnutzungstechniken (SQLi, file upload, priv esc, RCE, directory traversal,...)
- netcat und reverse shell
- Verschiedene Pen-Testing Softwareumgebungen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- gängige Methodologien des Pentests und seine Terminologie zu beherrschen,
- eine Hacking-Toolbox, z. B. Kali Linux routiniert zu benutzen,
- die Unterschiede zwischen passiver und aktiver Informationsbeschaffung zu kennen und sowohl die eine als auch die andere durchführen zu können
- gängige Angriffsmethodiken sowohl theoretisch als auch praktisch zu beherrschen und darlegen zu können, welche fehlerhaften Programmiermuster den entsprechenden Angriffsvektor erzeugen

- die Bausteine vom Einstiegspunkt bis zum Endziel passend zusammenzusetzen (z. B. SQLi - ShellUpload - SSH tunnel - interner Scan - Domain Admin),
- einen professionellen Pentestbericht zu schreiben.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	60
Eigenstudium zur Vor- und Nachbereitung	30
Workshop	30
Summe	**150**

Dozentinnen / Dozenten

- Kai Ullrich

Literatur

- Kumar Velu: Mastering Kali Linux for Advanced Penetration Testing. Packt Publishing
- Allsopp: Advanced Penetration Testing: Hacking the World's Most Secure Networks. Wiley.
- Kim: The Hacker Playbook: Practical Guide To Penetration Testing. CreateSpace Independent Publishing Platform.

Prototype It Yourself (PIY)

Name	Prototype It Yourself (PIY)
Kürzel	PIY
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Sommersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Kirstin Kohler
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- Realisierung und Validierung eigener Projektideen in Form von digitalen / physikalischen Prototypen
- Anwendung von Methoden des Design Thinkings
- Anwendung von Geräten zur digitalen Fabrikation
- Validierung von Varianten ihrer Ideen mittels Rapid-Prototyping
- Präsentation des entwickelten Demonstrators

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Methoden des Design Thinkings einzusetzen
- Geräte zur digitalen Fabrikation zu bedienen / Making
- in interdisziplinären Teams kooperativ zu arbeiten
- vage Ideen in anfassbare Prototypen zu überführen (kreatives Selbstvertrauen, Selbstwirksamkeit)
- mit Hilfe von Prototypen Ideen schnell zu validieren (Resilienz, positive Fehlerkultur)
- das Ergebnis vor Publikum zu präsentieren (Präsentationsskills)
- der Kurs unterstützt die Entwicklung von 21-Century Skills / Future Skills / Entrepreneurial Skills

Semesterwochenstunden

Vorlesung	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Kirstin Kohler

Reverse Engineering (REE)

Name	Reverse Engineering (REE)
Kürzel	REE
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Grundlegende Kenntnisse in der C-Programmierung

Inhalte

- Ziele des Reverse Engineerings
- Statische Analyse
- Dynamische Analyse
- Binäre Java Programme dekompilieren und modifizieren
- Grundlagen zu x86_64 Assembler
- Einsatz gängiger Werkzeuge des Reverse-Engineerings (radare2, Ghidra, etc.)
- Native Programme disassemblieren und dekompilieren
- Analyse von unbekannten Executables und Malware
- Patchen von nativen Programmen
- CrackMe und Capture the Flag (CTF) lösen
- Anti-Reversing-Strategien und Gegenmaßnahmen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- gängige Werkzeuge des Reverse-Engineerings einzusetzen
- unbekannte Binärdateien zu untersuchen und deren Funktion einzuschätzen

- Binärdateien zu patchen, um deren Verhalten zu ändern
- Crackmes und Capture-the-flag (CTF) zu lösen

Semesterwochenstunden

Vorlesung	1 SWS
Übung	3 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Thomas Smits

Literatur

- Sikorski, M. and Honig, A. (2012). Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software. No Starch Press.
- Andriesse, D. (2018). Practical Binary Analysis: Build Your Own Linux Tools for Binary Instrumentation, Analysis, and Disassembly. No Starch Press.
- Eagle C. and Nance K. (2020). The Ghidra Book: The Definitive Guide. No Starch Press.
- Yurichev, D. (2019). Reverse Engineering for Beginners. (https://beginners.re/main.html)

Robotik (ROB)

