Angewandte Informatik

Pflichtfächer

1. Semester
   - Mathematische Grundlagen
   - Grundlagen der Informatik
   - Programmiermethodik I
   - Programmiertechnik
   - Betriebswirtschaft I
2. Semester
   - Logik und Berechenbarkeit
   - Automaten und Formale Sprachen
   - Datenbanken
   - Programmiermethodik II
   - Rechnerstrukturen und maschinennahe Programierung
3. Semester
   - Graphentheoretische Konzepte und Algorithmen
   - Algorithmen und Datenstrukturen
   - Software Engineerinig I
   - Betriebssysteme
   - Betriebswirtschaft II
4. Semester
   - Intelligente Systeme
   - Software Engineering II
   - Rechnernetze
5. Semester
   - Architektur von Informationssystemenme
   - Verteilte Systeme
6. Semester
   - IT-Sicherheit

Modulbezeichnung	**Mathematische Grundlagen**	Kürzel	MG /MGÜ
Lehrveranstaltung(en)	Vorlesung: Mathematische Grundlagen Übung: Mathematische Grundlagen	Semester	1
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Christoph Klauck	SWS	3+1
Dozenten	Prof. Dr. Julia Padberg	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können wichtige mathematische Strukturen sicher verwenden • beherrschen die logischen und algebraischen Grundlagen der theoretischen Informatik • können eine präzise und abstrakte Denkweise sowie die formale Denk- und Argumentationsweise in praxisorientierten Problemen anwenden • können Definitionsprinzipien und Beweistechniken in unterschiedlichen Bereichen und an typischen Beispielen anwenden		
Inhalte	• Mathematische Grundlagen: Mengen, Relationen, Abbildungen, Funktionen und deren Operatoren, Boolsche Algebra, Aussagenlogik • Mathematische Techniken: Grundlegende Beweisstrategien, Vollständige Induktion • Mathematische Strukturen: Lösung von linearen Gleichungssystemen, Vektoren, Matrizen, Determinanten • Vertiefung in folgende Richtungen: - Modulare Arithmetik (Zahlentheorie), - Kombinatorik, - Diskrete Stochastik		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Übung: Selbständiges bearbeiten der Aufgaben, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführte Übung Übung: erfolgreiche Bearbeitung spezieller Übungsaufgaben (PVL)		
Literatur	• M. Aigner, G.M. Ziegler: Das Buch der Beweise • P. Bundschuh: Einführung in die Zahlentheorie • N. Dean: Diskrete Mathematik • P. Hartmann: Mathematik für Informatiker • C. Meinel, M. Mundhenk: Mathematische Grundlagen der Informatik • N. Nerode, R.A. Shore: Logic for applications • R. Staszewski, K. Strambach, H. Völklein: Lineare Algebra • C. Stein, R.L. Drysdale, K. Bogart: Discrete Mathematics for Computer Scientists		

Modulbezeichnung	**Grundlagen der Informatik**	Kürzel	GI /GIÜ
Lehrveranstaltung(en)	Vorlesung: Grundlagen der Informatik Übung: Grundlagen der Informatik	Semester	1
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Böhm	SWS	3+1
Dozenten	Prof. Dr. Michael Böhm, Prof. Dr. Kai von Luck, Prof. Dr. Birgit Wendholt	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen die Grundlagen, Prinzipien und Grenzen der Informatik		
Inhalte	Propädeutikum der Informatik • Information und Informatik • Entwicklungslinien der Informatik • Überblick über Kerngebiete und Anwendungsbereiche der Informatik • Grundlagender informatikspezifischen Herangehensweise an Probleme (Formalisierung, Modellbildung) • zukünftige Trends der Informatik • Informatik als Wissenschaftsdisziplin		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafel, Präsentation, Beispielaufgaben, Demos, freiwillige Übungsaufgaben, evtl. Tutorium Übung: Selbständiges bearbeiten der Aufgaben, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführte Übung Übung: erfolgreiche Bearbeitung der Übungsaufgaben (PVL)		
Literatur	• Harel, D.: Algorithmics. The Spirit of Computing , Addison-Wesley • Herold, Lurz, Wohlrab: Grundlagen der Informatik, Pearson		

Modulbezeichnung	**Programmiermethodik I**	Kürzel	PM1
Lehrveranstaltung(en)	Vorlesung: Programmiermethodik I	Semester	1
Arbeitsaufwand	48 Std. Vorlesung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Böhm	SWS	4+0
Dozenten	Prof. Dr. Michael Böhm, Prof. Dr. Birgit Wendholt, N.N.	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können die Basiskonzepte einer modernen Programmiersprache benennen und anwenden • können eine abstrakte Problembeschreibung in einen programmierbaren Algorithmus übertragen • können objektorientierte Problemlösungen modellieren		
Inhalte	• Vom Problem zum Programm: Strukturiertes Vorgehen beim Programmieren • Typisierungskonzepte, Werte- und Referenztypen • dynamischer Umgang mit Typen • Darstellung und Analyse von Kontrollflüssen • funktionale Abstraktion, Datenabstraktion (ADT), Kontrollabstraktion (z.B. Iteratoren, Streams) • Polymorphie (überladen von Methoden)		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat		
Literatur	• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben		

