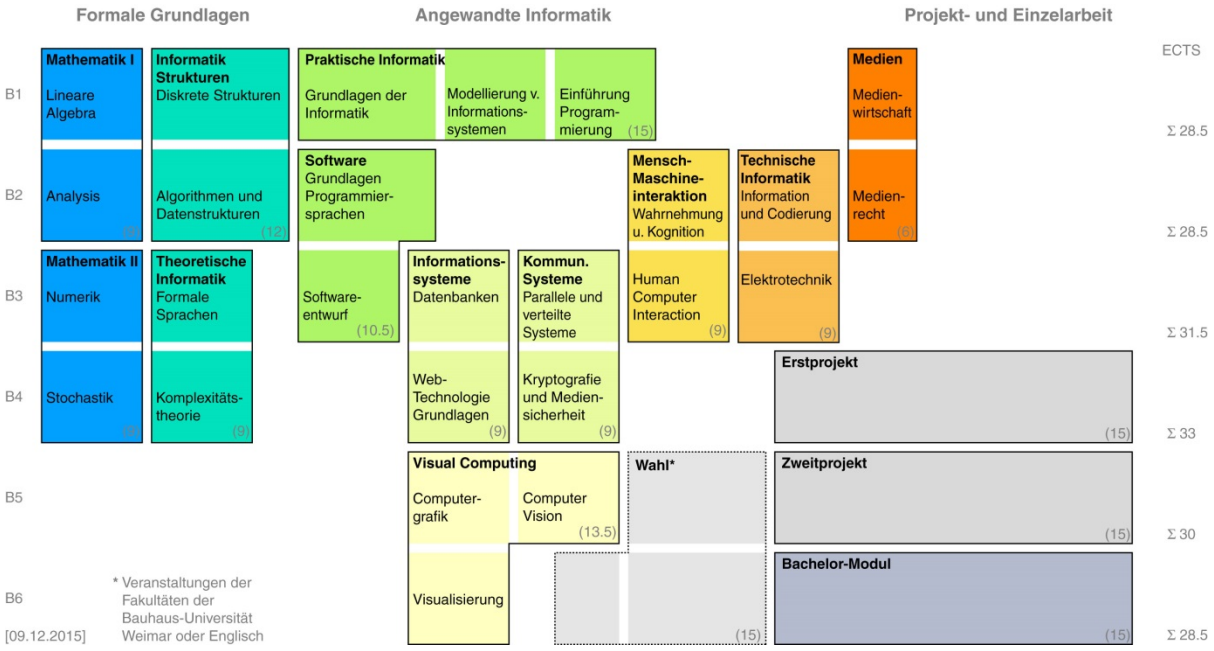
Modulkatalog Medieninformatik

Stand: 06. Februar 2019

Der Bachelor-Studiengang Medieninformatik ist forschungs- bzw. grundlagenorientiert, dauert 6 Semester (180 ECTS) und bereitet die Studierenden auf den Beruf des Informatikers/der Informatikerin mit einem Anwendungsschwerpunkt im Bereich der Digitalen Medien vor. Der Modulplan orientiert sich an den Empfehlungen der Gesellschaft für Informatik für sogenannte *Typ-2 Studiengänge*. Er ist untergliedert in Pflichtmodule, Wahl- bzw. Wahlpflichtmodule und die Anleitung zum wissenschaftlichen Arbeiten. Die folgende Abbildung gibt eine Übersicht über die Struktur des Studiums:



Pflichtmodule aus Informatik und Mathematik:

Der Studiengang beinhaltet 7 Pflichtmodule, die in den ersten 4 Semestern liegen und in denen die Grundlagen der Mathematik und Informatik vermittelt werden:

- Mathematik I (ab dem 1. Semester)
- Mathematik II (ab dem 3. Semester)
- Praktische Informatik (1. Semester)
- Informatik Strukturen (ab dem 1. Semester)
- Software (ab dem 2. Semester)
- Technische Informatik (ab dem 2. Semester)
- Theoretische Informatik (ab dem 3. Semester)

Anwendungsspezifische Pflichtmodule:

Im Lauf des Studiums werden zunehmend auch Kompetenzen mit einem dezidierten Anwendungsbezug zur Medieninformatik vermittelt. Dies geschieht im Rahmen der Anleitung

zur selbstständigen wissenschaftlichen Arbeit ab dem 4. Semester (siehe unten), aber auch schon ab dem 2. Semester im Rahmen der folgenden Pflichtmodule:

- Mensch-Maschine Interaktion (ab dem 2. Semester)
- Informationssysteme (ab dem 3. Semester)
- Kommunizierende Systeme (ab dem 3. Semester)
- Visual Computing (dem 5. Semester)

Wahlpflicht- und Wahlmodule:

In den ersten beiden Semestern ist das Medien-Modul verankert (6 ECTS). Es ist ein Wahlpflichtmodul, in dessen Rahmen den Studierenden empfohlen wird, jeweils eine Veranstaltung über Medienwirtschaft und Medienrecht für InformatikerInnen zu besuchen. Im 5. und 6. Semester liegt das Wahlmodul (15 ECTS), das den Studierenden ermöglicht, ihr Wissen durch den Besuch von Veranstaltungen aus anderen Bereichen und Fakultäten zu erweitern. Auch der Besuch von Englischveranstaltungen ist im Rahmen dieses Moduls möglich. Damit gibt es die folgenden Wahlpflicht- bzw. Wahlmodule:

- Medien (ab dem 1. Semester)
- Wahlmodul (ab dem 5. Semester)

Anleitung zur selbstständigen wissenschaftlichen Arbeit:

Bereits ab Beginn der zweiten Studienhälfte werden Studierende im Rahmen von Projekten in Kleingruppen an das selbstständige wissenschaftliche Arbeiten herangeführt. Die Projekte dienen auch dem Erwerb von berufsrelevanten „Soft Skills" wie Kommunikationskompetenz, Grundkenntnissen des Projektmanagements, ect. Im 6. Semester ist das Verfassen einer wissenschaftlichen Arbeit vorgesehen – natürlich unter Anleitung, durch einen Betreuer –, die, im Rahmen eines Vortrages vor den Mitgliedern der Fakultät, „verteidigt" wird. Insgesamt erfolgt die Anleitung zur selbständigen wissenschaftlichen Arbeit in drei Modulen:

- Erstprojekt (4. Semester)
- Zweitprojekt (5. Semester)
- Bachelor-Modul (6. Semester)

Leistungspunkte/Credits:

Den einzelnen Modulen sind jeweils ECTS-Leistungspunkte zugeordnet, entsprechend dem European Credit Transfer System. Sie sind quantitatives Maß für die zeitliche Belastung der Studierenden. In der Regel werden pro Studienjahr etwa 60 ECTS-Punkte vergeben. Nach nationalen und internationalen Standards (für Deutschland: Beschluss der Kultusministerkonferenz vom 24.10.1997) wird pro Leistungspunkt eine Arbeitsbelastung (workload) für Studierende im Präsenz- und Selbststudium von 30 Stunden angenommen. Etwa 60 ECTS-

Punkte im Jahr entsprechend damit einem Arbeitsaufwand von etwa 45 Wochen zu jeweils 40 Stunden.

Bewertung:

Jedes Modul wird mit einer Note abgeschlossen. Diese Note kann sich ggf. auf mehrere Teilleistungen stützen. Die Details dazu sind in den jeweiligen Modulbeschreibungen festgehalten.

Sieht ein Modul eine oder mehrere Teilleistungen vor, die als Klausur zu erbringen sind, kann die Prüfungsleistung abweichend im Rahmen einer mündlichen Prüfung erbracht werden, wenn einschlägige pädagogische Gründe vorliegen. Solche Gründe sind insbesondere:

- Eine sehr kleine Teilnehmerzahl (z.B. weniger als 10). Klausuren mit sehr wenigen Teilnehmern sind schwierig skalierbar. In solchen Fällen kann eine mündliche Prüfung das fairere Bewerten der Studierenden durch die Lehrenden erleichtern.
- Es handelt sich um die letzte Wiederholungsprüfung des Prüflings, und bei ihrem Nichtbestehen ist der Prüfling endgültig durchgefallen. Insbesondere Studierende, die besondere Schwierigkeiten mit der Prüfungsform der Klausur haben, können im Rahmen einer mündlichen Prüfung ggf. zeigen, dass sie die erwarteten Kompetenzen sehr wohl erworben haben.

Titel **Mathematik I** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
1 und 2	jährlich	(Siehe unten)	9	Präsenz (Vorlesung + Übung): 90 Selbststudium: 140 Prüfungsvorbereitung (inklusive Klausur): **40** **Summe 270**	Deutsch	Prof. Gürlebeck

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science		Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Das Modul zielt auf das Verständnis von Sachverhalten, Grundlagen und Methoden, die für ein Studium der Informatik bzw. Medieninformatik unverzichtbar sind. Den Studierenden wird ein Einstieg in die Grundlagen der Teilgebiete der Mathematik ermöglicht, die für Informatiker besonders wichtig sind. Zu diesem Zweck wird ein Teil Elementarmathematik vorgeschaltet, der die Studierenden auf ein einheitliches, die Studierfähigkeit garantierendes Niveau bringt. Ihnen werden Begriffe und Verfahren der Linearen Algebra und der Analysis vermittelt. Nach einem erfolgreichen Besuch der beiden Lehrveranstaltungen können die Studierenden konkrete Probleme mit Begriffen der Mathematik formulieren und analysieren, die wesentlichen Merkmale erfassen (Abstraktion) und mit Standardmethoden der Linearen Algebra und der Analysis geeignete Lösungsansätze entwickeln. Dabei werden sowohl die geistigen Grundtechniken Konkretisieren bzw. Spezialisieren als auch Verallgemeinern vermittelt. Spezielle Aufmerksamkeit wird dem Training des logischen Denkens und des korrekten Schließens gewidmet. Sie sind in der Lage, unter unterschiedlichen Problemlösungsansätzen bzw. unterschiedlichen Algorithmen einen geeigneten auszuwählen und diese Wahl nachvollziehbar zu begründen. Nicht zuletzt soll das Modul zur Förderung des objektiven und sicheren Denkens beitragen sowie zur Urteilsfähigkeit und Selbstkontrolle erziehen.

Spezielle Qualifikationsziele aus der Linearen Algebra:

- Die Studierenden sind sicher im Umgang mit den folgenden Begriffen und können sie aktiv verwenden:
  - Mengen und ihre Operationen
  - Grundbegriffe der Logik
  - Natürliche Zahlen, ganze Zahlen, rationale Zahlen, komplexe Zahlen
  - Funktionen und Abbildungen
  - Elementare Funktionen und ihre Umkehrfunktionen
  - Vektoren in 2 und 3D, Vektoralgebra
  - Vektoren im R^n
  - Geometrie im R^n, Cauchy-Schwarzsche Ungleichung, allgemeine Dreiecksungleichung, Orthogonalität
  - Basis, Dimension, lineare Unabhängigkeit
  - Teilraum, lineare Mannigfaltigkeit

- Orthogonalisierung von Vektoren, Projektionen
- Euklidische Vektorräume
- Matrizen und Matrixoperationen
- Matrizen als lineare Abbildungen, Darstellungsmatrix
- Rang, Determinante, Regularität, inverse Matrix
- Lösung linearer Gleichungssysteme
- Matrizeneigenwertprobleme

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:
  - Qualitative und quantitative Charakterisierung elementarer Funktionen
  - Analytische Geometrie der Ebene und des Raumes
  - Anwendung der Theorie linearer Vektorräume zur Beschreibung geometrischer Objekte in höheren Dimensionen
  - Beschreibung und Konstruktion spezieller linearer Abbildungen
  - Lösung linearer Gleichungssysteme
  - Bestimmung der Eigenwerte einer Matrix und Charakterisierung ihrer Invarianten

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:
  - Lösung komplexer geometrischer Probleme in der Ebene und im Raum
  - Koordinatentransformationen
  - Auswahl und Anpassung entsprechender Algorithmen
  - Konstruktionen spezieller Abbildungsmatrizen
  - Modellierung komplexer linearer Systeme
  - Lösung entsprechender linearer Gleichungssysteme
  - Bestimmung der Invarianten linearer Systeme

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten
  - der linearen Algebra vereinfachend beschreiben, algorithmisch bearbeiten und adäquate Datenstrukturen auswählen und zur Lösung auf dem Computer vorbereiten.

Sie können formalisieren und abstrahieren, konkrete Fragestellungen analysieren, systematisch nach möglichen mathematischen Lösungen suchen und selbst entsprechende Algorithmen entwickeln und zur Implementation auf dem Computer vorbereiten. Sie sind in der Lage, die Auswahl der Verfahren zu beurteilen und die erhaltenen Ergebnisse qualitativ und quantitativ zu bewerten.