Name	Robotik (ROB)
Kürzel	ROB
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Ihme
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Referat (R)
Prüfungsleistung	Hausarbeit (HA)

Empfohlene Vorkenntnisse

- aufgrund der hohen Interdisziplinarität ist vorherige oder gleichzeitige Hören der Module WVG, MLE, BIV, SIM und Analysis vorteilhaft
- Technische Informatik 1 (TEI1)
- Technische Informatik 2 (TEI2)
- Mathematik für die Informatik 1 (MA1)
- Mathematik für die Informatik 2 (MA2)
- Mathematik für die Informatik 3 (MA3)
- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Programmierung 3 (PR3)

Inhalte

- Überblick zu aktuellen Entwicklungen
- Teilsysteme von Robotern
- Transformationen
- Kinematik / Dynamik
- Sensoren und Sensorsignalverarbeitung
- ausgewählte Steuerungskonzepte und -systeme
- praktische Arbeit mit mobilen Kleinrobotern, Implementierung von einfachen Steuerungsalgorithmen

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- den grundlegenden Aufbau von Robotern zu erläutern,
- Konzepte zur Steuerung von Robotern zu beschreiben,
- das Zusammenwirken der Teilsysteme, insbesondere Zusammenhänge zwischen Hardware (Mechanik, Elektrisches System) und Software zu erläutern und
- Konzepte zur Steuerung auf einfache Roboter anzuwenden.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Thomas Ihme

Literatur

- Siegert, H.-J.; Bocinek, S.: Robotik: Programmierung intelligenter Roboter, Springer-Verlag
- Wloka, D. W.: Robotersysteme I: Technische Grundlagen. Springer-Verlag
- Everett, H. R.: Sensors for Mobile Robots. Theory and Praxis. A. K. Peters Ltd.
- Dillmann, R.; Huck, M.: Informationsverarbeitung in der Robotik. Springer-Verlag
- Knieriemen, T.: Autonome Mobile Roboter. Sensordateninterpretation und Weltmodellierung zur Navigation in unbekannter Umgebung. In Reihe: Böhling, K. H.; Kulisch, U.; Maurer, H. (Hrsg.): Reihe Informatik, Bd 80. BI Wissenschaftsverlag
- Hoppen, P.: Autonome Mobile Roboter. Echtzeitnavigation in bekannter und unbekannter Umgebung. In Reihe: Böhling, K. H.; Kulisch, U.; Maurer, H. (Hrsg.): Reihe Informatik, Bd 87. BI Wissenschaftsverlag
- Snyder, W. E.: Computergesteuerte Industrieroboter. Grundlagen und Einsatz., VCH
- Weber, W.: Industrieroboter. Methoden zur Steuerung und Regelung. Fachbuchverlag Leipzig
- Artikel aus aktuellen Konferenzen/Zeitschriftenbeiträge

Skalierbare Systemarchitekturen für IoT und Cloud-Computing (SAC)

Name	Skalierbare Systemarchitekturen für IoT und Cloud-Computing (SAC)
Kürzel	SAC
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Sandro Leuchter
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Pflichtübung (PU)
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

- Programmierung 1 (PR1)
- Programmierung 2 (PR2)
- Software Engineering 1 (SE1)
- Softwareprojekt (SP)

Inhalte

- Rechnernetze, Netzwerkdienste, Internet-Adressierung (IP, DNS, DHCP, UDP, TCP), Socket-Programmierung
- Serialisierungsformate (z. B. XML, JSON, BSON, ProtocolBuffers)
- Nachrichtenverteilung und Message Broker (z. B. JMS (ActiveMQ), MQTT (Paho), Cluster, JGroups, DDS, Kafka, Akka)
- Aufruf verteilter Unterprogramme (z. B. gRPC, Java RMI, CORBA)
- Webservices (z. B. XML-RPC, SOAP, JAX-WS, REST, JAX-RS, Swagger, CoAP)
- Microservices, Resource Injection, JNDI, Spring Boot, Docker, Service Discovery, Serverless Architekturen
- (es ist nur entweder SAC oder VS anrechenbar)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die systemarchitekturellen Anforderungen an verteilte Anwendungen im Bereich Internet of Things und Cloud Computing zu analysieren
- geeignete Architekturen für verteilte Anwendungen im Bereich Internet of Things und Cloud Computing auszuwählen und zu bewerten
- eine geeignete Middleware und Ablaufumgebung auszuwählen