Modulbezeichnung	**Programmiertechnik**	Kürzel	PT/PTP
Lehrveranstaltung(en)	Vorlesung: Programmiertechnik Praktikum: Programmiertechnik	Semester	1
Arbeitsaufwand	24 Std. Vorlesung, 24 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Böhm	SWS	2+2
Dozenten	Prof. Dr. Michael Böhm, Prof. Dr. Birgit Wendholt, N.N.	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können eine moderne Entwicklungsumgebung bedienen (Editor, Debugger) • können einfache Programme in einer modernen Programmiersprache entwickeln • können fremden Quellcode analysieren und in eigene Programme integrieren • können Qualitätskriterien für lesbaren, wartbaren, wiederverwendbaren Quellcode nennen und diese beim Erstellen eigener Programme umsetzen • können Programme automatisiert testen		
Inhalte	• Syntax einer modernen Programmiersprache: primitive Datentypen, Unicode, Arrays, Referenztypen, Sequenz, Selektion, Iteration, Klassen, Objekte • Testabdeckung und Frameworks • überführen eines Entwurfs in ein lauffähiges Programm • Umgang mit Daten		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in 2-er Gruppen, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben		

Modulbezeichnung	**Betriebswirtschaftslehre I**	Kürzel	BW1/BWÜ1
Lehrveranstaltung(en)	Vorlesung: Betriebswirtschaftslehre I Übung: Betriebswirtschaftslehre I	Semester	1
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Martin Hübner	SWS	3+1
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Gerken	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen rechtliche, finanzielle und organisatorische Strukturen von Unternehmen, • verstehen die Bedeutung von wirtschaftlichen Vorgehensweisen und können entsprechende Controlling-Instrumente anwenden • können Kostenberechnungen selbstständig durchführen, • können Investitionsentscheidungen nach betriebswirtschaftlichen Kriterien treffen.		
Inhalte	• Das Unternehmen als System • Rechtsformen und Aufbauorganisation • Ablauforganisation und Methoden zu ihrer Beschreibung • Grundlagen der Finanzbuchhaltung (Buchführung und Jahresabschluss) • Kosten- und Leistungsrechnung • Finanzierung und Investitionsrechnung		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben Übung: selbstständiges Lösung von Übungsaufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich absolvierte Übung Übung: erfolgreiche Bearbeitung aller Aufgaben (PVL)		
Literatur	• G. Wöhe: Einführung in die allgemeine BWL, Verlag Franz Vahlen • A. J. Schwab: Managementwissen für Ingenieure, Springer-Verlag • Dietmar Vahs, Jan Schäfer-Kunz: Einführung in die BWL, Schäffer-Poeschel Verlag • Siegfried Schmolke, Manfred Deitermann: Industrielles Rechnungswesen IKR, Winklers Verlag • Eigene Skripte der Dozenten		

Modulbezeichnung	**Logik und Berechenbarkeit**	Kürzel	LB /LBP
Lehrveranstaltung(en)	Vorlesung: Logik und Berechenbarkeit Praktikum: Logik und Berechenbarkeit	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Christoph Klauck	SWS	3+1
Dozenten	Prof. Dr. Thomas Thiel-Clemen, Prof. Dr. Christoph Klauck	Sprache	deutsch
Voraussetzungen	Grundlagen der Informatik, Mathematische Grundlagen	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können formale Modellierungskonzepte auf Basis von Kalkülen und Grammatiken anwenden • können eigenständig präzise Beschreibungen in praxisorientierten Probleme erstellen • verstehen die Methoden der Korrektheitsbeweise für Modelle und Algorithmen • besitzen die Fähigkeit, Berechenbarkeitgrenzen zu erkennen und praktische Probleme zu klassifizieren		
Inhalte	• Einführung in die Aussagenlogik: logische Systeme, wohlgeformte Ausdrücke, Wahrheitswerte/-tafeln, Belegungsfunktion, Normalformen, Literale und Klauseln • Grundkonzepte der Semantik: Allgemeingültigkeit, Unerfüllbarkeit, Erfüllbarkeit und Folgerung • Beweistheorie und Ableitungssysteme für die Aussagenlogik: Schlussregeln, modus ponens, Resolutionskalkül und Tableaukalkül in der Aussagenlogik • Einführung in die Prädikatenlogik erster Stufe: Charakterisierung wohlgeformter Ausdrücke, Quantoren und Quantorenskopus • Entscheidbarkeit und Berechenbarkeit: primitiv rekursive und allgemein-rekursive Funktionen, Register- und Turing-Maschine als formale Algorithmusdefinition, Halteproblem und andere unentscheidbare Fragen • Konkrete Komplexitätstheorie: Landausche Notation, einfache Algorithmen, auch für einfache randomisierte Algorithmen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen, Selbständiges bearbeiten der Aufgaben, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• C. Baier, A. Asteroth: Theoretische Informatik; Einführung in Berechenbarkeit, Komplexität und formale Sprachen • J.E. Hopcroft, R. Motwani, J.D. Ullman: Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie • M. Kreuzer, S. Kühling: Logik für Informatiker • E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kann, D.B. Shmoys: The Traveling Salesman Problem • R. Socher: Theoretische Grundlagen der Informatik • G. Vossen, K.-U. Witt: Grundkurs Theoretische Informatik • Eigene Skripte der Dozenten		