Spezielle Qualifikationsziele aus der Analysis:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:
  - Zahlenfolgen und Zahlenreihen
  - Grenzwert und Grenzwertsätze
  - Konvergenz von Zahlenfolgen und Zahlenreihen
  - Konvergenzkriterien
  - Funktionsbegriff; Stetigkeit und Differenzierbarkeit
  - Bestimmung von Nullstellen
  - Iterative Lösung nichtlinearer Gleichungen, Intervallhalbierung, Newtonverfahren, Banachscher Fixpunktsatz
  - Approximation von Funktionen
  - Taylorsche Formel
  - Funktionenreihen – Taylorreihe

- Bestimmtes und unbestimmtes Integral
- Zusammenhang von Integration, Flächeninhalt und Stammfunktion
- Integrationsregeln
- Approximation von Integralen
- Funktionenreihen – Fourierreihen

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:

  - Berechnung des Grenzwertes von Zahlenfolgen und Zahlenreihen
  - Analyse des Konvergenzverhaltens
  - Klassifikation von Funktionen
  - Berechnung von Extremwerten, Nullstellen, Fixpunkten nichtlinearer Funktionen
  - Beschreibung von Kurven und Flächen und ihre geometrische Analyse
  - Entwicklung von Funktionen in Potenzreihen
  - Entwicklung von Funktionen in trigonometrische Reihen
  - Fehleranalyse für die Partialsummen dieser Reihen
  - Berechnung von Kurvenlängen, Flächeninhalten und Volumina
  - Bestimmung von Stammfunktionen
  - Elementare Aufgaben der Differentialgeometrie nach entsprechender Aufbereitung
  - Konstruktion und Charakterisierung allgemeiner krummliniger Koordinatensysteme

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:

  - Berechnung von Flächeninhalten und Volumina kompliziert geformter Körper
  - Analyse von Signalen durch Fourierreihen
  - Approximation von Funktionen durch Potenzreihen
  - Iterative Lösung nichtlinearer Gleichungen

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten

  - der Analysis der Funktionen einer oder mehrerer Veränderlicher,
  - der Differential und Integralrechnung,
  - durch Reihenentwicklungen

  formalisieren, konkrete Fragestellungen analysieren, systematisch nach möglichen Lösungen suchen und selbst entsprechende Algorithmen entwickeln. In einfachen Fällen können sie die Angemessenheit der angewandten Verfahren und die Aussagekraft der erzielten Ergebnisse bewerten und Fehlerabschätzungen erarbeiten.

- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden der Differential- und Integralrechnung der Funktionen einer oder mehrerer Veränderlicher zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

Lehrinhalte

- Lineare Algebra

  - Zahlenbereiche, Axiomatik, Rechengesetze.
  - Einführung in die Mengenlehre
  - Grundlagen der Logik
  - Funktionen und Abbildungen
  - Elementare Funktionen und ihre Umkehrfunktionen
  - Vektorrechnung im R^2 und R^3, analytische Geometrie der Ebene und des Raumes
  - Vektorraum R^n und allgemeine Euklidische Vektorräume

- Skalarprodukt, Cauchy-Schwarz Ungleichung und ihre Anwendung auf die Lösung geometrischer Probleme
- Orthogonalisierungsverfahren
- Matrizen und Matrixoperationen
- Matrizen und lineare Abbildungen
- Determinante, Rang
- Lineare Gleichungssysteme
- Eliminationsverfahren, Cramersche Regel
- Matrixeigenwertprobleme
- Invarianten einer Matrix
- Koordinatentransformationen

- Analysis

  - Zahlenfolgen
  - Grenzwert
  - Konvergenz
  - Zahlenreihen
  - Stetige Funktionen
  - Differenzierbare Funktionen
  - Nullstellen, Fixpunkte, Extremwerte
  - Bestimmtes und unbestimmtes Integral
  - Taylorpolynom und Taylorreihe, Potenzreihen
  - Fourierreihen
  - Beschreibung von Kurven und Flächen, geometrische Charakterisierung

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Sie werden im Rahmen einer Übung vertieft. Dabei sind zunächst im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Nach der Korrektur der Übungsaufgaben durch die Betreuenden werden die Lösungen in einer gemeinsamen Sitzung besprochen (nun wieder Präsenzstudium).). Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen. Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden. Zur Selbstkontrolle werden studienbegleitende Kurztests angeboten.

Bestimmte Lehrinhalte überlappen sich mit Lehrinhalten aus anderen Modulen. Dieses Vorgehen soll eine Orientierung in der Stofffülle vermitteln und Zusammenhänge erkennbar machen. Es basiert auf einem didaktischen Konzept, das sich an das Spiralprinzip aus der Schuldidaktik anlehnt.

Da sich gezeigt hat, dass viele StudienanfängerInnen Schwierigkeiten beim Einstieg in die Mathematik haben, wurde ein Mathematik-"Liftkurs" in die Veranstaltung Lineare Algebra integriert. Der Liftkurs ermöglicht es den Studierenden, beim Studienanfang ihr Mathematik-Schulwissen wieder aufzufrischen.

Hinweise

Literatur:

- Lothar Papula: Mathematik für Ingenieure und Naturwissenschaftler, Band 1,2,3. Vieweg+Teubner Verlag
- Burg, Haf, Wille, Höhere Math. für Ingenieure, Bde 1-2, Vieweg+Teubner Verlag
- Jänich, Lineare Algebra,Springer

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Lineare Algebra (wöchentlich im Wintersemester) • Analysis (wöchentlich im Sommersemester)	• 4.5 ECTS-Punkte (SWS: V3+Ü2) • 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Mathematik II** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
3 und 4	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Sommer- und einer im Wintersemester	9	Präsenz (Vorlesung + Übung): **67,5** Selbststudium: **172,5** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 270**	Deutsch	Prof. Gürlebeck

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science	Inhalte von Mathematik I	Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen aufeinander aufbauenden Lehrveranstaltungen in separaten Prüfungen geprüft. Mündliche Prüfung in Numerik. Klausur in Stochastik. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben sind eine Voraussetzung für die Zulassung zu den Prüfung. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Das Modul zielt auf das Verständnis von Sachverhalten, Grundlagen und Methoden, die für ein Studium der Informatik bzw. Medieninformatik unverzichtbar sind. Den Studierenden wird ein Einstieg in Teilgebiete der Mathematik ermöglicht, die für Informatiker besonders wichtig sind. Ihnen werden Begriffe und Verfahren der Numerischen Mathematik und der Stochastik vermittelt. Nach einem erfolgreichen Besuch der beiden Lehrveranstaltungen können die Studierenden konkrete Probleme mit Begriffen der Mathematik formulieren und analysieren, die wesentlichen Merkmale erfassen (Abstraktion) und mit Standardmethoden der Numerik und Stochastik geeignete Lösungsansätze entwickeln. Dabei werden sowohl die geistigen Grundtechniken Konkretisieren bzw. Spezialisieren als auch Verallgemeinern vermittelt. Sie sind in der Lage, unter unterschiedlichen Problemlösungsansätzen bzw. unterschiedlichen Algorithmen einen geeigneten auszuwählen und diese Wahl nachvollziehbar zu begründen. Nicht zuletzt soll das Modul zur Förderung des objektiven und sicheren Denkens beitragen sowie zur Urteilsfähigkeit und Selbstkontrolle erziehen.

Spezielle Qualifikationsziele aus der Numerik:

- Die Studierenden sind sicher im Umgang mit den folgenden Begriffen und können sie aktiv verwenden:
  - Computerzahlen, Rechnen mit Computerzahlen
  - Rundungsfehler für Grundrechenarten, Funktionen einer und mehrerer Veränderlicher und für Matrixoperationen
  - Interpolation
  - Steigungen
  - Interpolationsfehler
  - Splinefunktionen, Basissplines
  - Approximation
  - Finite Differenzen, numerische Differentiation

- Taylorabgleich
- Numerische Integration, Quadraturformeln, Exaktheitsgrad, Stabilität
- Stabilität

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:
  - Rundungsfehler und ihre Fortpflanzung in elementaren Rechnungen zu beschreiben und abzuschätzen
  - Die Fehler beim numerischen Lösen von Gleichungssystemen zu beurteilen
  - Messdaten zu interpolieren bzw. zu approximieren sowie Auswahl einer geeigneten Methode
  - Aufwandsanalyse für Interpolation und Approximation
  - Ableitungen erster und höherer Ordnung zu approximieren
  - Funktionen numerisch zu integrieren

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:
  - Rundungsfehleranalyse komplexer Berechnungen und Anpassung entsprechender Algorithmen basierend auf der Abschätzung der Konditionszahlen.
  - Darstellung von Signalen und Messdaten mit Hilfe geeignet ausgewählter Interpolationsverfahren bzw. auf der Basis von Approximationsverfahren.
  - Erkennen von Querbezügen zur Stochastik
  - Approximation von Ableitungen direkt oder im Zusammenhang mit der Lösung von Differentialgleichungen, näherungsweise Lösung von Problemen aus der Analysis der Funktionen einer oder mehrerer Veränderlicher
  - Numerische Integration im Zusammenhang mit Längenberechnungen, Flächenberechnungen, Fourierkoeffizienten, Stammfunktionen

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten
  - der Interpolations- und Approximationstheorie vereinfachend beschreiben, diskrete Daten glätten und die Fehlereinflüsse durch die Implementierung auf dem Computer erfassen und zum Teil steuern
  - Probleme aus der Differential- und Integralrechnung diskretisieren

und formalisieren, konkrete Fragestellungen analysieren, systematisch nach möglichen mathematischen Lösungen suchen und selbst entsprechende

Algorithmen entwickeln und zur Implementation auf dem Computer vorbereiten. Sie sind in der Lage, die Auswahl der Verfahren zu beurteilen und

die erhaltenen Ergebnisse qualitativ und quantitativ zu bewerten.

Spezielle Qualifikationsziele aus der Stochastik:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:
  - Zufällige Ereignisse und relative Häufigkeiten
  - Grundbegriffe der Kombinatorik
  - Axiomatische Definition der Wahrscheinlichkeit, verschiedene Wahrscheinlichkeitsmodelle
  - Bedingte Wahrscheinlichkeit und stochastische Unabhängigkeit
  - Zufallsgrößen und Verteilungsfunktionen
  - Wichtige Beispiele diskreter Zufallsvariabler (Binomialverteilung, Poissonverteilung,…)
  - Wichtige Beispiele stetiger Zufallsvariabler, insbesondere Gaußsche Normalverteilung
  - Erwartungswerte und Varianzen
  - Stochastische Konvergenz und Grenzwertsätze
  - Beschreibende Statistik: Verschiedene Skalenniveaus der Merkmale, Grafische Darstellung, Boxplots
  - Verschiedene Lage- und Streuungsparameter, deren Berechnung und Eigenschaften
  - Zweidimensionale Darstellungen, Kovarianz und Korrelation
  - Lineare Regression

- Schließende Statistik: Stichprobenfunktionen
- Punktschätzungen
- Intervallschätzungen, Zusammenhang mit Parametertests

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:

  - Die Struktur zufälliger Ereignisse mittels Ereignisoperationen analysieren
  - Die Sätze über bedingte Wahrscheinlichkeiten in konkreten Situationen einsetzen
  - Die Abhängigkeit bzw. Unabhängigkeit von Ereignissen zu erkennen (insbesondere Bernoulli-Ketten)
  - Den Typ der Verteilung einer konkreten Zufallsgröße erkennen
  - Die zur Beschreibung einer Verteilung notwendigen Parameter identifizieren und bei Bedarf zu schätzen
  - Vorliegende Daten anschaulich, übersichtlich und kompakt darzustellen
  - Das Skalenniveau eines konkreten Merkmals zu erkennen
  - Zweidimensionale Merkmale übersichtlich darzustellen und mittels Korrelations- und Regressionsanalyse zu untersuchen
  - In einfachen Fällen von Eigenschaften eines Merkmals in der Stichprobe auf Gesetzmäßigkeiten schließen, die dem Gesamtvorgang zu Grunde liegen

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:

  - Die verschiedenen Möglichkeiten der Approximation von Verteilungen effizient einsetzen
  - Den notwendigen Stichprobenumfang für eine statistische Analyse abzuschätzen
  - Bei nur wenig Information über einen zufälligen Vorgang die Grenzwertsätze anwenden
  - Verhältnis von Sicherheit und Genauigkeit bei statistischen Schätzverfahren abwägen

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten

  - der Wahrscheinlichkeitsrechnung,
  - der Theorie der Zufallsgrößen,
  - der beschreibenden Statistik und
  - der schließenden Statistik

  formalisieren, konkrete Fragestellungen analysieren, systematisch nach möglichen Lösungen suchen und selbst entsprechende Algorithmen entwickeln. In einfachen Fällen können sie die Angemessenheit der angewandten Verfahren und die Aussagekraft der erzielten Ergebnisse bewerten.

- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden

  - der Wahrscheinlichkeitsrechnung,
  - der Theorie der Zufallsgrößen,
  - der beschreibenden Statistik und
  - der schließenden Statistik

  zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

Lehrinhalte

- Numerik

  - Rechnen mit Computerzahlen
  - Rundungsfehler, Fortpflanzung und Abschätzung, relative Konditionszahlen
  - Konditionszahl einer Matrix, Einfluss auf numerische Lösung linearer Gleichungssysteme
  - Interpolation; Lagrange-, Newton-, Hermite-, Spline- und trigonometrische Interpolation
  - Abschätzung des Interpolationsfehlers
  - Approximation; beste Approximation, kleinste Fehlerquadratmethode

- Numerische Differentiation; Konstruktionsprinzipien, Taylorabgleich, Fehlerabschätzung
- Numerische Integration; elementare Formeln vom Newton und Newton-Cotes Typ, Gauß-Quadraturformeln, Integration periodischer Funktionen

- Stochastik
  - Zufallsereignisse und deren Wahrscheinlichkeit
  - Bedingte Wahrscheinlichkeit und Unabhängigkeit von Zufallsereignissen
  - Verteilungen diskreter und stetiger Zufallsgrößen
  - Summen unabhängiger Zufallsgrößen und zentraler Grenzwertsatz
  - Beschreibende Statistik
  - Schließende Statistik, Parameter- und Intervallschätzungen, statistische Tests
  - Korrelation und Regression

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Sie werden im Rahmen einer Übung vertieft. Dabei sind zunächst im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Nach der Korrektur der Übungsaufgaben durch die Betreuenden werden die Lösungen in einer gemeinsamen Sitzung besprochen (nun wieder Präsenzstudium). Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen. Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden.