- verteilte Anwendungen im Bereich Internet of Things und Cloud Computing zu entwerfen, zu implementieren und in Betrieb zu nehmen

Semesterwochenstunden

Vorlesung	2 SWS
Labor	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Präsenzübungen und Testate	30
Eigenstudium zur Vor- und Nachbereitung	60
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr.-Ing. Sandro Leuchter

Literatur

- Leuchter, S. (2017). Verteilte Architekturen. Eine praktische Einführung mit Java. Band 1: Internet-Kommunikation.
- Leuchter, S. (2018). Verteilte Architekturen. Eine praktische Einführung mit Java. Band 2: Aufruf verteilter Unterprogramme.

Scientific Computing (SCO)

Name	Scientific Computing (SCO)
Kürzel	SCO
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Sommersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Markus Gumbel
Dauer	1 Semester
Studiengänge	IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Keine besonderen Vorkenntnisse. Interesse an der Programmentwicklung für naturwissenschaftlich und technische Fragestellungen, beispielsweise Spieleprogrammierung.

Inhalte

- Einführung in Scientific Computing (u.a. Fließkommazahlen und Rundungsfehler)
- Sprachen und Entwicklungsumgebungen: R/RStudio, NetLogo (Matlab, Python, jupyter o.ä.)
- Ausgewählte nummerische Methoden (Lösen von Gleichungssystemen, Interpolation, Integrieren,); Pseudo-Zufallszahlen
- Simulationsmethoden (mittels Differentialgleichungen, ereignis-orientiert, agenten-basiert; zelluläre Automaten) incl. Parameteroptimierung
- Hochleistungsrechner und Cloud-Computing für naturwissenschaftlich-technische Anwendungen
- Zahlreiche Anwendungsbeispiele aus verschiedenen Domänen (Wirtschaft, Technik, Medizin uvm.)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- grundlegende technische Probleme in einer Programmiersprache zu formulieren und zu implementieren.
- komplexe technische oder wissenschaftliche Zusammenhänge zu beschreiben.
- die Komplelixtät der technischen Anwendungen, die in der Industrie zum Einsatz kommen, einzuschätzen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	1 SWS
Labor	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	40
Präsenzübungen und Testate	25
Eigenstudium zur Vor- und Nachbereitung	40
Selbständiges Bearbeiten der Übungen	25
Prüfungsvorbereitung	20
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Markus Gumbel

Literatur

- Hans-Joachim Bungartz, Stefan Zimmer, Martin Buchholz, Dirk Pflüger. Modeling and Simulation: An application-oriented introduction. Springer, 2014.
- Ulrich Hedtstück. Simulation diskreter Prozesse. Methoden und Anwendungen, Berlin Heidelberg, 2013.
- Uri Wilensky and William Rand. An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo. MIT Press, 2015.
- Thomas Huckle and Stefan Schneider. Numerische Methoden: Eine Einführung für Informatiker, Naturwissenschaftler, Ingenieure und Mathematiker. Springer, 2 Edition, 2006.

Suchmaschinenoptimierung und Social Media Marketing (SEO)

Name	Suchmaschinenoptimierung und Social Media Marketing (SEO)
Kürzel	SEO
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Frank Dopatka
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Voraussetzungen gemäß RGS zum Absolvieren eines Wahlfachs. Vertiefte Kenntnisse in den folgenden Bereichen sind von Vorteil:

- Webbasierte Systeme (WEB)

Inhalte

- OnPage Optimierung mit Keyword-Analyse, Strukturierung, strukturierten Daten, Konversionsrate und Server-Geschwindigkeit
- OffPage Optimierung mit Pagerank, Backlinks und Link-Building, RSS-Newsfeed, Blog und Social Signals, Google Analytics, AdWords und DSGVO
- YouTube Kanalaufbau mit Follower, Likes und Playtime, Erstellung von gutem Content, Zeitstempeln, Monetarisierung und das Partnerprogramm, Zukäufe
- Twitter, Instagram, TikTok, Xing, LinkedIn und Co.

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- eine OnPage- und eine OffPage-Optimierung für eine Webseite durchzuführen und Kunden dahingehend zu beraten.
- einen erfolgreichen YouTube Kanal aufzubauen.
- erfolgreiche Konten auf Social Media Plattformen wie Twitter, Instagram, TikTok, Xing und LinkedIn zu etablieren.