Modulbezeichnung	**Automaten und formale Sprachen**	Kürzel	AF/AFÜ
Lehrveranstaltung(en)	Vorlesung: Automaten und formale Sprachen Übung: Automaten und formale Sprachen	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Julia Padberg	SWS	3+1
Dozenten	Prof. Dr. Padberg, Prof. Dr. Neitzke, Prof. Dr. Buth, Prof. Dr. Korf, Prof.Dr. Köhler-Bußmeier	Sprache	deutsch
Voraussetzungen	Mathematische Grundlagen, Kenntnisse von formalen Beweisen, Prädikatenlogik, speziell Induktion	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können formale Beweise erläutern und selber durchführen • können formale Modelle in Form von Automaten, Regulären Ausdrücken und Grammtiken verstehen und erstellen • können Zusammenhänge zwischen Automatenmodellen, regulären Sprachen und Grammatiken herstellen und Modelle ineinander überführen • können formale Spezifikationen auf Problemstellungen der realen Welt anwenden		
Inhalte	• Automaten: Die Grundlagen und Methoden • Endliche Automaten • Reguläre Ausdrücke und Sprachen • Eigenschaften regulärer Sprachen • Kontextfreie Grammatiken und Sprachen • Eigenschaften kontextfreier Sprachen • Vertiefung in eine der folgenden Richtungen: Kellerautomaten, zeitbehaftete Automaten, Modellierung mit formalen Methoden		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Präsentationen, Übungsaufgaben, studentische Referate, Gruppenarbeit Übung: selbstständiges Lösen der Übungsaufgaben in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführte Übung Übung: erfolgreiche Bearbeitung der Übungsaufgaben (PVL)		
Literatur	• J.E. Hopcroft, R. Motwani; J.D. Ullman: Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie, Addison-Wesley, 2011 • C. Baier, A. Asteroth: Theoretische Informatik, 2002 • R. Socher: Theoretische Grundlagen der Informatik, 2005 • The Theory of Timed I/O Automata, Second Edition (Synthesis Lectures on Distributed Computing Theory) von Dilsun Kaynar, Nancy Lynch, Roberto Segala und Frits Vaandrager von Morgan & Claypool Publishers (1. Dezember 2010)		

Modulbezeichnung	**Datenbanken**	Kürzel	DB/DBP
Lehrveranstaltung(en)	Vorlesung: Datenbanken Praktikum: Datenbanken	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Thomas Thiel-Clemen	SWS	3+1
Dozenten	Prof. Dr. Wolfgang Gerken, Prof. Dr. Stefan Sarstedt , Prof. Dr. Thomas Thiel-Clemen, Prof. Dr. Olaf Zukunft	Sprache	deutsch
Voraussetzungen	Programmieren I (PR1, PRP1), Mathematische Grundlagen (MG)	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden: • kennen die Einsatzgebiete und Grenzen von Datenbanken • beherrschen des Prozess des Datenbankentwurfs • kennen die theoretischen Grundlagen von Datenbanksystemen • können einfache Datenbankanwendungen entwickeln • beherrschen die relationale Anfragesprache SQL im Rahmen des Standards • verstehen Datenschutzmechanismen und gesellschaftliche Auswirkungen großer Datensammlungen		
Inhalte	• Grundkonzepte relationaler Datenbanksysteme • der logische Entwurf und die Überführung in das technische Design • Implementierung und Befüllung von Datenbanksystemen • Anfragen und Transaktionen • programmiersprachliche Schnittstellen • Alternativen zum relationalen Modell		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in Kleingruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• A. Kemper / A. Eickler, Datenbanksysteme – Eine Einführung, Oldenbourg 2011. • R.A. Elmasri / S.B. Navathe, Grundlagen von Datenbanksystemen, Pearson 2009 • Eigene Skripte der Dozenten		

Modulbezeichnung	**Programmiermethodik II**	Kürzel	PM2/PMP2
Lehrveranstaltung(en)	Vorlesung: Programmiermethodik II Praktikum: Programmiermethodik II	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Böhm	SWS	3+1
Dozenten	Prof. Dr. Michael Böhm, Prof. Dr. Birgit Wendholt, N.N.	Sprache	deutsch
Voraussetzungen	PM1, PT1, GWI	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • beherrschen fortgeschrittene Programmiertechniken und können diese in kleinen Beispielprogrammen in einer aktuellen Programmiersprache einsetzen • können technische Basis-Hilfsmittel für die teamorientierte SW-Entwicklung anwenden (z.B. Versionsverwaltung)		
Inhalte	• Generische Template-Klassen (Generics, Reflexion) • Nebenläufige bzw. asynchrone Programmierung (Threads) • Vertiefungen: Typ- vs. Implementierungshierarchie, elementare Entwurfsmuster, UML, XML • Entwurf gemäß Vertrag (Bedingungen, Invarianten) • vertiefte Teststrategien • GUI-Programmierung • Datenbankanbindung		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in 2-er Gruppen, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben		