Bestimmte Lehrinhalte überlappen sich mit Lehrinhalten aus anderen Modulen. Zum Beispiel werden Mengenoperationen (hier: Ereignisoperationen) und diskrete Wahrscheinlichkeiten auch im Modul Informatik-Strukturen behandelt. Dieses Vorgehen soll eine Orientierung in der Stofffülle vermitteln und Zusammenhänge erkennbar machen. Es basiert auf einem didaktischen Konzept, das sich an das Spiralprinzip aus der Schuldidaktik anlehnt.

Hinweise

Literatur:

- Lothar Papula: Mathematik für Ingenieure und Naturwissenschaftler, Band 1,2,3. Vieweg+Teubner Verlag
- Gerald Teschl: Mathematik für Informatiker, Band 1,2. Springer-Verlag
- H. Schwetlick, H. Kretzschmar: Numerische Verfahren für Naturwissenschaftler und Ingenieure, Fachbuchverlag Leipzig
- W. Dahmen, A. Reusken: Numerik für Ingenieure und Naturwissenschaftler, Springer

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Numerik (wöchentlich im Wintersemester) • Stochastik (wöchentlich im Sommersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1) • 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Praktische Informatik** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
1	jährlich	Wöchentlich im Sommersemester	15	Präsenz (Vorlesung + Übung): **112.5** Selbststudium (einschließlich Tutorien, falls angeboten): **217.5** Prüfungsvorbereitung einschließlich der Prüfung: **20** **Summe 450**	Deutsch	

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science		Die Form der Prüfung für die einzelnen Lehrveranstaltungen orientiert sich an dem Typ des in der Veranstaltung vermittelten Fachwissens. Die Prüfungsleistung der Lehrveranstaltung ‚Grundlagen der Informatik' ist eine Klausur. Dauer: 100 Minuten. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zur Prüfung. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die Veranstaltung ‚Einführung durch die Modellierung' wird durch bewertete Hausaufgaben geprüft. Die Veranstaltung ‚Einführung in die Programmierung' wird durch bewertete Programmieraufgaben und die bewertete Vorstellung der Programmierprojekte geprüft. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Im Rahmen des Moduls werden die zentralen Grundlagen der Informatik vermittelt, welche für ein Studium der Informatik bzw. Medieninformatik besonders wichtig sind. Den Studierenden wird ein Einstieg in die Begriffe und Methoden der Analyse von Problemen und deren Modellierung

und algorithmischen Lösung, sowie deren Umsetzung in ein Programm vermittelt. Nach einem erfolgreichen Besuch der Lehrveranstaltungen dieses Moduls können die Studierenden konkrete einfache Aufgaben mit Begrifflichkeiten und Methoden der Informatik darstellen und mit Hilfe eines Programms lösen.

Spezielle Qualifikationsziele aus den Grundlagen der Informatik

Die Studierenden kennen die grundlegenden Prinzipien einer problemorientierten Vorgehensweise eines Informatikers: algorithmisches Denken ausgehend von klaren Spezifikationen. Darüber hinaus kennen die Studierenden typische Programmierparadigmen und grundlegende Begrifflichkeiten wie Felder, lineare Suche, binäre Suche, Sortierverfahren sowie Rekursion und O-Notation und deren typische Anwendungsfälle. Außerdem sind grundlegende Konzepte der Rechnerarchitektur (von-Neumann-Modell) und von Betriebssystemen (Ressourcenverwaltung, Shells) bekannt und in ihren Aufgaben erfasst.

Die Studierenden verstehen, wozu algorithmisches Denken eingesetzt wird und warum es als zentrales Handwerkszeug eines Informatikers gilt. Sie verstehen grundlegende Fragestellungen wie Suchen, Sortieren und rekursive Problemlöseansätze und die Wichtigkeit strukturierten Vorgehens bei der Problemanalyse und genauer Analysen zur Effizienzbewertung verschiedener Lösungsansätze. Zusätzlich verstehen die Studierenden den Zusammenhang zwischen Rechnermodell, praktischer Umsetzung in Hardware, und den Verwaltungsaufgaben eines

Betriebssystems. Aus dem Umfeld der Computerarchitektur kennen die Studierenden insbesondere die folgenden Begriffe und Zusammenhänge und können diese anderen erklären: Funktionsprinzipien von informationsverarbeitenden Geräten, Begriff Gerät, Architektur, Ubicomp, mobile Systeme, Abstraktionseben von Software und Hardware, Hardware-Schichtenmodell, Moorsches Gesetz, Perspektiven, Siliziumtechnologie und künftige Alternativen, Speichersystem, Anforderungen und Grundstrukturen, Arbeitsprinzipien und Technologie, Register, Cache, Hauptspeicher, Virtueller Speicher, Externer Speicher, Kommunikation, Systematik, interne und externe Bussysteme, Interfaces.

Die Studierenden können einfache Problemstellungen spezifizieren, algorithmisch erfassen und in Pseudocode umsetzen. Vorgegebene Lösungsansätze können unter Effizienzgesichtspunkten analysiert und ggf. leicht verbessert werden. Einfache Abschätzungen zur Performanceverbesserung durch Parallelisierbarkeit können durchgeführt und berechnet werden. Die Studierenden sind in der Lage die Effizienz der Speicherhierarchien und Parallelisierungen nach Amdahl zu analysieren. Sie kennen externe Schnittstellen der Medieninformatik, und sind in der Lage diese problemspezifische auszuwählen.

Qualifikationsziele der Modellierung von Informationssystemen:

Die Studierenden kennen Grundbegriffe der Modelltheorie (Verkürzung, Pragmatik, Abbild) sowie zentrale Modellierungsmethoden der Informatik (Daten-, Steuer-, Verhaltens-, sowie Interaktionsmodelle, Klassen und Objekte, Vererbung, Logik).

Die Studierenden verstehen, warum Modellbildung ein zentraler Bestandteil der Informatik ist. Sie verstehen, dass jeder Modellierungsprozess einem bestimmten Modellierungszweck dient und dass Modellbildung keine wertfreie Tätigkeit ist. Sie verstehen die Wichtigkeit einer definierten Notation für die Modellierung.

Sie können vorgegebene einfache Diagramme (Entitity Relationship, Flowchart, Zustandsautomat, Klassenmodell, etc.) lesen bzw. interpretieren und können solche Diagramme für einfache Beispiele selber erstellen.

Spezielle Qualifikationsziele der Einführung in die Programmierung:

Die Studierenden kennen die Grundlagen, Konzepte und Eigenschaften der Programmiersprache Java, dieses umfasst insbesondere Datentypen, Variablen, Ausdrücke, Operatoren, Kontrollstrukturen, Blöcke, Methoden, Klassen, Objekte, Vererbung, Pakete, Schnittstellen und die Oberflächenprogrammierung mit Swing. Sie haben ein grundsätzliches Verständnis für die Architektur und die Funktionsweise der Virtuellen Maschine.

Die Studierenden können Classen für einfache Datentypen eigenständig implementieren und grundlegende Algorithmen für diese Datentypen in Methoden dieser Klassen umsetzen (z.B. für Komplexe Zahlen). Basierend auf dem Konzept der Vererbung können bestehende Datentyp-Klassen erweitert werden (z.B. Stacks). Die Studierenden können eine einfache Oberfläche zur Eingabe von Zeichenketten mit Hilfe von Swing implementiert.

Die Studierenden wissen wie in Kleingruppen einfache Schnittstellen entworfen werden und wie basierend auf diesen Schnittstellen Programme in einem Team entwickelt werden können.

Lehrinhalte

Grundlagen der Informatik

Die Veranstaltung gibt eine Einführung in die zentralen Konzepte der Informatik und die algorithmische Denkweise als Problemlösestrategie.

- Informatikbegriff und Teilgebiete der Informatik
- Algorithmusbegriff (Algorithmen und Spezifikationen, Übergang zu Programmiersprachen)
- Informationen und Daten (Bits und Bytes, Kodierung)
- Betriebssysteme(Ressourcen, Aufbau von Betriebssystemen, Treiber)
- Shell-Programmierung (Shell-Begriff, Shell-Befehle, Shell-Skripte)
- Imperative Programmierung (Arten von Programmiersprachen, Berechnungen, Steuerungsanweisungen)
- Suchen und Sortieren (Lineare Suche, Binäre Suche, Bubble-Sort, Insertion-Sort, Selection-Sort)
- Laufzeitverhalten (Laufzeit von Programmen, O-Notation,Einfache vs. schwierige Probleme)
- Rekursion (Begriff der Rekursion, Rekursives Sortieren, Merge-Sort, Quick-Sort)
- Objektorientierte Programmierung (Modularisierung von Software, Attribute, Methoden, Objektorientierte Modellierung)
- Rechnerarchitektur (von-Neumann-Modell)
- Funktionsprinzipien von informationsverarbeitenden Geräten, Begriff Gerät, Architektur, Ubicomp, mobile Systeme,
- Abstraktionseben Software und Hardware, Hardware-Schichtenmodell,
- Moorsches Gesetz, Perspektiven, Siliziumtechnologie und künftige Alternativen,
- Rechnerinterne Informationsdarstellung, Logikpegel, n-Tupel-Verarbeitung, Datentypen, Bustypen,
- Organisationsprinzip, Rechnergenerationen und -klassen, von-Neumann-Maschine und ihre Evolution,
- Taxonomie insbes. der Parallelverarbeitung, Flynn, Betriebsarten Bit-, Instruction-, Task-, Processor-Level,
- CPU- und Busstrukturen, CPU-Philosophien, Ebenen der Parallelverarbeitung, Registerstrukturen, Multimedia-Erweiterungen, Innovation, Technologie, Realisierung,
- Leistungsbewertung, Amdahl-Prinzip der Effizienz von Parallelstrukturen,

- Speichersystem, Anforderungen und Grundstrukturen, Arbeitsprinzipien und Technologie, Register, Cache, Hauptspeicher, Virtueller Speicher, Externer Speicher,
- Kommunikation, Systematik, interne und externe Bussysteme, Interfaces.

Modellierung von Informationsystemen

Die Veranstaltung gibt eine Einführung in zentrale Modellierungsmethoden der Informatik sowie Grundprinzipien der Modellierung.

- Der Modellbegriff, Grundprinzipien der Modellierung (Abstraktion, Klassifikation, Generalisierung, Komposition, Benutzung), Original-Abbild-Modell Relation, Verkürzung und Pragmatik der Modellierung, Modellbildung als Realitätskonstruktion,
- Logik als Modellierungssprache (einfache Aussagen- und Prädikatenlogik)
- Modellierungsmethoden: Datenmodelle (Entity Relationship), Verhaltensmodellierung (Zustandsautomaten und –diagramme, Statecharts, Petrinetze), Klassen- und Objektmodelle (mit UML Notation), Interaktionsmodelle (Kontextdiagramme, Anwendungsfälle / Use Cases, Szenarien, Aktivitätsdiagramme)

Einführung in die Programmierung

Die Veranstaltung gibt eine Einführung in die Implementierung von Programmen basierend auf der Programmiersprache Java.

- Datentypen, Variablen
- Arithmetische und Boolesche Ausdrücke
- Operatoren, Kontrollstrukturen, Blöcke, Methoden
- Klassen, Objekte, Vererbung
- Pakete, Schnittstellen
- Oberflächenprogrammierung mit Swing
- Umgang mit Programmierwerkzeug zur Entwicklung von Software

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert. Sie werden im Rahmen einer Übung vertieft. Dabei sind zunächst im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Nach der Korrektur der Übungsaufgaben durch die Betreuenden werden die Lösungen in einer gemeinsamen Sitzung besprochen (in Vorlesung oder Übung). Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen. Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden.

Über regelmäßig zu bearbeitende Übungsaufgaben wird eine kontinuierliche Aufarbeitung der Inhalte bewirkt.

Im Sinne des Spiralprinzips werden Inhalte, die in höheren Semestern im Detail behandelt werden, hier einführend behandelt.

Zur Reduzierung der Prüfungsbelastung und im Sinne einer Prüfung praktischer Fähigkeiten erfolgt für zwei der Fächer eine studienbegleitende Prüfungsform durch Haus- und Programmieraufgaben. Dies gilt insbesondere für die ‚Einführung in die Modellierung'. Diese Veranstaltung wird durch mehrere zu bearbeitete Hausaufgaben geprüft (maximal 6 verteilt über das Semester).

Die Vorlesung ‚Einführung in die Programmierung' zielt auf das Erlernen einer konkreten Programmiersprache und die Programmentwicklung in dieser Sprache. Daher finden die Übungen als betreute Lab-Stunden in einem Rechnerpool statt. Die Veranstaltung wird durch drei kleinere und eine größere (auch außerhalb der eingeplanten Lab-Stunden zu bearbeitende) über das Semester verteilte Programmieraufgaben geprüft.

Wissenschaftliche Mitarbeiter und studentische Tutoren aus höheren Semestern betreuen die Studierenden in den Übungen und stehen für Rückfragen und Diskussion zur Verfügung

Die ‚Grundlagen der Informatik' besteht aus einer wöchentlichen 135-minütigen Vorlesung (mit einer 15-minütigen Pause) sowie aus einer vierzehntägigen 90-minütigen Übung.