Semesterwochenstunden

Vorlesung	2 SWS
Projekt	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	45
Projektarbeit	105
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Frank Dopatka

Literatur

- Torsten Schwarz, Danylo Vakhnenko: Erfolgreiches Online-Marketing: Das Standardwerk; 5. Auflage 2021; Haufe-Verlag; https://doi.org/10.34157/9783648149706
- Dirk Lewandowski: Suchmaschinen verstehen; 3., vollständig überarbeitete und erweiterte Auflage; Springer Vieweg; https://doi.org/10.1007/978-3-662-63191-1
- Udo Raaf: Der SEO Planer - Suchmaschinenoptimierung in Unternehmen richtig organisieren und umsetzen; Springer Gabler Verlag; 2021; https://doi.org/10.1007/978-3-658-33192-4

Software-Ergonomie und Usability (SEU)

Name	Software-Ergonomie und Usability (SEU)
Kürzel	SEU
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Kirstin Kohler
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- Einführung in die Software-Ergonomie
- Unterschied Usability und User Experience
- Neue Interaktionsformen (Natürliche Interaktion, Begreifbare Interaktion)
- Psychologische Grundlagen der Mensch-Maschine-Interaktion
- Grundlagen der Kognition
- Sehen & Farbwarhnehmung
- Handlungsmodelle
- Aufmerksamkeit und Gedächtnismodelle
- Zeitwahrnehmung
- Gesetze, Verordnungen und Normen und deren Beziehung zu den Psychologischen Grundlagen
- Benutzerzentrierte Entwicklungsprozesse
- Bedarfs- und Anforderungsanalysen (Storyboarding, Taskanalyse, Hierarchische Aufgabenzerlegung)
- Spezifikation und Prototyping (High- und Low-Fidelity Prototyping, High-Fidelity Prototyping mittels eins Werkzeuges)
- Evaluation von Gestaltungsergebnissen (Experteninspektionen, Walkthrough-Verfahren, Usability-Tests, Fragebögen)
- Formen der Evaluation: summative und formative Evaluation
- Planung, Durchführung, Auswertung und Dokumentation eines Usability Tests

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- darzustellen und zu begründen, auf welche Weise benutzerbezogene Anforderungen in den Systementwicklungsprozess einfließen,
- Aufgaben- und Nutzeranalysen durchzuführen,
- eine Gestaltungslösung gemäss ihrer Gebrauchstauglichkeit zu bewerten,
- eine Produktvision in einen Prototypen für ein User Interface zu überführen und
- einfache Evaluationen u.A. im Usability Labor durchzuführen.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Prof. Kirstin Kohler
- Prof. Dr. Till Nagel

Literatur

- Shneiderman, B. & Plaisant, C.: Designing the User Interface, Pearson, 2010
- Lauesen, S.: User Interface Design - A Software Engineering Perspective, Addison Wesley, 2005
- Dahm, M.: Grundlagen der Mensch-Computer-Interaktion. München, Pearson, 2005
- Mayhew, D.: The Usability Engineering Lifecycle. A Practitioners Handbook for User Interface Design. San Francisco: Morgan Kaufmann, 1999
- Richter, M. & Flückiger, M.: Usability Engineering kompakt, Spektrum Akademischer Verlag, 2007
- DATech-Prüfhandbuch Gebrauchstauglichkeit. Leitfaden für die ergonomische Evaluierung von Software auf der Grundlage von DIN EN ISO 9241, V 3.3.; 2004, http://www.datech.de.
- S.C. Seow: Designing and Engineering Time: The Psychology of Time Perception in Software,Addison-Wesley, 2009
- J. Johnson: Designing with the Mind in Mind: Simple Guide to understanding User Interface Design Rules, Morgan Kaufmann, 2010

Sichere Internet-Dienste (SID)

Name	Sichere Internet-Dienste (SID)
Kürzel	SID
Semester	6/7
Unterrichtssprache	Deutsch und Englisch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IM, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Erfahrungen in der Software-Entwicklung, zum Beispiel aus dem Softwareprojekt/Medizinisches Softwareprojekt/Unternehmensinformatikprojekt/Cybersecurity Projekt
- Grundlagen der Internet-Sicherheit (Kryptographie, Internet-Protokolle)