Modulbezeichnung	**Rechnerstrukturen und Maschinennahe Programmierung**	Kürzel	RMP/RMPP
Lehrveranstaltung(en)	Vorlesung: Rechnerstrukturen und Maschinennahe Programmierung Praktikum: Rechnerstrukturen und Maschinennahe Programmierung	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Andreas Meisel	SWS	3+1
Dozenten	Prof. Dr. Andreas Meisel, Prof. Dr. Heiner Heitmann , Prof. Dr. Reinhardt Baran	Sprache	deutsch
Voraussetzungen	Grundlagen der Informatik	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden: • können maschinennahe Datentypen und Programmierparadigmen verstehen und anwenden • verstehen grundlegende Rechnerarchitekturkonzepte sowie die Instruction-set-Architektur eines aktuellen Prozessors • verstehen den Zusammenhang zwischen einer maschinennahen Hochsprache (z.B. ANSI-C) und der unterlagerten Maschinensprache • können Programme auf niedrigem Abstraktionslevel strukturieren, modularisieren und entwickeln • können einfache Assembler-Programme für einen ausgewählten Prozessors erstellen • können in einer maschinennahen Hochsprache (ANSI-C) einfache Anwendungen entwickeln • können mit Entwicklungswerkzeugen für die maschinennahe Programmierung umgehen		
Inhalte	• Darstellung von Daten im Computer • Rechnerarchitekturgrundlagen auf Ebene der Instruction-set-Architektur: Rechner-Grundkomponenten (Speicher, ALU usw.), von Neumann- und Harvardarchitektur • maschinennahe Programmierung auf Basis des Maschinenbefehlssatzes eines aktuellen Prozessors • Konzepte einer hardwarenahen höheren Programmiersprache, zum Beispiel ANSI-C • Projekteverwaltung, Modultechnik, Bibliotheken • Interfaces zur Verzahnung von Hochsprachen und Assembler • Abbildung von Daten und Kontrollstrukturen prozeduraler Hochsprachen in maschinennahe Implementierungen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in Kleingruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Grundlagenbücher: s. Bibliothek, Signatur Dat 001 • Joachim Groll, Ulrich Bröckl, Manfred Dausmann: C als erste Programmiersprache, Teubner Verlag. • B.W. Kernighan, D.M. Ritchie: Programmieren in C, Hanser Verlag. • Andrew S. Tanenbaum, James Goodman: Computerarchitektur, Pearson Studium • W. Hohl: ARM Assembly Language: Fundamentals and Techniques • eigene Skripte der Dozenten + bereitgestellte Zusatzmaterialien (Befehlsreferenz)		

Modulbezeichnung	**Graphentheoretische Konzepte und Algorithmen**	Kürzel	GKA /GKAP
Lehrveranstaltung(en)	Vorlesung: Graphentheoretische Konzepte und Algorithmen Praktikum: Graphentheoretische Konzepte und Algorithmen	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Christoph Klauck	SWS	3+1
Dozenten	Prof. Dr. Julia Padberg, Prof. Dr. Christoph Klauck	Sprache	deutsch
Voraussetzungen	Grundlagen der Informatik, Mathematische Grundlagen, Logik und Berechenbarkeit, Automaten und formale Sprachen	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können in praxisorientierten Problemen erfolgreiche graphentheoretischen Modellierungsparadigmen und Formalismen anwenden • beherrschen die grundlegenden Konzepte, Formalismen und Notationen sowie die wichtigsten Algorithmen • können eigenständig praxisorientierten Probleme mit graphentheoretischen Methoden Modellieren und Lösen • besitzen die Fähigkeit zum eigenständigen Modellieren, einfachen Analyse und einem Redesign von nebenläufigen Prozessen mittels Petri-Netzen		
Inhalte	• Graphentheoretische Grundbegriffe, Wege, Kreise, Zusammenhang • Färbungen und Überdeckungen • Bäume, Wälder, Matroide • Suchstrategien, Kürzeste Wege, Flüsse und Strömungen • Matchings, Routing, Planare Graphen • Graphtransformationen • Grundlegende Eigenschaften von Petri-Netzen • Berechenbarkeit, Erreichbarkeit und Erzeugbarkeit von Petri-Netzen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen, Selbständiges bearbeiten der Aufgaben, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• R. Diestel: Graph Theory. • H. Ehrig, K. Ehrig, U. Prange, G. Taentzer: Fundamentals of Algebraic Graph Transformation • H. Ehrig, W. Reisig, G. Rozenberg: Petri Net Technology for Communication-Based Systems • J. Hromkovic, M. Nagl, B. Westfechtel: Graph-Theoretic Concepts in Computer Science • S.O. Krumke, H. Noltemeier: Graphentheoretische Konzepte und Algorithmen • L. Priese, H. Wimmel: Theoretische Informatik; Petri-Netze • Eigene Skripte der Dozenten		