Die Veranstaltungen ‚Modellierung von Informationssystemen' und ‚Einführung in die Programmierung' bestehen aus jeweils einer wöchentlichen 90-minütigen Vorlesungen sowie einer Einzelstunde (45 min) Übung bzw. einer betreuten 45-minütigen Lab-Stunde im Rechnerpool.

Hinweise

Literatur zu Grundlagen der Informatik:

- Gumm, Sommer: Einführung in die Informatik. Oldenbourg Verlag.
- Hennessy, J.L.; Patterson, D.A.: Computer Architecture. A Quantiative Approach.
- Tanenbaum, A.: Structured Computer Organization.

Literatur zu Einführung in die Programmierung:

- Goll, Heinisch: Java als erste Programmiersprache: Ein professioneller Einstieg in die Objektorientierung mit Java, Springer Vieweg, 2013

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Grundlagen der Informatik • Modellierung von Informationssystemen • Einführung in die Programmierung	• 6.0 ECTS-Punkte (SWS: V3+Ü1) • 4.5 ECTS-Punkte (SWS: V2+Ü1) • 4.5 ECTS-Punkte (SWS: V1+Ü2)

Titel **Informatik Strukturen** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
1 und 2	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Sommer- und einer im Wintersemester	12	Präsenz (Vorlesung + Übung): **90** Selbststudium (einschließlich Tutorium, falls angeboten): **240** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 360**	Deutsch und Englisch	Stefan Lucks

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science	Zulassung zum Studium	Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Das Modul zielt auf den Erwerb der zentraler Grundlagen und Methoden, die für ein Studium der Informatik bzw. Medieninformatik unverzichtbar sind. Den Studierenden wir ein Einstieg in Teilgebiete der Mathematik vermittelt, die für Informatiker besonders wichtig sind, ihnen werden Begriffe und Methoden der Algorithmik vermittelt. Nach einem erfolgreichen Besuch der beiden Lehrveranstaltungen können die Studierenden konkrete Probleme mit Begrifflichkeiten der Mathematik abstrakt darstellen, mit Methoden der Mathematik algorithmische Lösungsansätze entwickeln, diese Algorithmen analysieren (vor allem in Bezug auf Laufzeit und Korrektheit) und die Algorithmen implementieren. Sie sind in der Lage unter unterschiedlichen Problemlösungsansätzen bzw. unterschiedlichen Algorithmen einen geeigneten auszuwählen und diese Wahl nachvollziehbar zu begründen.

Spezielle Qualifikationsziele aus den Diskreten Strukturen:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:
  - grundlegende Eigenschaften der natürlichen Zahlen
  - grundlegende Beweismethoden der Mathematik bzw. der Theoretischen Informatik: direkte und indirekte Beweise, vollständige Induktion
  - Endlichkeit, Abzählbarkeit, Über-Abzählbarkeit
  - elementare algebraische Algorithmen (Addition, Multiplikation, Potenzbildung)
  - die Laufzeiten dieser Algorithmen
  - den Zusammenhang von vollständiger Induktion und rekursiven Programmen
  - Syntax und Semantik einer einfachen Programmiersprache
  - Rechenregeln in der Menge der Restklassen mod n, Besonderheiten, wenn n prim ist, Inverse mod n
  - Grundlegende algebraische Strukturen (Gruppe, Ring, Körper)
  - Beispiele für die Anwendung der Arithmetik mod n in der Informatik (Prüfusummen, Public-Key Kryptographie)

- Probabilistische Algorithmen, insbesondere probabilistische Primzahltests
- Polynomringe und ihre Eigenschaften
- Endliche Körper und ihre Eigenschaften
- das Geburtstagsparadoxon
- Wesentliche Begriffe der Diskreten Wahrscheinlichkeit (Wahrscheinlichkeitsraum, Zufallsvariable, bedingte Wahrscheinlichkeit, …)
- Grundlegende Begriffe der Graphentheorie (gerichtete und ungerichtete Graphen, Zusammenhangskomponenten, Wege, Kreise, …)
- Beispiele für effizient lösbare Probleme der Graphentheorie (Eulerkreis, TSP-Approximation) und für NP-harte Probleme (Hamiltonkreis, TSP-Suchproblem)

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:

  - die grundlegenden Beweistechniken einsetzen, um einfache Beweise zu führen
  - entscheiden, ob eine gegebene Menge endlich, abzählbar oder über-abzählbar ist
  - einfache Verfahren zum Berechnen von Prüfsummen anwenden
  - einfache Algorithmen in einer einfachen Programmiersprache implementieren
  - mit Hilfe des Erweiterten Euklidischen Algorithmus feststellen, ob eine Zahl ein Inverses mod n hat, und wenn ja, dieses Inverse mod n berechnen
  - fehlererkennende Codes als Anwendung der Arithmetik in Polynomringen berechnen
  - Secret-Sharing-Verfahren als Anwendung der Arithmetik in endlichen Körpern einsetzen
  - einfache Aufgaben der Wahrscheinlichkeitsrechnung und der bedingten Wahrscheinlichkeit lösen

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:

  - mittelschwere Algorithmen in einer einfachen Programmiersprache implementieren (z.B. Simulation von Zufallsereignissen, probabilistische Primzahltests, Zeichnen von Fraktalen, …)
  - offen gestellte mathematische Fragestellungen mit Hilfe der oben genannten Beweistechniken beweisen oder, durch Angabe von Gegenbeispielen, widerlegen
  - selbstständig abstrakte Begriffe in mathematisch-formaler Darstellung verstehen sich selbständig in die Anwendung grundlegender Methoden einarbeiten
  - das WWW als gerichteten Graphen modellieren und die Funktion einer Suchmaschine auf Basis des Page-Rank Algorithmus nachvollziehen

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten

  - der Zahlentheorie,
  - der diskreten algebraischen Strukturen,
  - der diskreten Wahrscheinlichkeit und
  - der Graphentheorie

  formalisieren, formale Fragestellungen analysieren, systematisch nach möglichen Lösungen suchen und selbst entsprechende Algorithmen entwickeln. In einfachen Fällen können sie die Korrektheit von Algorithmen begründen und ihre Laufzeit grob abschätzen (linear, quadratisch, kubisch, …).

- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden

  - der Zahlentheorie,
  - der diskreten algebraischen Strukturen,
  - der diskreten Wahrscheinlichkeit und
  - der Graphentherie

  zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

Spezielle Qualifikationsziele aus dem Bereich Algorithmen und Datenstukturen:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:

- grundlegende Methoden der Organisation von Daten
- Analyse und Klassifikation von Algorithmen (Untersuchung der Leistungsfähigkeit von Algorithmen und Berechnungskomplexität im ungünstigsten Fall)
- Suchalgorithmen, Sortierverfahren und Algorithmen für Graphen
- Laufzeiten dieser Algorithmen und ihre Eigenschaften
- „Teile und Herrsche" Prinzip für die Entwicklung von Algorithmen
- Geometrische Algorithmen wie Bestimmung der konvexen Hülle und das Problem des nächsten Punktes
- Fluss in einem Netzwerk
- Mathematische Algorithmen (Zufallszahlen, Arithmetik, …)
- Verfahren zur Lösung des Problems der Datenanpassung (Interpolation mit Hilfe von Polynomen, Spline-Interpolation und Methode der kleinsten Quadrate)
- Einige NP-vollständige Probleme wie Erfassung von Knoten, Hamilton-Zyklus und das Problem des Handlungsreisenden

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden und zu programmieren:
  - die Wahl der richtigen Datenstruktur bei der Implementation der Algorithmen
  - die Leistungsfähigkeit von Algorithmen beurteilen und ihre Komplexität berechnen
  - die Wahl der geeigneten Algorithmen zur Lösung von Problemen und der Implementierung auf einem Computer
  - die Besonderheiten des Problems analysieren und eine gut angepasste Lösung finden
  - Algorithmen entwickeln und implementieren

- Die Studierenden können wesentliche Merkmale von
  - Suchalgorithmen
  - Algorithmen für Graphen
  - Geometrische Algorithmen
  - Sortieralgorithmen
  - Mathematische Algorithmen

  verstehen und Ihre Besonderheiten beachten, um sie anwenden zu können. Sie können entscheiden, welche Algorithmen für eine bestimmte Problemstellung geeignet sind, welche nicht, und sie können diese Entscheidung auch begründen.

- Die Studierenden sind in der Lage, ihr Wissen auch auf algorithmische Fragestellungen, jenseits der Suchalgorithmen, Algorithmen für Graphen, Geometrischen Algorithmen, Sortieralgorithmen und Mathematische Algorithmen anzuwenden.

Lehrinhalte

- Diskrete Strukturen
  - Beweistechniken
  - Abzählbarkeit und Über-Abzählbarkeit
  - Einführung in eine einfache Programmiersprache (Teilmenge von Python)
  - Restklassen mod n
  - Primzahlen
  - Gruppen
  - Ringe und Körper
  - Diskrete Wahrscheinlichkeit
  - Graphentheorie

- Algorithmen und Datenstrukturen
  - Datenstrukturen
  - Analyse von Algorithmen

- Hashing
- Suchalgorithmen
- Algorithmen für Graphen
- Geometrische Algorithmen
- Sortieralgorithmen
- Teile und Herrsche
- Mathematische Algorithmen
- NP-vollständige Probleme

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Sie werden im Rahmen einer Übung vertieft. Dabei sind zunächst im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Nach der Korrektur der Übungsaufgaben durch die Betreuenden werden die Lösungen in einer gemeinsamen Sitzung besprochen (nun wieder Präsenzstudium). Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen. Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden.

Bestimmte Lehrinhalte überlappen sich mit Lehrinhalten aus anderen Modulen. Zum Beispiel werden Beweismethoden auch in Mathematik I behandelt, und im Modul Praktische Informatik werden Programmierkenntnisse vermittelt. Dieses Vorgehen soll eine Orientierung in der Stofffülle vermitteln und Zusammenhänge erkennbar machen. Es basiert auf einem didaktischen Konzept, das sich an das Spiralprinzip aus der Schuldidaktik anlehnt.

Hinweise

Handouts und Literatur zu den Diskreten Strukturen:

- Lucks: Handout für den Start
- Lucks: Einfache Algorithmen mit Python
- Jukna: Crashkurs Mathematik für Informatiker
- Albertson, Hutchinson: Discrete Mathematics with Algorithms

Literatur zu Algorithmen und Datenstukturen:

- R. Sedgewick, „Algorithmen“
- M. Goodrich and R. Tamassia ‚Algorithm Design”

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Diskrete Strukturen (wöchentlich im Wintersemester) • Algorithmen und Datenstrukturen (wöchentlich im Sommersemester)	• 6 ECTS-Punkte (SWS: V3+Ü1) • 6 ECTS-Punkte (SWS: V2+Ü2)

Titel **Software** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
2 und 3	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Winter- und einer im Sommer-semester	10.5	Präsenz (Vorlesung + Übung): **90** Selbststudium: **195** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 315**	Deutsch	Bernd Fröhlich

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science	Modul „Praktische Informatik“	Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. Dauer: 90-120 Minuten In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Das Modul zielt auf den Erwerb zentraler Grundlagen und Methoden ab, die für die Entwicklung von Software unverzichtbar sind. Den Studierenden wird ein Einstieg in die moderne imperative, objektorientierte und generische Programmierung ermöglicht, und ihnen werden Begriffe und Methoden des Softwaredesigns vermittelt. Nach einem erfolgreichen Besuch der beiden Lehrveranstaltungen können die Studierenden konkrete Probleme mit Begrifflichkeiten der objektorientierten Programmierung abstrakt darstellen, mit Methoden des Softwareentwurfs Lösungsansätze entwickeln, diese Ansätze analysieren (vor allem in Bezug auf Vor- und Nachteile bezüglich konkreter Einsatzszenarien) und die Lösungsansätze implementieren. Sie sind in der Lage unter unterschiedlichen Problemlösungsansätzen bzw. unterschiedlichen Prozessmodellen einen geeigneten auszuwählen und diese Wahl nachvollziehbar zu begründen.

Spezielle Qualifikationsziele aus den Grundlagen der Programmiersprachen:

- Die Studierenden kennen und beherrschen wesentliche Konzepte imperativer, objektorientierter und generischer Programmierung uns können diese erklären, anwenden und gegeneinander abgrenzen.
  1. Objekte, Zustand, Verhalten, Botschaften, Methoden, Klassen, Instanzen, Klassenhierarchien
  2. Klassendefinition, Konstruktor, Überladung, Initialisierung vs. Zuweisung, Deklaration vs. Definition, grundlegende Regeln für den Aufbau einer Klasse (class mechanics), Gültigkeitsbereich und Lebensdauer von Objekten
  3. const correctness als wichtiges Element zur sicheren Programmierung und als Interface-Versprechen: konstante Objekte, const Parameter und Methoden
  4. Übergabe- und Rückgabemechanismen für Methoden und Funktionen: Korrekte Verwendung von Übergabe und Rückgabe per value und per reference in Kombination mit const
  5. Templates, Standard Template Library in C++, Container, Iteratoren, Algorithmen
  6. Funktoren, anonyme Funktionen (Lambdas), Closures, Funktionen höherer Ordnung
  7. Speicherverwaltung, Stack, Freestore, Größe zusammengesetzter Objekte, globale und automatische Variablen, Zeiger, Smart Pointer, Zeigersemantik vs. Wertsemantik, Einsatz von Freestore-Objekten vs. automatische Variablen, Klassendesign für die Verwaltung dynamischer Ressourcen
  8. Klassenhierarchien und Vererbung, virtuelle Funktionen, Überschreiben, Interfaces und abstrakte Klassen, polymorphe Variablen, statische und dynamische Typisierung, Einsatz von Vererbung vs. Templates vs. Containment, Liskov-Prinzip
  9. Überladen vs. Überschreiben vs. Überdecken

- Für einfache Aufgabenstellungen können die Studierenden die passenden Konzepte imperativer, objektorientierter und generischer Programmierung auswählen und kombinieren sowie in modernem C++ umsetzen. Sie können die Auswahl begründen und die Funktionsweise der

Implementierung im Detail erklären.
- Die Studierenden kennen grundlegende Regeln für das robuste Design einzelner Klassen und Klassenhierarchien und können diese anwenden
- Die Studierenden haben grundlegende Programmiererfahrung durch die Übungen und ein abschließendes Miniprojekt zum Thema Ray Tracing erworben. Sie kennen sich mit einer Entwicklungsumgebung, Versionsverwaltung und Software Tests aus und können diese gekonnt einsetzen.