Inhalte

- Sicherheitsanforderungen
- Sicherheitskonzept
- Sicherheitsarchitektur
- Security Patterns (Kryptographie, Identitätsmanagement, Authentifizierung)
- Quellen für Schwachstellen der Sicherheit
- Validieren von Schwachstellen
- Ausgewählte Internet-Dienste (Chat, VOIP, DRM, Cloud-Speicher, eGK, ...)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Die Sicherheitsanforderungen unterschiedlicher Interessensgruppen an eine Internet-basierte Lösung zu identifizieren und zu formulieren,
- Das Sicherheitskonzept bestehender Anwendungen aus öffentlich verfügbarer Information und aus der Inspektion von Code zu extrahieren und zu beschreiben,
- Schwachstellen zu identifizieren und praktisch zu valideren,
- Vergleiche von bestehenden Produkten mit vergleichbaren Anforderungen durchzuführen und zu bewerten,
- Empfehlungen zu entwickeln, wie die gewünschte Funktionalität sicher genutzt werden kann, und

- Die gewonnenen Erkenntnisse einer breiten Zielgruppe angemessen zu präsentieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	60
Selbständiges Bearbeiten der Übungen	60
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Sachar Paulus

Literatur

- bisher keines bekannt, darüber hinaus:
- Selbststudium, Untersuchung der Protokoll-Standards,...

Sichere Java Entwicklung (SJE)

Name	Sichere Java Entwicklung (SJE)
Kürzel	SJE
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	1 Semester
Studiengänge	CSB, IB, IMB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Klausur 90 Minuten (K90)

Empfohlene Vorkenntnisse

Ein Jahr Erfahrung in JAVA Programmierung

Inhalte

- XML External Entity Injection
- Unsicherer Datei-Upload
- JNDI-Injection
- Unsichere Deserialisierung
- Java Management Extensions
- Spring Boot Actuator, Jolokia, Leaks durch schlechte Fehlerbehandlung
- Injection-Angriffe
- Benutzung des Java Decompilers, Debuggers

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die wichtigsten Angriffsvektoren zu benennen,
- Bedingungen für eine erfolgreichen Angriff und Maßnahmen dagegen zu benennen,
- die Wirksamkeit von Maßnahmen selbständig zu beurteilen und angemessene Maßnahmen auszuwählen,
- Härtungsmaßnahmen anzuwenden

Semesterwochenstunden

Vorlesung	4 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	30
Prüfungsvorbereitung	30
Summe	**150**

Dozentinnen / Dozenten

- Kai Ullrich

Security Management (SMA)

Name	Security Management (SMA)
Kürzel	SMA
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- keine

Inhalte

- Security Organisation
- Security Policy
- Risikomanagement
- Sicherheitsanalysen
- Sicherheitsprozesse
- Normen und Standards für Informationssicherheit
- Return-on-Security-Investment-Berechnungen
- Krisenmanagement
- Business Continuity Management
- Zudem Ausgewählte Vertiefungsbereiche der IT- und der Unternehmenssicherheit

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Sicherheitsanalysen zu erstellen,
- Risikobewertungen durchzuführen,
- Sicherheitslagen zu analysieren, und die Sinnhaftigkeit von Gegenmaßnahmen zu bewerten,
- Zu verstehen, warum Sicherheit im Entscheidungsprozess bei Unternehmern nachgeordnet ist,
- Eine Sicherheitsorganisation für ein Unternehmen zu entwerfen und

- Sicherheitsmaßnahmen vorzuschlagen und vor einem Entscheidungsgremium erfolgreich zu platzieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	60
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	60
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Sachar Paulus

Literatur

- Security Management 2011: Handbuch für Informationsschutz, IT-Sicherheit, Standortsicherheit, Wirtschaftskriminalität und Managerhaftung von Guido Birkner, 2011.
- Handbuch Unternehmenssicherheit: Umfassendes Sicherheits-, Kontinuitäts- und Risikomanagement mit System von Klaus-Rainer Müller, 2010.
- Unternehmenssicherheit von Stephan Gundel, und Lars Mülli, 2009.