Modulbezeichnung	**Algorithmen und Datenstrukturen**	Kürzel	AD/ADP
Lehrveranstaltung(en)	Vorlesung: Algorithmen und Datenstrukturen	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Böhm	SWS	3+1
Dozenten	Prof. Dr. Michael Böhm, Prof. Dr. Christoph Klauck, Prof. Dr. Birgit Wendholt	Sprache	deutsch
Voraussetzungen	PR 1+2, GI	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können Algorithmen und Datenstrukturen entwerfen, • können Algorithmen und Datenstrukturen analysieren • können Algorithmen und Datenstrukturen anwenden		
Inhalte	• Prinzipien der Algorithmenanalyse • Design von Algorithmen (z.B. Divide and Conquer, Randomisierung) • Komplexität (Komplexitätsklassen, NP-Vollständigkeit) • Datenstrukturen (z.B. Liste, Stack, Queue) • Suchen (z.B. Suchbäume, Hashing) • Sortieren • Bäume und Graphen • Ausgewählte Anwendungsbeispiele (z.B. Datenkompression, Simulation)		
Lehr- und Lernformen	Vorlesung: Tafel, Präsentation, Beispielaufgaben, Demos, freiwillige Übungsaufgaben, evtl. Tutorium Praktikum: Selbständige Bearbeitung der Aufgaben in 2-er Gruppen, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Cormen,T.H., Leiserson,C.E., Rivest,R.L.: Introduction to Algorithms, McGraw-Hill • Sedgewick, R.: Algorithms in Java – 3rd ed., Addison-Wesley		

Modulbezeichnung	**Software Engineering I**	Kürzel	SE1/SEP1
Lehrveranstaltung(en)	Vorlesung: Software-Engineering I Praktikum: Software-Engineering I	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Bernd Kahlbrandt	SWS	3+1
Dozenten	Prof. Dr. Bernd Kahlbrandt, Prof. Dr. Thomas Lehmann, Prof. Dr. Stefan Sarstedt, Prof. Dr. Thiel-Clemen, Prof. Dr. Olaf Zukunft	Sprache	deutsch
Voraussetzungen	Datenbanken (DB, DBP), Programmieren (PM1, PT 1, PM2)	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden Technisch, operational: • können die Begriffswelt des Anwenders durch geeignete Vorgehensweisen erfassen und zu einer fachlichen Terminologie im Projekt verdichten • können Ausschnitte aus der Realwelt mit Hilfe geeigneter, aktueller Methoden modellieren • beherrschen Standardsituationen im Bereich der Modellierung (Architekturen, Entwurfsmuster) • können Benutzungsschnittstellen als Teil der Anwendung konzipieren (einschließlich Benutzungs-schnittstellenentwurf) • können einen Systementwurf in eine produktiv einsetzbare Systemimplementierung überführen Methodisch, konzeptionell: • können sich in einen Problembereich einarbeiten • können ein zutreffendes Modell der Realwelt in eine qualitativ hochwertige Anwendung überführen • können Prototyping als Hilfsmittel des Erkenntnisprozesses nutzen • können neben den erlernten Methoden auch andere Methoden einordnen • können sich in andere Methoden einarbeiten • können Qualitätswesen als Bestandteil des Entwicklungsprojektes einsetzen Übergreifend entwickeln die Studierenden: • technische Kompetenz (Programmiersprache, Entwicklungsumgebung, Tools, Notation, etc.) • allgemeine Methodenkompetenz • Transferkompetenz (Theorie in Praxis, Anwendung von Bekanntem auf neue Situationen etc.) • Soziale Kompetenz und • können die ökonomischen Randbedingungen und deren Auswirkungen und sicher beurteilen und dementsprechend die Projekte planen		
Inhalte	• Grundlagen: Qualitätsbegriff, Problemlösungsstrategien, systematisches, wissenschaftliches Arbeiten, Architektur und Komponenten, Benutzer-Entwickler-Kommunikation • Systementwicklung: Software-Entwicklungsprozess, Entwicklungsergebnisse, Entwurfsprachen (UML), Etablierung von Projekten und Softwareentwurf, Erheben und Formulieren von Anforderungen, Analyse und Design, Dokumentation (Benutzer-, Entwickler-, Projekt-), Planen und Steuern • Im Praktikum: Etablierung eines Projekts, Entwurf und Architekturgestaltung, Modellierung im Projektkontext.		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in Gruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• - Ian Sommerville. Software Engineering. Pearson Education, München, jeweils aktuelle Auflage • - Aktuelle UML Literatur • - Eigene Skripte der Dozenten		