Spezielle Qualifikationsziele aus dem Software-Entwurf:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:
  1. Softwareprozessmodelle und -methoden
     1. traditionell: Wasserfallmodell, Spiralmodell
     2. agil: Scrum, Extreme Programming
     3. open-source: „Bazaar“-Modell
  2. Revision Control System & Build-System
  3. Unit Testing & Continuous Integration
  4. Design Patterns
     1. Creation: Abstract Factory, Factory Method, Singleton, Prototype
     2. Behaviour: Command, Iterator, Visitor, Observer
     3. Structure: Adapter, Facade, Proxy, Composite, Decorator
     4. UI Patterns
  5. Compiler, Linker, Library, Object, Debugger
  6. Qualitätsmetriken für Software
  7. Requirements Engineering

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:
  1. die behandelten Design Patterns in einer objektorientierten Programmiersprache implementieren (z.B. Composite zur Darstellung von Baumstrukturen)
  2. ein Lasten- und Pflichtenheft für ein Softwareentwicklungsprojekt erstellen
  3. User Stories und Tasks für einen agilen Softwareentwicklungsprozess erstellen
  4. ein objektorientiertes Softwaredesign in UML modellieren
  5. traditionelle (SVN) und verteilte (Git) Revisionskontrollsysteme sinnvoll einzusetzen

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:
  1. die Zusammenhänge zwischen bestimmten Design Patterns nachvollziehen (z.B. AbstractFactory, FactoryMethod & Prototype)
  2. ein gegebenes UML-Diagramm oder „Box&Line“-Diagramm analysieren und in der dargestellten Softwarearchitektur verwendete Konzepte identifizieren
  3. eine Anforderungsanalyse für ein gegebenes Software-Entwicklungsproblem durchführen und ein geeignetes Software-Prozessmodell auswählen
  4. gegebenen Source-Code analysieren, Qualitätsmetriken dafür bestimmen sowie darin verwendete Design Patterns (auch UI-Patterns) erkennen und beschreiben
  5. Fehler in einem bestehenden Softwareprojekt mit Hilfe eines Debuggers finden und analysieren

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten aus
  1. Unified Modelling Language (UML)
  2. Design Patterns
  3. Software-Prozessmodellen

  formalisieren, formale Fragestellungen analysieren, systematisch nach möglichen Lösungen suchen und selbst entsprechende Lösungsansätze entwickeln. Die Studierenden können in Anwendungsfällen zudem entscheiden und bewerten, welche dieser Konzepte und Methoden zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

- Zusätzlich werden soziale Fähigkeiten und Schlüsselqualifikationen durch Gruppenarbeit basierend auf konkreten Problemen und Aufgaben geübt.

Lehrinhalte

A. Grundlagen der Programmiersprachen
- A.I. Klassen und Klassenhierarchien
- A.II. Übergabe- und Rückgabemechanismen für Funktionen und Methoden
- A.III. const correctness
- A.IV. Speicherverwaltung und Zeiger
- A.V. generische Programmierung
- A.VI. Robustes Design einzelner Klassen und Klassenhierarchien

- Software-Entwurf
  - Best Practices der Softwareentwicklung

- Software-Prozessmodelle (traditionell & agil)
- Design Patterns für Architektur & Code
- Codequalität, Testing & Continuous Integration
- Build-Systeme, Compiler/Linker & Debugger
- Open-Source-Software
- Requirements Engineering

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Sie werden im Rahmen einer Übung vertieft. Dabei sind zunächst im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Nach der Korrektur der Übungsaufgaben durch die Betreuenden werden die Lösungen in einer gemeinsamen Sitzung besprochen (nun wieder Präsenzstudium). Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen. Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden.

Bestimmte Lehrinhalte überlappen sich mit Lehrinhalten aus anderen Modulen. Zum Beispiel werden Modellierungskonzepte auch im Modul Praktische Informatik behandelt. Dieses Vorgehen soll eine Orientierung in der Stofffülle vermitteln und Zusammenhänge erkennbar machen. Es basiert auf einem didaktischen Konzept, das sich an das Spiralprinzip aus der Schuldidaktik anlehnt.

Hausaufgaben umfassen ein Maximum an 14 Aufgabenzetteln verteilt über das gesamte Modul. Wissenschaftliche Mitarbeiter und studentische Tutoren aus höheren Semestern betreuen die Studierenden in den Übungen und stehen für Rückfragen und Diskussion zur Verfügung.

Nach der Korrektur der eingereichten Übungsaufgaben durch die Betreuenden werden diese mit Anmerkungen an die Studierenden zurückgegeben und beispielhafte Lösungen sowie typische Fehler werden in der Präsenzphase besprochen.

Die einzelnen Veranstaltungen bestehen aus wöchentlichen 90-minütigen Vorlesungen sowie einer Doppelstunde (90 min für Grundlagen der Programmiersprachen) bzw. Einzelstunde (45 min für Software-Entwurf) Übung.

Hinweise

Literatur zu Grundlagen der Programmiersprachen:

- B. Stroustrup: Programming: Principles and Practice Using C++

Literatur zu Software-Entwurf:

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Grundlagen der Programmiersprachen (wöchentlich im Sommersemester) • Software-Entwurf (wöchentlich im Wintersemester)	• 6.0 ECTS-Punkte (SWS: V2+Ü3) • 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Technische Informatik** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
2 und 3	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Winter- und einer im Sommersemester	9	Präsenz (Vorlesung + Übung): **67,5** Selbststudium: **172,5** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 270**	Deutsch	Günther Schatter

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science		Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen zum Teil aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Das Modul vermittelt erforderliche Kenntnisse an der Schnittstelle von Informationstechnik/Elektrotechnik und Medieninformatik. Nach einem erfolgreichen Besuch der Lehrveranstaltungen sind die Studierenden in der Lage, informations- und elektrotechnische Denkvorstellungen auf konkrete Probleme der Medieninformatik zu projizieren. Dies umfasst Informations- und codierungstechnische Fragestellungen zur Beschreibung medialer Sachverhalte als auch elektrotechnisch geprägtes Denken zur Beschreibung von Hardware- und Systemstrukturen. Damit können die Studierenden konkrete Probleme mit den Begrifflichkeiten der Informationstheorie abstrakt darstellen und Lösungsansätze entwickeln, als auch Anforderungen an die Hardware- und Kommunikationstechnik schlüssig formulieren.

Spezielle Qualifikationsziele aus Information und Codierung:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:
  - Zeichentheorie als Grundlage von Verständigungsvorgängen,
  - Semiotisches Dreieck, Zeichentypen, Zeichen in Kalkülen, Alphabet,
  - Signale als Zeichen, Signalkennwerte, Pegel,
  - Beschreibung und Eigenschaften kontinuierlicher und diskreter Signale,
  - Quantisierungs- und Samplingtheorem, Abweichungen und Artefakte,
  - Mathematische Modelle von Reiz-Reaktionsgesetzen akustischer und visueller Signale nach Weber-Fechner, Stevens

- Mediale Anwendungen wie Farbraummodelle, Hörfläche,
- Struktur und Analyse von Mediendaten, Header und Metadaten,
- Physik der Kommunikation, Spektrum elektromagnetischer Wellen,
- Netzwerktopologien, Beschreibung durch Graphen und Matrizen,
- Mathematischer Informationsbegriff,
- N-Gramm-Analyse, Dice-Koeffizient,
- Zusammenhang von Entropie und Redundanz, Kompression
- Codierungstheorem, Decodierungsbedingung, Entropiecodierung, Bitverarbeitung
- Kanalcodierung, lineare Blockcodes, Hammingdistanz, Fehlererkennung und -korrektur,
- Hammingcode, Hammingschranke,
- Irrelevanzcodierung, Typologie medialer Dateiformate.

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:

  - entscheiden, welche Quantisierungs- und Abtastbedingungen zur Informationsgewinnung erforderlich sind,
  - Kennwerte für Analog-Digital- bzw. Digital-Analog-Umsetzer selbstständig auswählen,
  - Pegelberechnungen verstehen und durchführen,
  - über die Angemessenheit von Kommunikationstopologien entscheiden,
  - Zusammenhang zwischen Datenvolumen, Übertragungszeit und Datenrate sicher beherrschen ,
  - einfache Algorithmen zur Entropiecodierung anwenden,
  - Codiereffizienz und Decodierfähigkeit bewerten,
  - einfache Sicherungs- und Korrekturalgorithmen anwenden,
  - Bitoperationen beherrschen,
  - Headerstrukturen von Mediendateien analysieren und manipulieren,

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:

  - Netzwerktopologien als Graphen und Matrizen modellieren und bewerten, einfache Algorithmen auf Graphen,
  - N-Gramm-Analysen durchführen und auswerten,
  - Datenkompressionsverfahren mathematisch als Entropieproblem bzw. sinnesphysiologisch als Irrelevanzproblem bearbeiten,
  - Fehlererkennungs- und -korrekturprobleme analysieren und bearbeiten einschließlich Problem der Einzelbitverarbeitung lösen,

- selbstständig abstrakte Begriffe in mathematisch-formaler Darstellung verstehen sich selbständig in die Anwendung grundlegender Methoden der Informations- und Codierungstechnik einarbeiten.

- Die Studierenden können Probleme der Medieninformatik formalisieren und systematisch nach Lösungen suchen mit Hilfe von Konzepten

  - der Signaltheorie,
  - der Informationstheorie,
  - der Graphentheorie,
  - der Codierungstheorie.

- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden

  - der Signaltheorie,
  - der Informationstheorie,
  - der Graphentheorie,
  - der Codierungstheorie.

  zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

Spezielle Qualifikationsziele aus der Elektrotechnik und Systemtheorie:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:

  - Ladung, Spannung, Strom, Widerstand, Kapazität, Induktivität, Leistung, Energie,
  - Kirchhoffsche Regeln, Lösungsmethoden algebraischer Gleichungssysteme,
  - Vierpolparameter, einfache Netzwerkanalyse
  - Ideale und reale Quellen, Spannungs- und Leistungsanpassung,
  - Gewöhnliche Differenzialgleichungen, Lösungsmethoden, Übergangsfunktion,
  - Anwendungen zur Abschätzung von Datenraten auf diskreten Kanälen,
  - Kenngrößen von Wechselgrößen wie Mittelwerte und spektrale Eigenschaften berechnen,
  - Kalküle zur Beschreibung des Verhaltens linearer zeitinvarianter Systeme,
  - Komplexe Zahlen, komplexe Widerstände, spektrales Verhalten,
  - Übertragungsfunktion, Frequenzgang, Bodediagramm,
  - Dimensionierung von Filter-, und Resonanzsystemen,
  - Analogiebetrachtungen der Systemtheorie

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung

der folgenden Fragestellungen anzuwenden:

- Elementare elektrische Phänomene erkennen und systematisch einordnen,
- Dimensionierungen und Abschätzungen für passive Netzwerke durchführen,
- Fragen der Spannungs- und Leistungsanpassung zwischen Systembaugruppen bewerten,
- einfache Hardwareprobleme der Interfacegestaltung analysieren und bearbeiten,
- das energetische Verhalten von Spannungsquellen und Schaltungen hinsichtlich der Lebensdauer insbesondere mobiler Systeme bewerten,
- das statische Verhalten von einfachen Systemen erklären und bewerten wie Kennlinien, Nichtlinearität, Hysterese,
- das Zeitverhalten von einfachen Systemen erklären und abschätzen wie Verzögerung, Totzeiten, Datenraten, Speicherzeiten,
- das Frequenzverhalten von einfachen Systemen erklären und dimensionieren wie Tiefpass, Hochpass, Bandpass, Bandsperre, Schwingkreis,

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:
  - Netzwerkanalysen mit den Mitteln der linearen Algebra,
  - Anwendung der Fourier-Analysis auf spektrale Problemstellungen,
  - Einarbeitung in Fragen der elektronischen Schaltungstechnik insbesondere für Interfaceprobleme,
  - selbstständig abstrakte Begriffe in mathematisch-formaler Darstellung verstehen sich selbständig in die Anwendung grundlegender Methoden einarbeiten wie Operatorenrechnung und Analyse einfacher nichtlinearer Systeme.
- Die Studierenden können Probleme der Elektrotechnik und Systemtheorie mit Hilfe von Konzepten
  - der Elektrostatik und -dynamik,
  - der linearen Algebra,
  - der Differenzial- und Integralrechnung,
  - der komplexen Rechnung und systemtheoretischen Betrachtungsweise.

  formalisieren, formale Fragestellungen analysieren, systematisch nach möglichen Lösungen suchen und selbst entsprechende Algorithmen entwickeln.
- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden
  - der elektrischen Schaltungstechnik an der Schnittstelle zur Elektronik,
  - der Betrachtung im Zeitbereich,
  - der Analyse im Frequenzbereich,
  - der Grundlagen der Systemtheorie

zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

Lehrinhalte

- Information und Codierung
  - Zeichen, Signale, Information, Codierung,
  - Quantisierungs- und Abtasttheorem,
  - Physik der Kommunikationsmedien,
  - Netzwerktopologien, Graphen und Matrizen,
  - Mathematischer Informationsbegriff,
  - Entropie und Redundanz,
  - Diskrete Quellen und Kanäle,
  - Entropie- und Irrelevanzcodierung,
  - Kanalcodierung, Modulation.
- Elektrotechnik und Systemtheorie
  - Elektrotechnische Grundgesetze,
  - Grundlagen zur Berechnung elektrischer Schaltungen,
  - Passive Bauelemente und deren Grundschaltungen,
  - Differentialgleichungen und und Übergangsfunktionen, Zeitverhalten,
  - Kontinuierliche Systeme im Frequenzbereich, Spektralverhalten,
  - Komplexe Widerstände, Filter und Resonanzsysteme,
  - Modellierung von dynamischen Systemen.