Softwareentwicklung für Medizinprodukte (SMP)

Name	Softwareentwicklung für Medizinprodukte (SMP)
Kürzel	SMP
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr.-Ing. Gerd Marmitt
Dauer	1 Semester
Studiengänge	IB, IM, IMB

Schwerpunkt

- Medical Data Science

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Programmierung 1 & 2
- Software Engineering 1 & 2

Inhalte

- Software-Lebenszyklus-Prozesse nach IEC 62304
- Technische Dokumentation für Medizingeräte-Software
- Agile Softwareentwicklung in der Medizingeräte-Software
- Medizingeräte-Software-Risikomanagement
- Qualitätsmanagementsysteme für Medizingeräte-Software-Hersteller

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- normgerechte Prozesse zur Entwicklung und Wartung von Medizingeräte-Software zu erstellen
- und Medizingeräte-Software zu entwickeln und normgerecht zu dokumentieren

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Gerd Marmitt

Literatur

- DIN EN 62304:2016-10: Medizingeräte-Software – Software-Lebenszyklus-Prozesse (IEC 62304:2006 + A1:2015); Deutsche Fassung EN 62304:2006 + Cor.:2008 + A1:2015
- DIN EN ISO 13485:2021-12: Medizinprodukte – Qualitätsmanagementsysteme – Anforderungen für regulatorische zwecke (ISO 13485:2016); Deutsche Fassung EN ISO 13485:2016 + AC:2018 + A11:2021
- ISO/TC 210 (2017): ISO 13485:2016: Medical devices – A practical guide. Advice from ISO/TC 210.
- AAMI (2012): Guidance on the use of AGILE practices in the development of medical device software, Technical Information Report (TIR), Revision 2018.
- C. Johner, M. Hölzer-Klüpfel, S. Wittorf: Basiswissen Medizinische Software: Aus- und Weiterbildung zum Certified Professional for Medical Software.

SQUAD (SQUAD)

Name	SQUAD (SQUAD)
Kürzel	SQUAD
Semester	6/7
Unterrichtssprache	Englisch
Häufigkeit	Unregelmäßig
Kreditpunkte	10 ECTS
Modulverantwortlich	Prof. Thomas Smits
Dauer	2 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Auswahl durch Interviews
- Gute Englischkenntnisse

Inhalte

- Grundlagen des Design Thinking
- Entwickeln einer Produktidee mit den Methoden des Design Thinking, ausgehend von der Challenge eines Unternehmens
- Selbständige Projektarbeit im (nationalen Informatik-)Teilteam und im internationalen, interdisziplinären Team
- Das Modul erstreckt sich über zwei Semester (von Oktober bis Mai)

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- die Methoden des Design Thinking unter Anleitung anzuwenden
- Nutzer-zentriert und Prototyp-basiert zu arbeiten
- als Informatiker in einem internationalen, interdisziplnären Team innovative Produktideen zu entwickeln

Semesterwochenstunden

Projekt	6 SWS
Workshop	2SWS
Summe	**8 SWS**

Arbeitsaufwand (work load)

Projektarbeit	240
Workshop	60
Summe	**300**

Dozentinnen / Dozenten

- Prof. Thomas Smits
- Prof. Dr. Oliver Hummel

Literatur

- Michael Lewrich, Patrick Link, Larry Leifer: Das Design Thinking Playbook, 2. Auflage, Vahlen, 2018

Sichere Softwarentwicklung (SSE)

Name	Sichere Softwarentwicklung (SSE)
Kürzel	SSE
Semester	6/7
Unterrichtssprache	Deutsch und Englisch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Sachar Paulus
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

- Erfahrungen in der Software-Entwicklung, zum Beispiel aus dem Softwareprojekt/Medizinisches Softwareprojekt/Unternehmensinformatikprojekt
- Erste Erfahrungen im Programmieren von Web-Anwendungen für das Beispiel-Szenario
- Ansonsten: Selbststudium, z. B. mit PHP 5.3: Dynamische Websites professionell programmieren von Christian Wenz und Tobias Hauser (Dezember 2009)

Inhalte

- Grundsätze der sicheren Software-Entwicklung:
- Sicherheitsanforderungen
- Sicheres Design und Bedrohungsmodellierung
- Architekturanalysen
- Sicheres Kodieren
- Sicherheitstests
- Sichere Einrichtung
- Security Response
- Schutz der eigenen Software for Manipulation und Know-How-Diebstahl

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Best Practices für sichere Software während der Entwicklung von IT-basierten Systemen anzuwenden,
- Akzeptanzkriterien für nicht-funktionale Sicherheitsanforderungen zu entwickeln,

- Bedrohungsmodellierungen durchzuführen,
- Schwachstellen während der Software-Entwicklung zu vermeiden,
- Bestehende Software auf Sicherheitsschwachstellen hin zu analysieren und
- die Ergebnisse eines Penetrationstests management-tauglich zu präsentieren.