Modulbezeichnung	**Betriebssysteme**	Kürzel	BS/BSP
Lehrveranstaltung(en)	Vorlesung: Betriebssysteme Praktikum: Betriebssysteme	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Martin Hübner	SWS	3+1
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Bettina Buth, Prof. Dr. Wolfgang Fohl, Prof. Dr. Franz Korf	Sprache	deutsch
Voraussetzungen	Programmieren, Rechnerstrukturen und Maschinennahe Programmierung	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen die Architektur, die Konzepte und die Funktionsweise moderner Betriebssysteme sowie des Zusammenspiels von Hard- und Software, • verstehen die Konzepte zur Implementierung systemnaher Software • können das Verhalten von Computersystemen analysieren und beschreiben • können Grundkonzepte der nebenläufigen Programmierung anwenden		
Inhalte	• Architekturen und Betriebsarten • Prozess- und Thread-Konzept, Scheduling • Synchronisation, Interprozesskommunikation, Deadlocks • Hauptspeicherverwaltung, Virtueller Speicher • Verwaltung externer Geräte • Dateisysteme • Schutzmechanismen, Sicherheitsaspekte • Exemplarische Betrachtung aktueller Betriebssysteme		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Andrew S. Tanenbaum, Modern Operating Systems, Pearson Studium Verlag • Abraham Silberschatz, Peter Galvin, Greg Gagne: Operating System Concepts with Java, John Wiley & Sons • Eduard Glatz: Betriebssysteme: Grundlagen, Konzepte, Systemprogrammierung, dpunkt Verlag • Eigene Skripte der Dozenten		

Modulbezeichnung	**Betriebswirtschaftslehre II**	Kürzel	BWL2/BWLP2
Lehrveranstaltung(en)	Vorlesung: Betriebswirtschaftslehre II Praktikum: Betriebswirtschaftslehre II	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Wolfgang Gerken	SWS	3+1
Dozenten	Prof. Dr. Wolfgang Gerken, Prof. Dr. Ulrike Steffens, Prof. Dr. Thiel-Clemen	Sprache	deutsch
Voraussetzungen	BWL1, Datenbanken	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden sollen die Rolle und Bedeutung der Informatik mit ihren Organisationsformen, Methoden und Lösungen in Unternehmen verstehen und in der Lage sein, betriebswirtschaftliche Anwendungsaspekte in Informatik-Projekten umzusetzen.		
Inhalte	• Informationsmanagement in Unternehmen - BWL und Informationstechnologie - Aufbauorganisation IT - IT-Governance - IT-Servicemanagement • Materialwirtschaft und Produktion - Betriebswirtschaftliche Grundlagen - PPS-Systeme • Marketing und Vertrieb - Betriebswirtschaftliche Grundlagen - Electronic Commerce • Unternehmensführung und Controlling - Betriebswirtschaftliche Grundlagen - Management Informationssysteme • Integrierte Informationssysteme - ERP-Systeme - Services und Prozesse als Integrationsmittel		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Andreas Gadatsch: Grundkurs Geschäftsprozessmanagement, Vieweg 2012 • Kenneth und Jane Laudon, Wirtschaftsinformatik, Addison-Wesley 2009 • Eigene Skripte der Dozenten		

Modulbezeichnung	**Intelligente Systeme**	Kürzel	IS / ISP
Lehrveranstaltung(en)	Vorlesung: Intelligente Systeme Praktikum: Intelligente Systeme	Semester	4
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Kai von Luck	SWS	3+1
Dozenten	Prof. Dr. Kai von Luck, Prof. Dr. Michael Neitzke, Dr. Sabine Schumann, Prof. Dr. Christoph Klauck	Sprache	deutsch
Voraussetzungen	Logik und Berechenbarkeit, Programmieren (PM1, PT1, PM2), Datenbanken, Algorithmen und Datenstrukturen, Automaten und formale Sprachen, Software-Engineering	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen die Grundlagen und Paradigmen der Künstlichen Intelligenz • können zu Grunde liegende Mechanismen (z.B. wissensbasierte Methoden, Heuristiken etc.) anwenden • erkennen Probleme, die zu komplex und/oder unscharf spezifizierbar für geschlossene algorithmische Lösungen sind • können reale Probleme adäquat repräsentieren/modellieren • besitzen Kenntnisse in Modellierung von Such- und Planungsprobleme, unscharfe Verfahren • können für reale Probleme die Anwendbarkeit verschiedener KI Verfahren bewerten und in einer für das ausgewählte Verfahren adäquaten Darstellung/Repräsentation modellieren und lösen		
Inhalte	• Grundlagen und Paradigmen der Künstlichen Intelligenz • Anwendungsgebiete intelligenter Systeme • Logik und logische Programmierung • logikbasierte Modellbildung • Planen und Suchen (uninformiert, informiert; Constraints, A* und verwandte Verfahren, approximative Verfahren wie Antime) • Ausgewählte Themen aus verschiedenen Bereichen: Robotik, (Multi-)Agentensysteme, • natürlichsprachliche Systeme, Repräsentation und Verarbeitung von Unschärfe, • Neuronale Netze, maschinelles Lernen und Data Mining, Semantische Netze		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Stuart Russell, Peter Norvig: Künstliche Intelligenz – Ein moderner Ansatz, Pearson, 2012 • Nils J. Nilsson: Artificial Intelligence: a new synthesis, Morgan Kaufmann, 1998 • Uwe Lämmel, Jürgen Cleve: Künstliche Intelligenz, Hanser, 2012 • Ingo Boersch, Jochen Heinsohn, Rolf Socher: Wissensverarbeitung: Eine Einführung in die Künstliche Intelligenz für Informatiker und Ingenieure, Spektrum, 2007 • Rudolf Kruse et. al.: Computational Intelligence: Eine methodische Einführung in Künstliche Neuronale Netze, Evolutionäre Algorithmen, Fuzzy-Systeme und Bayes-Netze., Vieweg+Teubner, 2011 • Ivan Bratko: PROLOG – Programming for Artificial Intelligence, 2011		