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Zunächst sind im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Ausgewählte Lösungen von Übungsaufgaben sind darüber hinaus im WWW zu veröffentlichen, um Techniken der wissenschaftlichen Arbeit zu entwickeln. Die Lehrinhalte und Lösungen werden in der Übung diskutiert und vertieft. Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen.

Ausgewählte Lehrinhalte berühren sich mit Lehrinhalten aus anderen Modulen in sinnvoller Weise. So werden Kenntnisse der Funktionsweise von elektrischen Phänomenen in der Elektronik bzw. in Computersystemen im Modul Praktische Informatik als auch im Modul Kommunizierende Systeme behandelt. Weiterhin ergeben sich Korrespondenzen zum Modul Mathematik und Informatik-Strukturen. Durch diese systematischen Abstimmungen werden Zusammenhänge für die Studierenden deutlich.

Hinweise

Literatur zu Information und Codierung: Klimant, Herbert u.a.: Informations- und Kodierungstheorie. Shannon, Claude; Weaver, Warren: The Mathematical Theory of Communication. Werner, Martin: Nachrichtentechnik. Literatur zu Elektrotechnik und Systemtheorie: Paul, Reinhold: Elektrotechnik für Informatiker. Paul, Reinhold; Paul, Steffen: Repetitorium Elektrotechnik. Scheithauer, Rainer: Signale und Systeme.	
Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Information und Codierung (wöchentlich im Sommersemester) • Elektrotechnik (wöchentlich im Wintersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1) • 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Theoretische Informatik** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
3 und 4	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Sommer- und einer im Winter-semester	9	Präsenz (Vorlesung + Übung): **67,5** Selbststudium: **172,5** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 270**	Deutsch	Dr. PD A. Jakoby

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science		Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. Dauer: 90 Minuten In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Im Rahmen des Moduls werden die theoretischen Grundlagen der Informatik vermittelt, welche für ein Studium der Informatik bzw. Medieninformatik besonders wichtig sind. Den Studierenden wird ein Einstieg in die Begriffe und Methoden der Analyse von Problemen vermittelt. Nach einem erfolgreichen Besuch der beiden Lehrveranstaltungen können die Studierenden konkrete Probleme mit Begrifflichkeiten der Mathematik abstrakt darstellen, mit Methoden der Theoretischen Informatik analysieren und bezüglich ihrer Lösbarkeit und Komplexität charakterisieren. Sie sind somit in der Lage verschiedene grundlegende Fragen für unterschiedliche Probleme mit Hilfe von Standardverfahren algorithmisch zu lösen bzw. deren Unlösbarkeit mit den zur Verfügung stehenden Ressourcen nachvollziehbar nachzuweisen.

Spezielle Qualifikationsziele aus den Formalen Sprachen:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:
  - Formale Sprachen und deren Grammatiken
  - grundlegende Formen von Grammatiken und deren Charakterisierung bezüglich der Chomsky-Hierarchie
  - reguläre Sprachen, kontextfreie Sprachen, kontextsensitive Sprachen
  - grundlegende Automaten-Modelle und deren Zuordnung zu den unterschiedlichen Stufen der Chomsky-Hierarchie: Endliche Automaten, deterministischer Kellerautomat, Kellerautomaten, Turing Maschine (mit und ohne Platzbeschränkung)
  - reguläre Ausdrücke und deren Anwendungen
  - Verfahren für Umwandlungen zwischen Grammatiken und den jeweiligen Automaten

- Verfahren für Umwandlungen zwischen den verschiedenen Beschreibungen für reguläre Sprachen: deterministischer endlicher Automat, nicht-deterministischer endlicher Automat, Epsilon nicht-deterministischer endlicher Automat, reguläre Grammatiken, reguläre Ausdrücke, minimaler endlicher Automat
- Normalformen von Grammatiken
- Pumping-Lemmata und deren Beweis: für reguläre Sprachen und für kontextfreie Sprachen
- Anwendbarkeit der Pumping-Lemmata und deren Schranken
- Satz von Nerode und dessen Beziehung zu minimalen endlichen Automaten
- Algorithmen zur Umwandlung einer Grammatik in eine Normalform
- elementare Eigenschaften der Stufen der Chomsky-Hierarchie
- elementare Problemstellungen für Formale Sprachen
- elementare Algorithmen für Formale Sprachen: Membership, Gleichheit, Teilmenge
- die Laufzeit und Korrektheit der entsprechenden Algorithmen
- Kenntnis der Abschlusseigenschaften bezüglich Mengenoperationen der jeweiligen Stufen der Chomsky-Hierarchie
- Endlichkeit, Entscheidbarkeit, Abzählbarkeit, Über-Abzählbarkeit
- grundlegende Beweismethoden der Mathematik bzw. der Theoretischen Informatik: direkte und indirekte Beweise, vollständige Induktion, Diagonalisierung
- das Halteproblem

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:
  - die grundlegenden Beweistechniken einsetzen, um einfache Beweise zu führen
  - entscheiden, ob eine Menge (bzw. Sprache) endlich, regulär, kontextfrei, kontextsensitiv, abzählbar oder über-abzählbar ist
  - Grammatiken für grundlegende Formale Sprachen angeben und den Beweis der Korrektheit führen
  - die Nicht-Zugehörigkeit einer Sprache zu den regulären bzw. zu den kontextfreien Sprachen mit Hilfe des jeweiligen Pumping-Lemmas zeigen
  - die Zugehörigkeit bzw. die Nicht-Zugehörigkeit einer Sprache zu den regulären Sprachen mit Hilfe der Neroden Relation nachweisen
  - einen Automaten für eine Sprache aus einer gegebenen Grammatik herzuleiten (und umgekehrt)
  - Abschlusseigenschaften bezüglich Mengenoperationen der jeweiligen Stufen der Chomsky-Hierarchie herzuleiten
  - grundlegende Algorithmen (z.B. der CYK-Algorithmus) aus dem Bereich der Formalen Sprachen anzuwenden und zu simulieren
  - die Chomsky- und die Greibach-Normalform für kontextfreie Grammatiken herzuleiten
  - eine einfache Diagonalisierung durchzuführen

- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen anwenden:

    - Mittelschwere Sprachen untersuchen (z.B. Herleiten einer entsprechenden Grammatik und Beweis der Korrektheit, Herleiten eines Automaten für diese Sprache, Zeigen der Nicht-Zugehörigkeit zu einer Hierarchiestufe durch Widerspruchsannahme, Pumping-Lemma, Satz von Nerode, Anwendung von Sätzen aus der Vorlesung, …)
    - offen gestellte mathematische Fragestellungen mit Hilfe der oben genannten Beweistechniken beweisen oder, durch Angabe von Gegenbeispielen widerlegen
    - selbstständig abstrakte Begriffe in formaler Darstellung verstehen, sich selbständig in die Anwendung grundlegender Methoden einarbeiten
    - Anwendung von regulären Ausdrücken zur Suche in großen Datenmengen

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten

    - der Automatentheorie,
    - der Grammatiken von Formalen Sprachen,
    - der Algorithmik und
    - der Berechenbarkeitstheorie

    formalisieren, formale Fragestellungen analysieren, systematisch nach möglichen Lösungen suchen und selbst entsprechende Algorithmen entwickeln. In einfachen Fällen können sie die Korrektheit von Algorithmen begründen und ihr asymptotisches Laufzeitverhalten abschätzen.

- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden

    - der Automatentheorie,
    - der Grammatiken von Formalen Sprachen,
    - der Algorithmik und
    - der Berechenbarkeitstheorie

    zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

Spezielle Qualifikationsziele aus der Komplexitätstheorie:

- Die Studierenden kennen insbesondere die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:

    - Modelle für Berechnungen: primitive Rekursion, μ-Schema, Loop-Berechenbarkeit, While-Berechenbarkeit, RAM, Turing-Maschine
    - elementare Techniken zum Umgang mit Turing-Maschinen
    - die Universelle Turing-Maschine
    - Entscheidbarkeit, Nicht-Entscheidbarkeit
    - Church-Turing-These

- Many-One-Reduktion und Turing Reduktion
- Zentrale Sätze der Rekursionstheorie: S-m-n Theorem, Rekursionstheorem, Fixpunktsatz, Satz von Rice
- Determinismus und Nichtdeterminismus
- Ressourcen und Komplexitätsmasse
- Asymptotische Notationen: O, o, Ω, ω, Θ
- Rechenregeln für asymptotische Wachstumsklassen
- Rekursionsformeln und asymptotische Wachstumsklassen
- Komplexitätsklassen versus Formale Sprachen
- Platz- und Zeit-Komplexitätsklassen
- Hierarchie-Sätze für Platz und Zeit
- der Satz von Savitch
- der Satz von Immerman und Szlepscenyi
- Ressourcen beschränkte Reduktionen
- Formen des Erfüllbarkeitsproblems: CVP, SAT, 2-SAT, 3-SAT, k-SAT, Tautologie, QBF
- die Komplexitätsklassen L, NL, P, NP, co-NP, PSPACE
- Graphenprobleme: Erreichbarkeit in Graphen, hamiltonischer Pfad, TSP, Cliquen-Problem, Färbungsproblem, Vertex-Cover Problem
- Approximierbarkeit von Problemen
- Labyrinth-Problem und die Klassen L und NL
- die Komplexität von Spielen und die Klasse PSPACE am Beispiel von GO
- Puzzleprobleme

- Die Studierenden sind in der Lage, ihr Wissen unter anderem in den folgenden Bereichen bzw. zur Beantwortung der folgenden Fragestellungen anzuwenden:
  - die grundlegenden Beweistechniken einsetzen, um einfache Beweise zu führen
  - entscheiden ob eine Menge rekursiv, rekursiv aufzählbar oder nicht rekursiv aufzählbar ist
  - den Satz von Rice zur Entscheidung der Nicht-Rekursivität einer Menge anwenden
  - nichtdeterministische Verfahren zur Lösung von Problemen angeben
  - die Komplexität von Problemen einzuordnen
  - Komplexitätsklassen miteinander in Relation zu setzen
  - aus der Komplexität von Problemen eine Aussage über die effiziente Lösbarkeit der Probleme herzuleiten
- Die Studierenden können ihr Wissen auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen

Beispielen anwenden:

- Mittelschwere Sprachen bezüglich ihrer Lösbarkeit oder Komplexität untersuchen (Suche von geeigneten Problemen für eine Reduktion, eine Reduktion durchführen)
- offen gestellte Fragestellungen über die effiziente Lösbarkeit von Problemen mit Hilfe der oben genannten Beweistechniken beweisen oder, durch Angabe von Reduktionen, widerlegen
- selbstständig abstrakte Begriffe in formaler Darstellung verstehen, sich selbständig in die Anwendung grundlegender Methoden einarbeiten
- Anwendung von regulären Ausdrücken zur Suche in großen Datenmengen

- Die Studierenden können Probleme der Informatik mit Hilfe von Konzepten
  - der Rekursionstheorie,
  - der Komplexitätstheorie und
  - der Algorithmik

  formalisieren, formale Fragestellungen analysieren, systematisch nach möglichen Lösungen suchen und selbst entsprechende Reduktionen entwickeln. In einfachen Fällen können sie die Komplexität analysieren und das asymptotische Laufzeitverhalten von Lösungsverfahren abschätzen.
- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden
  - der Rekursionstheorie,
  - der Komplexitätstheorie und
  - der Algorithmik

  zur Modellbildung geeignet sind. Insbesondere können sie diese Entscheidung auch nachvollziehbar begründen.