Semesterwochenstunden

Vorlesung	2 SWS
Übung	1 SWS
Projekt	1 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	30
Selbständiges Bearbeiten der Übungen	30
Projektarbeit	60
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Sachar Paulus

Literatur

- Basiswissen sichere Software von Sachar Paulus, dpunkt 2011.
- Software-Qualität, Testen, Analysieren und Verifizieren von Software von Peter Liggesmeyer, Spektrum Akademischer Verlag, 2002.
- Writing Secure Code von Michael Howard & David LeBlanc, 2003
- www.owasp.org

Studienarbeit (STA)

Name	Studienarbeit (STA)
Kürzel	STA
Semester	6/7
Unterrichtssprache	Deutsch und Englisch
Häufigkeit	Jedes Semester
Kreditpunkte	10 ECTS
Modulverantwortlich	Prof. Dr. Peter Knauber
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Hausarbeit (HA)

Empfohlene Vorkenntnisse

- Module des ersten bis fünften Semesters bestanden

Inhalte

- Modulübergreifendes praktisches Projekt, Mitarbeit an Fakultäts-internen Forschungsprojekten, Weiterentwicklung innovativer Lehr- und Lernkonzepte
- Recherche der fachlichen und methodischen Grundlagen
- Realisierung einer Lösung unter Anwendung wissenschaftlicher Methoden der Informatik
- Validierung der Lösung
- Angemessene Dokumentation der Lösung und Präsentation der Ergebnisse

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- eine kleine wissenschaftliche Forschungs- oder Entwicklungsaufgabe aus dem Bereich der Informatik selbständig zu lösen
- wissenschaftliche Methoden der Informatik zielgerichtet einzusetzen
- Ergebnisse in angemessenem Format (z. B. Teil eines umfassenden Projektberichts, Anleitung zur Durchführung eines Experiments in einer Vorlesung) zu dokumentieren
- Ergebnisse für nicht-Fachleute des jeweiligen Spezialgebiets zu präsentieren

Arbeitsaufwand (work load)

Projektarbeit	300
Summe	**300**

Dozentinnen / Dozenten

- Alle Dozenten der Fakultät

Literatur

- Laplante, Phillip A (2011). Technical writing: a practical guide for engineers and scientists. CRC Press.
- Kirkman, John und Christopher Turk (2002). Effective writing: improving scientific, technical and business communication. Taylor & Francis.
- Alred, Gerald J, Charles T Brusaw und Walter E Oliu (2015). Handbook of technical writing. Bedford/St. Martin's.
- Könneker, Carsten (2012). Wissenschaft kommunizieren: ein Handbuch mit vielen praktischen Beispielen. John Wiley & Sons.

Urban Hacking (UHA)

Name	Urban Hacking (UHA)
Kürzel	UHA
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Till Nagel
Dauer	1 Semester
Studiengänge	IB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- Physical Visualization / Physicalization
- Schnelle prototypische Umsetzungstechniken
- Datenerhebung, -aufbereitung, und -nutzung aus nicht-traditionellen Quellen
- Urbane Interventionen und kritische Nutzung des öffentlichen Raums

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- datengestützte Prototypen zu entwerfen.
- interdisziplinär mit Studierenden anderer Fakultäten Probleme zu bewältigen.
- Ideen und Konzepte auszuarbeiten und gegenüber anderen zu begründen und zu verteidigen.
- soziale und technologische Aspekte gesellschaftlich relevanter Themen einzuschätzen.