Modulbezeichnung	**Software-Engineering II**	Kürzel	SE2/SEP2
Lehrveranstaltung(en)	Vorlesung: Software-Engineering II Praktikum: Software-Engineering II	Semester	4
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Stefan Sarstedt	SWS	3+1
Dozenten	Prof. Dr. Stefan Sarstedt, Prof. Dr. Bernd Kahlbrandt, Prof. Dr. Olaf Zukunft, Prof. Dr. Thiel-Clemen, Prof. Dr. Thomas Lehmann	Sprache	deutsch
Voraussetzungen	Software-Engineering I (SE1, SEP1), Datenbanken (DB, DBP), Programmieren I+II (PR1, PRP1, PR2, PRP2)	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können wichtige Begriffe im Kontext von Softwareprojekten darstellen • können unterschiedliche Vorgehens- und Prozessmodelle und deren Einsatzbereiche erläutern • können ausgewählte Vorgehens- und Prozessmodelle anwenden • können typische Aufgaben eines Projektleiters benennen • können typische Tätigkeiten in verschiedenen Projektphasen (u.a. Planung und Kontrolle) beschreiben • können Aspekte des Qualitäts-, Risiko- und Konfigurationsmanagement darstellen • können ausgewählte Qualitätssicherungsmaßnahmen anwenden und Softwarequalität bewerten • können Software-Ergonomie als Bestandteil des Entwicklungsprojektes berücksichtigen		
Inhalte	• Begriffe im Kontext eines Softwareprojekts: Projekt, Projektphase, u.a. • Vorgehens- und Prozessmodelle in der Software-Entwicklung • Kostenschätzung • Projektplanung • Projektcontrolling • Risikomanagement • Qualitätssicherung • Konfigurationsmanagement • Software-Ergonomie		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Rechnerpräsentationen, freiwillige Übungsaufgaben, Gruppenarbeit Praktikum: Aufgabenbearbeitung in Gruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Grundlagenbücher: s. Bibliothek • Tom DeMarco: Bärentango: Mit Risikomanagement Projekte zum Erfolg führen. Hanser, 2003. • Tom DeMarco: Vom Mythos des Mann-Monats. Essays zum Software-Engineering. Mitp-Verlag, 2003. • Dirk W. Hoffmann. Software-Qualität. Springer Vieweg, 2013. • Ian Sommerville. Software Engineering. Pearson Education, 2012. • Jochen Ludewig und Horst Lichter. Software Engineering. Grundlagen, Menschen, Prozesse, Techniken. dpunkt Verlag, 2010. • eigene Skripte der Dozenten		

Modulbezeichnung	**Rechnernetze**	Kürzel	RN/RNP
Lehrveranstaltung(en)	Vorlesung: Rechnernetze Praktikum: Rechnernetze	Semester	4
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Martin Hübner	SWS	3+1
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Hans Heinrich Heitmann, Prof. Dr. Thomas Schmidt	Sprache	deutsch
Voraussetzungen	Programmieren, Betriebssysteme	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen die Konzepte und die Funktionsweise von Rechnernetzen • können grundlegende Konzepte zur Perfornanceanalyse von Rechnernetzen anwenden • können auf der Socket-Schnittstelle basierende Client- / Server-Anwendungen erstellen • können Methoden und Werkzeuge für die Konfiguration und Administration von Rechnernetzen anwenden		
Inhalte	• Architektur des Internet • Grundkonzepte der Datenübertragung • Wichtige Anwendungsschichtprotokolle (HTTP, SMTP, DNS) • Socket-Programmierung • Protokolle und Dienste der Netzwerk- und Transportschicht, insbesondere die TCP/IP-Protokollsuite • Protokolle der Sicherungsschicht • Grundlagen der LAN-Technologien • Sicherheit in Netzwerken		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• James F. Kurose, Keith W. Ross: Computernetze – Der Top-Down-Ansatz, Pearson Studium • Andrew S. Tanenbaum, David J. Wetherall: Computernetzwerke, Pearson Studium • Larry L. Peterson, Bruce S. Davie: Computernetze – Eine systemorientierte Einführung, dpunkt Verlag • Eigene Skripte der Dozenten		