Lehrinhalte

- Formale Sprachen
  - Beweistechniken
  - Grammatiken
  - Chomsky-Hierarchie
  - Endliche Automaten
  - Kellerautomaten
  - Turing-Maschine
  - Reguläre Sprachen
  - Kontextfreie Sprachen
  - Kontextsensitive Sprachen

- Chomsky-0 Sprachen
- Abschlusseigenschaften von Sprachklassen
- Algorithmen zur Lösung des Membership-Problems (CYK-Algorithmus)
- Algorithmen zur Überführung zwischen Automaten und Grammatiken (bzw. regulären Ausdrücken)
- Nerode Klassen
- Pumping-Lemma für reguläre Sprachen
- Pumping-Lemma für kontextfreie Sprachen
- Normalformen für kontextfreie Sprachen
- Halteproblem
- Diagonalisierung
- Entscheidbarkeit, Abzählbarkeit und Über-Abzählbarkeit

- Komplexitätstheorie
  - Beweistechniken
  - Loop-Berechenbar
  - While-Berechenbar
  - Turing-Maschine
  - RAM
  - Primitive Rekursion
  - Partielle Rekursion
  - μ-Schema
  - elementare Techniken zum Umgang mit Turing-Maschinen
  - Universelle Turing-Maschine
  - Entscheidbarkeit
  - Nicht-Entscheidbarkeit
  - Church-Turing-These
  - Many-One-Reduktion
  - Turing Reduktion
  - Rekursionstheorie
  - S-m-n Theorem
  - Rekursionstheorem

- Fixpunktsatz
- Satz von Rice
- Determinismus und Nichtdeterminismus
- Komplexitätsmasse
- Asymptotischen Notationen: O, o, Ω, ω, Θ
- Platz-Komplexitätsklassen
- Zeit-Komplexitätsklassen
- Hierarchie-Sätze
- Satz von Savitch
- Satz von Immerman und Szlepscenyi
- Formen des Erfüllbarkeitsproblems: CVP, SAT, 2-SAT, 3-SAT, k-SAT, Tautologie, QBF
- Komplexitätsklassen: L, NL, P, NP, co-NP, PSPACE
- Erreichbarkeit in Graphen
- hamiltonischer Pfad
- TSP
- Cliquen-Problem
- Färbungsproblem
- Vertex-Cover Problem
- Approximierbarkeit
- Labyrinth-Problem
- Komplexität von Spielen
- Puzzleprobleme

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Sie werden im Rahmen einer Übung vertieft. Dabei sind zunächst im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Nach der Korrektur der Übungsaufgaben durch die Betreuenden werden die Lösungen in einer gemeinsamen Sitzung besprochen (nun wieder Präsenzstudium). Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen.

Bestimmte Lehrinhalte überlappen sich mit Lehrinhalten aus anderen Modulen. Zum Beispiel werden Beweismethoden auch in Mathematik I und Informatik Strukturen behandelt, und im Modul Praktische Informatik werden Grundkenntnisse der Analyse des asymptotischen Laufzeitverhaltens und der Effizienz vermittelt. Dieses Vorgehen soll eine Orientierung in der Stofffülle vermitteln und Zusammenhänge erkennbar machen. Es basiert auf einem didaktischen Konzept, das sich an das Spiralprinzip aus der Schuldidaktik anlehnt.

Die jeweiligen Veranstaltungen bestehen aus jeweils einer wöchentlichen 90-minütigen Vorlesungen sowie einer Einzelstunde (45 min) Übung.

Hinweise

Handouts und Literatur zu Formalen Sprachen:

- Jakoby: Folien zur Vorlesung Formalen Sprachen
- Engeler, Peter Läuchli: Berechnungstheorie für Informatiker (Leitfäden und Monographien der Informatik); Vieweg+Teubner Verlag; 1992
- Hopcroft, Motwani, Ullman: Einführung in Automatentheorie, Formale Sprachen und Berechenbarkeit, Pearson Studium, 2011
- Tourlakis: Theory of Computation; Wiley, 2014

Handouts und Literatur zu Komplexitätstheorie:

- Jakoby: Folien zur Vorlesung Komplexitätstheorie
- Engeler, Peter Läuchli: Berechnungstheorie für Informatiker (Leitfäden und Monographien der Informatik); Vieweg+Teubner Verlag; 1992
- Reischuk: Komplexitätstheorie Band I: Grundlagen: Maschinenmodelle, Zeit- Und Platzkomplexität, Nichtdeterminismus; Teubner Verlag; 1999
- Wegener: Komplexitätstheorie: Grenzen der Effizienz von Algorithmen; Springer; 2003
- Homer, Selman: Computability and Complexity Theory (Texts in Computer Science); Springer; 2011
- Sipser: Introduction to the Theory of Computation; Cengage Learning, Inc.; 2012

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Formale Sprachen (wöchentlich im Wintersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1)
• Komplexitätstheorie (wöchentlich im Sommersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Mensch-Maschine-Interaktion** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
2 und 3	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Winter- und einer im Sommer-semester	9	Präsenz (Vorlesung + Übung): **67,5** Selbststudium: **172,5** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 270**	Deutsch	Eva Hornecker

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science		Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. Dauer: je 2 Stunden. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Im Rahmen des Moduls werden

- Die Studierenden kennen die unter ‚Lehrinhalte' genannten Begriffe und Methoden und können sie erklären (z.B. in Bezug auf relevante Theorien und Methoden).
- Sie verstehen und können begründen:
  - was Usability ausmacht und warum gute Usability wichtig für den erfolgreichen Einsatz von Softwaresystemen ist,
  - welche Eigenschaften menschlicher Wahrnehmung und Kognition Einfluss auf gute Usability besitzen,
  - warum ein Verstehen des Nutzungskontextes sowie der Benutzerbedürfnisse und –eigenschaften zentral für die Entwicklung aufgabenangemessener Systeme ist,
  - den Stellenwert von Prototyping, Nutzertests und sowie von Theorien und Modellen über Nutzer und deren Verhalten,
  - warum neue Technologien zur Entwicklung neuer Interaktionsmethoden führen und neue Usabilityprobleme erzeugen können,
- Die Studierenden sind in der Lage, dieses Wissen auf einfache Beispielszenarien anzuwenden.

- Sie können Interfaces bezüglich wichtiger Konsequenzen für Prozesse menschlicher Wahrnehmung und Kognition analysieren und bewerten sowie gängige Probleme erkennen.
- Sie können Beispielszenarien und Systeme bezüglich ihrer Usability analysieren und bewerten.
- Sie können unter Berücksichtigung des Wissens über Usabilityprinzipien sowie Grundlagen menschlicher Wahrnehmung und Kognition einfache Benutzungsschnittstellen entwerfen.

- Sie können ihr Wissen auch auf anspruchsvollere Probleme bzw. komplexere Beispielszenarien übertragen und anwenden
  - Sie können in Anwendungsfällen (Beispielszenarien und Systeme) entscheiden, ob diese Usabilityprinzipien bzw. Prinzipien des User-Centered Designs einhalten oder verletzen, und ihre Bewertung nachvollziehbar begründen
  - Sie können für ausgewählte Klassen von Usabilityproblemen Lösungsansätze generieren. Dies umfasst auch die wichtigsten Klassen von Usabilityproblemen, die in Eigenschaften menschlicher Wahrnehmung und Kognition fußen.
  - Sie können für vorgegebene Szenarien Vorgehensvorschläge für einen benutzerzentrierten Designprozess entwickeln und ihre Methodenwahl begründen.

- Die Studierenden können die gelehrten Methoden und ihr Wissen im Kontext einfacher Beispielszenarien praktisch anwenden
  - z.B. Papierprototypen entwickeln, einen einfachen Usability Test durchführen, eine KLM-Analyse durchführen, Systemanforderungen zu bestimmen, ...
  - z.B. ein Interface bezüglich auftretender (Farb-)Kontraste, Gruppierung von Bedienelementen/Anzeige oder Anforderungen an Aufmerksamkeit und Gedächtnissysteme analysieren und in Hinblick auf eine Eignung für generelle wie spezielle Nutzergruppen bewerten.

- Zusätzlich werden soziale Fähigkeiten und Schlüsselqualifikationen durch Gruppenarbeit basierend auf konkreten Problemen und Aufgaben geübt

Lehrinhalte

Grundlagen von Wahrnehmung und Kognition für Usability und HCI (WaKog)

- Aufmerksamkeits-, Gedächtnis- und Lernmodelle (sensorisches Gedächtnis, Arbeitsgedächtnis nach Baddeley/Logie, Langzeitgedächtnis), Lern- und Vergessensfunktionen, prozedurales und deklaratives Wissen, Theorie der dualen Kodierung, Erinnerung als Konstruktionsprozess, Prototypen, Kategorien, Schemata, Skripte, dual tasks, cueing, Richtlinien für das Design multimodaler Systeme (z.B. auf Basis der Theorie von Mayer & Moreno).
- Eigenschaften, neuronale/psychologische Grundlagen und Grenzen menschlicher Wahrnehmung, insbesondere visueller. Wahrnehmung von Farbe, Helligkeit, Kontrast, Kontur, Bewegung, Objekten, Bildvorder-/Hintergrund; photopisches und skotopisches Sehen; visuelle Suche; Fechnersches Gesetz, Signalentdeckungstheorie; Photometrie und Radiometrie, Vergleich menschlicher Wahrnehmung (z.B. in der Kontrast- und Objekterkennung) mit technischen Verfahren. Design von Interfaces für Tag-/Nachtgebrauch, für Normal- und Farbfehlsichtigkeiten. Blickbewegungen und deren Messen. Gestaltregeln im Sehen und Hören. Eigenschaften von Wahrnehmungskanälen im Vergleich.
- Grundlagen menschlichen Problemlösens und Schlussfolgerns, Expertiseerwerb, Heuristiken im menschlichen Entscheiden und deren Konsequenzen für Interfacedesign.
- Interindividuelle Unterschiede menschlicher Perzeption und Kognition, insbesondere von Fähigkeiten
- Grundlagen des Experimentaldesigns, Vorbereitung auf Usability testing.

HCI (Benutzungsoberflächen)

- Prozess des User-Centered Designs (Requirements Analyse, Prototyping, Evaluation)
- Grundprinzipien der Software-Ergonomie und Usability, Designregeln und -prinzipien, wie z.B. Affordances, Constraints, Mapping, etc.
- Grundlegende Methoden der Benutzerforschung (Interviews, Fokusgruppen, Beobachtung, Logfile-Analyse, Usability-Tests, Feldstudien, Experiment) sowie einfache Design- und Dokumentationsmethoden (Personas, Storyboards, User Profile, Paper Prototyping, horizontale und vertikale Prototypen),
- Grundlegende Interaktionstypen und –stile sowie Eingabe- und Ausgabetechnologien, die Rolle von Interaktionsmetaphern
- Deskriptive und vorhersagende Modellierungsmethoden der HCI (Hierarchical Task Analysis, Fitts Law, Keystroke-Level Model etc.)

Lehr- und Lernmethoden / Didaktisches Konzept

Vorlesungen und praktische Übungen kombiniert mit individueller und Gruppenarbeit zu theoretischen und praktischen Aspekten der Inhalte.

In praktischen Übungen werden die Inhalte vertieft. Über regelmäßig zu bearbeitende Übungsaufgaben wird eine kontinuierliche Aufarbeitung der Inhalte eingefordert. Die Übungen enthalten Elemente projektorientierter Gruppenarbeit zu konkreten Beispielproblemen und –szenarien (problembasiertes Lernen). Die vermittelten Methoden werden exemplarisch eingesetzt und angewendet sowie das vermittelte Wissen praktisch umgesetzt.

Nach der Korrektur der eingereichten Übungsaufgaben durch die Betreuenden werden diese mit Anmerkungen an die Studierenden zurückgegeben und beispielhafte Lösungen sowie typische Fehler werden in der Präsenzphase besprochen.

Hausaufgaben umfassen ein Maximum an 12 Aufgabenzetteln (je 6 für WaKog und HCI) verteilt über das Modul. Wissenschaftliche Mitarbeiter und studentische Tutoren aus höheren Semestern betreuen die Studierenden in den Übungen und stehen zusammen mit den Lehrenden für Rückfragen und Diskussion zur Verfügung

Die einzelnen Veranstaltungen bestehen aus wöchentlichen 90-minütigen Vorlesungen sowie einer Einzelstunde (45 min) Übung, welche typischerweise zweiwöchentlich stattfinden 90-minütigen Übungen. Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen.

Das Modul vermittelt praktisch-methodisches und theoretisches Wissen auf Bachelorniveau, das in der Klausur geprüft wird.

Hinweise

Literatur
J. Wolfe, K. Kluender, D. Levi. Sensation and Perception, 3rd ed., Sinauer Associates, 2012.
E.B. Goldstein. Sensation and Perception, 8th ed., Wadsworth, 2010.
J. Andersen. Cognitive Psychology and its Implications, 7th edition, Worth Publishers, 2009.
J. Preece, H. Sharp, Y. Rogers. Interaction Design: Beyond Human-Computer Interaction, 4th Edition. Wiley 2015
D. Norman. The Design of Everyday Things. Basic Books
A. Dix, J. Finlay, G. Abowd, R. Beale. Human-Computer Interaction. Prentice Hall 2014

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Wahrnehmung und Kognition (wöchentlich im Sommersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1)
• Human-Computer Interaction (wöchentlich im Wintersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Informationssysteme** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
3 und 4	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Sommer- und einer im Winter-semester	9	Präsenz (Vorlesung + Übung): **78,75** Selbststudium: **161,25** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 270**	Deutsch	Benno Stein

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science	Erfolgreiche Absolvierung der Vorlesungen • Einführung in die Informatik • Diskrete Strukturen	Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. Dauer: 1.5h In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts (Basis: ECTS) der Noten in den enthaltenen Kursen.

Qualifikationsziele

Im Rahmen des Moduls werden Grundlagen moderner Informationssysteme mit den Schwerpunkten Datenbanken und Web-basierte Systeme behandelt. Die Vorlesungen dieses Moduls sind grundlagen- und methodenorientiert ausgerichtet.