Semesterwochenstunden

Vorlesung	1 SWS
Projekt	3 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	30
Projektarbeit	90
Summe	**150**

Dozentinnen / Dozenten

- Prof. Dr. Till Nagel

Literatur

- Townsend, A. (2014) Smart Cities: Big Data, Civic Hackers, and the Quest for a New Utopia. New York: W.W. Norton & Company
- Jansen et al. (2015) Opportunities and Challenges for Data Physicalization, CHI 2015
- Huron et al. (2017) Let's Get Physical: Promoting Data Physicalization in Workshop Formats, DIS 2017

UX Research und Design (UXD)

Name	UX Research und Design (UXD)
Kürzel	UXD
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Unregelmäßig
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Kirstin Kohler
Dauer	1 Semester
Studiengänge	CSB, IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Continuous Assessment (CA)

Empfohlene Vorkenntnisse

Inhalte

- Wer sind User?
- User Research Methoden
- Interview Methoden
- User-centered Design
- Intuitive Design
- Design Thinking
- Prototyping (high-fidelity, low-fidelity)
- Feedback-Zyklen
- Double Diamond-Process
- Iconography
- Color Schemes
- Design Systeme
- Accessibility

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Probleme strukturiert zu analysieren und die bestmögliche Lösung dafür herauszufinden,
- sich in den User und die Usability von Software besser hineinzuversetzen,
- moderne Werkzeuge des Software-Designs einzusetzen und den Nutzer beim Design neuer Software regelmäßig miteinzubeziehen,

- kritisch bisherige Design-Lösungen auf ihre Usability zu hinterfragen und neu zu designen.

Semesterwochenstunden

Vorlesung	2 SWS
Projekt	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzstudium	30
Eigenstudium zur Vor- und Nachbereitung	90
Selbständiges Bearbeiten der Übungen	30
Summe	**150**

Dozentinnen / Dozenten

- Jacqueline Franßen

Literatur

- Laws of UX - Using Psychology to Design Better Products & Services (Jon Yablonski), O'Reilly Media, 2020, ISBN: 9781492055310
- The Design of Everyday Things (Don Norman), Basic Books, 2013, ISBN-10: 9780465050659
- Emotional Design (Don Norman), Basic Books, 2005, ISBN-13: 978-0465051366

Webarchitekturen und -frameworks (WAF)

Name	Webarchitekturen und -frameworks (WAF)
Kürzel	WAF
Semester	6/7
Unterrichtssprache	Deutsch
Häufigkeit	Wintersemester
Kreditpunkte	5 ECTS
Modulverantwortlich	Prof. Dr. Thomas Specht
Dauer	1 Semester
Studiengänge	IB, IMB, UIB

Studien-/Prüfungsleistung

Studienleistung	Keine
Prüfungsleistung	Klausur 60 Minuten (K60)

Empfohlene Vorkenntnisse

Für dieses Wahlpflichtmodul werden die Grundlagen webbasierter und verteilter Systeme sowie Datenbanken vorausgesetzt

- Webbasierte Systeme (WEB)
- Datenmanagement (DM)

VS , kann parallel gehört werden

- HTTP-Protokoll, HTML, JavaScript, JSON
- RESTful Web Services

Inhalte

- Fortgeschrittene serverseitige Web-Architekturen und -Frameworks
- npm, Node.js
- Express-Framework
- Nest-Framework
- MongoDB, Mongoose
- RESTful Web Services mit Node, Express/Nest und MongoDB
- Absicherung von RESTful Web Services
- RESTful Web Service Client mit Angular

Lernziele/Kompetenzen

Die Studierenden sind in der Lage,

- Anforderungen an eine komplexe Webanwendung analysieren und eine geeignete Architektur entwerfen

- Eigenschaften und Konzepte serverseitiger Web-Programmiersprachen beschreiben und bewerten
- Eigenschaften und Konzepte serverseitiger Webframeworks beschreiben und bewerten
- Webanwendungen mit serverseitigen Webframeworks entwickeln und absichern

Semesterwochenstunden

Vorlesung	2 SWS
Übung	2 SWS
Summe	**4 SWS**

Arbeitsaufwand (work load)

Präsenzübungen als Online Live-Stream mit Bearbeitung am eigenen Rechner

Dozentinnen / Dozenten

- Prof. Dr. Thomas Specht

Literatur

- npm Docs: https://docs.npmjs.com/
- Introduction to Node.js: https://nodejs.dev/learn
- Node.js API Documentation: https://nodejs.org/api/
- Express Guide: https://expressjs.com/
- Expresss API Documentation: https://expressjs.com/de/4x/api.html
- Nest Documentation: https://docs.nestjs.com/
- MongoDB Manual: https://docs.mongodb.com/manual/tutorial/getting-started/
- Mongoose Guide: https://mongoosejs.com/docs/guide.html