Modulbezeichnung	**Architektur von Informationssystemen**	Kürzel	AI/AIP
Lehrveranstaltung(en)	Vorlesung: Architektur von Informationssystemen Praktikum: Architektur von Informationssystemen	Semester	5
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Stefan Sarstedt	SWS	3+1
Dozenten	Prof. Dr. Stefan Sarstedt, Prof. Dr. Olaf Zukunft	Sprache	deutsch
Voraussetzungen	Software-Engineering I+II (SE1, SEP1, SE2, SEP2), Datenbanken (DB, DBP), Programmieren I+II (PR1, PRP1, PR2, PRP2)	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen wichtige Begriffe im Kontext der Software-Architektur • können typische Aufgaben eines Software-Architekten benennen • kennen Heuristiken zur Gestaltung von Architekturen, Muster und Architekturstile und können diese anwenden • können Einflüsse von Randbedingungen und Risiken auf die Architekturgestaltung ableiten • können unterschiedliche Architektursichten anhand von Fallbeispielen erarbeiten • besitzen Kenntnisse in ausgewählten technischen Konzepten und können diese anwenden • können die Qualität einer Architektur bewerten		
Inhalte	• Begriffe im Kontext Software-Architektur • Aufgaben von Software-Architekten • Vorgehen bei der Architekturentwicklung • Sichten und Modelle • Ausgewählte Diagrammarten der UML • Heuristiken und Muster • Architekturstile • Ausgewählte technische Konzepte • Qualität von Architekturen • Beispielarchitekturen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Rechnerpräsentationen, freiwillige Übungsaufgaben, Gruppenarbeit Praktikum: Aufgabenbearbeitung in Gruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Grundlagenbücher: s. Bibliothek + eigene Skripte der Dozenten • Eric Evans: Domain-Driven Design: Tackling Complexity in the Heart of Software. Addison-Wesley Longman, 2003. • Craig Larman: Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. Prentice Hall, 2004. • Robert C. Martin: Agile Software Development, Principles, Patterns, and Practices. Prentice Hall International, 2011. • Robert C. Martin: Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall International, 2008. • Johannes Siedersleben. Moderne Software-Architektur. Umsichtig planen, robust bauen mit Quasar. dpunkt, Heidelberg, 2004. • Gernot Starke. Effektive Software-Architekturen. Ein praktischer Leitfaden. Carl Hanser Verlag GmbH & Co. KG, 2011.		

Modulbezeichnung	**Verteilte Systeme**	Kürzel	VS /VSP
Lehrveranstaltung(en)	Vorlesung: Verteilte Systeme Praktikum: Verteilte Systeme	Semester	5
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Christoph Klauck	SWS	3+1
Dozenten	Prof. Dr. Birgit Wendholt, Prof. Dr. Christoph Klauck	Sprache	deutsch
Voraussetzungen	Betriebssysteme, Rechnernetze	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • beherrschen die Grundlagen verteilter Systeme; • können selbständig eine System-Infrastruktur eines verteilten Systems entwerfen und realisieren; • können selbständig eine Middleware entwerfen und realisieren; • besitzen die Fähigkeit ein Konzept für replizierte Daten zu entwerfen und zu realisieren; • können für praxisorientierten Probleme verteilte Algorithmen entwerfen und realisieren; • besitzen die Fähigkeit die Möglichkeiten, Grenzen und Risiken verteilter Systeme zu bewerten		
Inhalte	• Eine Einführung im Sinne einer Beschreibung der charakteristischen Eigenschaften verteilter Systeme; • Interprozesskommunikation und Namensdienste; • Zeit , Koordination und Übereinstimmung; • Wahlen, Wechselseitiger Ausschluss und Verteilte Transaktionen; • Verteilte Dateisysteme und Replikation; • Fehlertoleranz;		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen, Selbständiges bearbeiten der Aufgaben, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• J. Armstrong, R. Virding, C. Wikström, M. Williams: Concurrent Programming in ERLANG • G. Bengel, C. Baun, M. Kunze, K.-U. Stucky: Masterkurs Parallele und Verteilte Systeme • Oliver Haase: Kommunikation in verteilten Anwendungen • G. Coulouris, J. Dollimore, T. Kindberg. Distributed Systems: Concepts and Design. • A.S. Tanenbaum, M.v. Stehen. Distributed Systems: Principles and Paradigms • Gerard Tel: Introduction to Distributed Algorithms • Nancy A. Lynch. Distributed Algorithms • Peter Mandl: Masterkurs Verteilte betriebliche Informationssysteme • A. Schill, T. Springer: Verteilte Systeme		

Modulbezeichnung	**IT-Sicherheit**	Kürzel	IST/ITSP
Lehrveranstaltung(en)	Vorlesung: IT-Sicherheit Praktikum: IT-Sicherheit	Semester	6
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Martin Hübner	SWS	3+1
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Gunter Klemke, NN	Sprache	deutsch
Voraussetzungen	Programmieren, Betriebssysteme, Rechnernetze, verteilte Systeme	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen Konzepte und Methoden zur Konstruktion von sicheren verteilten Systemen und können diese praktisch anwenden • verstehen Sicherheitsmodelle und Sicherheitseigenschaften von Betriebssystemen und können diese zum sicheren Betrieb von Anwendungen einsetzen • können Angriffstechniken in Netzwerken sowie den gezielten Einsatz von Abwehrmaßnahmen beurteilen		
Inhalte	• Einführung in das Security Engineering • Angriffstechniken • Grundlagen der Kryptographie • Public Key Infrastrukturen (PKI) • Authentifikations- und Autorisationsmodelle und –verfahren • Sicherheitsprotokolle in Kommunikationsnetzen • Datenschutz		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Claudia Eckert: IT-Sicherheit, Oldenbourg Verlag • Klaus Schmeh: Kryptografie, dpunkt-Verlag • Charlie Kaufman, Radia Perlman, Mike Speciner: Network Security, Prentice Hall • Eigene Skripte der Dozenten		