Spezielle Qualifikationsziele aus dem Bereich Datenbanken:

- Grundbegriffe von Datenbanken kennen und einordnen können
- Kenntnis von und sicherer Umgang mit Techniken zur Modellierung von Datenbankanwendungen
- Beherrschung der Umsetzung externer Schemata in relationale Schemata
- Beherrschung der Logik-basierten Grundlagen von Anfragesprachen
- Erfahrung und verbesserter Umgang mit formalen Methoden
- Praktische Erfahrung bei der Datenanfrage und Datenmanipulation auf der Basis von SQL
- Beherrschung der theoretischen Grundlagen von Datenbanksystemen
- Verständnis für die Grenzen von Datenbanksystemen
- Beherrschung von Vorgehensweisen, um sich selbst weiterbilden können

Spezielle Qualifikationsziele aus den Grundlagen der Web-Technologie:

- Grundbegriffe des Webs und der Internettechnologie einordnen können
- Verständnis für den technischen Aufbau des Internets
- Erfahrung in der Zuordnung von Server- und Client-Technologien
- Beherrschung wichtiger Auszeichnungssprachen, um Web-Inhalte zu repräsentieren
- Praktische Erfahrung im Umgang ausgewählter Scriptsprachen zur Verarbeitung von Web-Inhalten

- Beherrschung der grundlegenden Verarbeitungsabläufe zur Verwaltung und Präsentation von Web-Inhalten
- Förderung der Fähigkeit, um zwischen Web-Grundlagen und Web-Werkzeugen zu unterscheiden
- Förderung der Fähigkeit, die Tragfähigkeit neuer Entwicklungen einzuschätzen
- Entwicklung von Prinzipien, um die Informationsmenge die durch die Web-Technologievielfalt entsteht, effektiv zu filtern
- Beherrschung von Vorgehensweisen, um sich selbst weiterbilden können

Lehrinhalte

Lehrinhalte aus dem Bereich Datenbanken:
Die Vorlesung gibt eine Einführung in die Konzepte moderner Datenbanksysteme und stellt den Datenbankentwurf für klassische Datenmodelle, insbesondere für das Relationenmodell vor. Hierzu gehören folgende Vorlesungseinheiten:

- Grundlegendes zum Datenbankentwurf und Datenbankmodelle
- Konzeptueller Datenbankentwurf mit einem syntaktisch reichem Modell
- Logischer Datenbankentwurf mit dem relationalen Modell
- Grundlagen relationaler Anfragesprachen: Relationale Algebra
- Grundlagen relationaler Anfragesprachen: Tupel- und Domänenkalkül
- SQL: Datenanfrage, Datendefinition, Datenmanipulation
- SQL: Anwendungsprogrammierung
- Relationale Entwurfstheorie: Funktionale Abhängigkeiten und Normalformen
- Relationale Entwurfstheorie: Entwurfsalgorithmen

Lehrinhalte aus dem Bereich Web-Technologie:
Web-basierte Systeme werden hinsichtlich ihrer Architektur und Funkti-onsweise eingeführt, wobei ein Schwerpunkt auf der Vorstellung Web-basierter Sprachen liegt. Grundlagen aus benachbarten Gebieten wie der Rechnerkommunikation werden behandelt, soweit dies für das Verständ-nis erforderlich ist. Hierzu gehören folgende Vorlesungseinheiten:

- Überblick über die historische Entwicklung des Internets
- Begriffe, Problemstellungen und Herausforderungen im Web-Kontext
- Internetprotokollaufbau und das HTTP-Protokoll
- Dokumentsprachen: HTML, CSS
- Dokumentsprachen: XML-Grundlagen
- Dokumentsprachen: XML-Schema
- Dokumentsprachen: XSL
- XML-Programmierung und APIs
- Ausgewählte Server-Technologien
- Ausgewählte Client-Technologien
- Einführung in Web-Middleware

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Es werden statische und dynamische Beamer-Präsentationen mit dem Einsatz der Tafel kombiniert. Der Beamer-Einsatz ermöglicht u.a. die schnelle und übersichtliche Präsentation und Kombination von Inhalten in verschiedenen Kontexten. Beispiele (Datenbanken): gleichzeitige Darstellung von ER-Diagrammen und ihrer relationalen Entsprechung, Animation von Relationenzerlegungen, Demonstration vom SQL-Aufrufen an Datenbanken. Beispiele (Web-Technologie): Darstellung von Protokollabläufen, Interaktive Manipulation von Programmen und Web-Sprachen im Web-Client.
Die Tafel wird eingesetzt, um Beweise und formale Definitionen schrittweise zu entwickeln und die Herausforderungen und Schwierigkeiten bei der Entwicklung zu diskutieren. Beispiel (Datenbanken): Schrittweise Herleitung eines Tupel-Kalkülausdrucks auf Basis einer relational spezifizierten Anfrage.
Weiterhin wird die Tafel dafür eingesetzt, um auf Rückfragen von Studierenden spontan zu reagieren und entsprechende Grundlagen oder Beispiele ad-hoc herzuleiten.
Alle Lehrinhalte werden im Rahmen von Übungen vertieft. Hierzu zählt sowohl die Anwendung der Theorie im Rahmen praktischer Modellierungs- und Programmieraufgaben als auch das Wiederholen und Nachvollziehen der theoretischen Grundlagen anhand von kleineren Beweisen.
Insbesondere werden in den Übungen zur Web-Technologie die verschiedenen Techniken und Konzepte zur Modellierung und Programmierung von Web-Inhalten sowohl isoliert (pro Übung) als auch im Zusammenspiel (über mehrere Übungen hinweg) behandelt.
Pro Übungsserie gibt es Pflichtaufgaben und freiwillige Aufgaben. Die Bearbeitung der Pflichtaufgaben geschieht in Gruppen von 2-3 Studierenden. Diese Gruppen bleiben typischerweise in ihrer Zusammensetzung im Verlauf einer Vorlesung unverändert. Die Pflichtaufgaben werden durch die Betreuenden korrigiert und die Lösungen in einer gemeinsamen Sitzung besprochen. Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen.
Zur Unterstützung des Selbststudiums kann ein Tutorium z bestimmten Inhalten angeboten werden.

Hinweise

Empfohlene Literatur aus dem Bereich Datenbanken:

- Ramez Elmasri, Shamkant B. Navathe. Fundamentals of Database Systems. 6th edition, Addison Wesley, 2010.
- Alfons Kemper, Andre Eickler. Datenbanksysteme - Eine Einführung. 8. Auflage, Oldenbourg, 2011.
- Andreas Heuer, Gunter Saake. Datenbanken: Konzepte und Sprachen. 5. Auflage, mitp, 2013.
- Gottfried Vossen. Datenmodelle, Datenbanksprachen und Datenbankmanagement-Systeme.5. Auflage, Oldenbourg, 2008.

Empfohlene Literatur aus dem Bereich Web-Technologie:

- Comer. Computer Networks and Internets with Internet Applications. 5. Auflage, Pearson Prentice Hall, 2008.
- Meinel/Sack. WWW - Kommunikation, Internetworking, Web-Technologien. Springer, 2013.
- Meinel/Sack. Internetworking: Technische Grundlagen und Anwendungen.. Springer, 2012.
- Harold/Means. XML in a Nutshell. 3. Auflage, OReilly, 2004.

Umfangreiche und aktuelle Literaturangaben sind im Script verlinkt.

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Datenbanken (wöchentlich im Wintersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1)
• Grundlagen der Web-Technologie (wöchentlich im Sommersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü2)

Titel **Kommunizierende Systeme** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
3 und 4	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Sommer- und einer im Wintersemester	9	Präsenz (Vorlesung + Übung): **67,5** Selbststudium: **172,5** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 270**	Deutsch und Englisch	Volker Rodehorst

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science	• Mathematik I, • Informatik Strukturen • Praktische Informatik	Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen Lehrveranstaltungen in separaten Klausuren geprüft. Je Lehrveranstaltung eine Klausur. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Klausuren. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Im Rahmen des Moduls werden die wesentlichen Aspekte von verteilten Systemen behandelt. Die Veranstaltung Parallele und verteilte Systeme behandelt die grundlegenden Konzepte der parallelen und verteilten Verarbeitung von Daten. Im praktischen Kontext werden auch Mehrkern-Systeme, die verteilte Berechnung auf Rechenclustern und der Einsatz von Grafikprozessoren für wissenschaftliches Rechnen behandelt. Im Rahmen der Veranstaltung Moderne Kryptographie geht es dagegen um verteilte Systeme, die versuchen, im Beisein von Gegenspielern bzw. Angreifern miteinander zu kommunizieren. Es geht um grundlegende Sicherheitsmerkmale wie Vertraulichkeit, Authentizität und Nicht-Abstreitbarkeit, aber auch um theoretische und praktische Angriffe auf kommunizierende Systeme.

Spezielle Qualifikationsziele aus den Parallelen und verteilten Systemen:

- Die Studierenden sind in der Lage, ihr theoretisches und praktisches Wissen zur Analyse und effektiven Beschleunigung von aufwändigen Algorithmen einzusetzen.
- Sie kennen folgende Grundbegriffe und können die wesentlichen Zusammenhänge erläutern:
  - Möglichkeiten und Grenzen der Parallelprogrammierung (kontrollparallel, datenparallel)
  - Wettrennen um Ressourcen (Synchronisation, Barriere, Mutex, Semaphor)
  - Analyse und Beseitigen von Datenabhängigkeiten
  - Effiziente Parallelisierung (Granularität, Lastenausgleich)
  - Mehrkern- bzw. Multiprozessor-Berechnung von Threads mit gemeinsamem Speicher
  - Verteiltes Rechnen durch expliziten Nachrichtenaustausch von Prozessen (Cluster, Grid, Cloud)
  - Interprozesskommunikation ((nicht-)blockierend, (a)synchron, gepuffert, kollektiv)
  - Speichermodelle (Cache, global, lokal, privat)
  - Ausführungsmodelle (Online-Kompilierung, Topologie, Befehlswarteschlange)
  - Mathematische Darstellung zur Modellierung und Verifikation
- Durch die begleitenden Übungen mit den Entwicklungsumgebungen OpenMP, Open MPI und OpenCL sind die Studierenden in der Lage ihr theoretisches Wissen auch in der Praxis auf anspruchsvolle Probleme und komplexe Beispiele zu übertragen.

- Sie können für eigene Anwendungen abwägen, welche Verfahren zur Beschleunigung am besten geeignet sind und die Entscheidung nachvollziehbar begründen.

- Die Studierenden können Fragestellungen von verteilten Systemen mit Hilfe
  - von Petrinetzen
  - der (linearen) temporalen Logik

  mathematisch formalisieren, Spezialfälle untersuchen und geeignete Lösungen finden.

Spezielle Qualifikationsziele aus der Modernen Kryptographie:

- Die Studierenden kennen die folgenden Begriffe und Zusammenhänge und können sie anderen erklären:
  - klassische Kryptosysteme (Caesar, Substitution) und perfekte Kryptosysteme (Vernam)
  - Entropie und Passwort-Sicherheit
  - Theorie der Abstrakten Strom- und Blockchiffren
  - Praktische Blockchiffren (DES, AES)
  - Theorie der Public-Key Kryptographie
  - Praktische Public-Key Kryptosysteme (RSA, Rabin, Diffie-Hellman, ElGamal)
  - Digitale Unterschriften
  - Angriffe auf fehlerhaft implementierte Public-Key Kryptosystem, Digitale Unterschriften und auf Textbook-RSA, sowie Gegenmaßnahmen
  - Definition der Vertraulichkeit (Real-or-Random) und der Authentizität
  - Privacy Modes of Operation für Blockchiffren
  - Blockchiffren-basierte Message Authentication Codes und Authenticated Encryption Modes
  - Beweisbar sichere Public-Key Kryptographie (RSA-OAEP und Full-Domain-Hash Unterschriften)

- Die Studierenden sind in der Lage, ihr Wissen zur Beantwortung von Sicherheitsfragen anzuwenden.

- Sie können ihr Wissen auch auf anspruchsvolle Probleme bzw. im Zusammenhang mit komplexen Beispielen übertragen. Sie können vorgegebene unsichere Systeme angreifen. Ebenso können Sie zu gegebenen sicheren Systemen begründen, warum diese, nach dem Stand der Technik, als sicher anzusehen sind.

- Die Studierenden können Probleme der Kryptographie bzw. IT-Sicherheit mit Hilfe von Konzepten
  - der Kryptanalyse,
  - der symmetrischen Kryptographie und
  - der asymmetrischen Kryptographie

  formalisieren, formale Fragestellungen analysieren und systematisch nach möglichen Lösungen suchen.

- Die Studierenden können in Anwendungsfällen entscheiden und bewerten, welche Konzepte und Methoden der Kryptographie geeignet sind, ein Sicherheitsproblem zu lösen, und welche nicht. Insbesondere können sie eine derartige Entscheidung auch nachvollziehbar begründen.

Lehrinhalte

- Parallele und verteilte Systeme
  - Grundlagen und Architekturen
  - Mehrkern-Programmierung mit OpenMP
  - Programmierung verteilter Systeme mit Open MPI
  - Massive Parallelisierung auf Grafikkarten mit OpenCL
  - Modellierung und Verifikation mit Petrinetzen und temporaler Logik

- Moderne Kryptographie
  - Grundlagen
  - Passwörter und Entropie
  - Stromchiffren
  - Blockchiffren
  - Public-Key Kryptographie
  - Unsichere Systeme
  - Vertraulichkeit
  - Authentizität

- Authentisierte Verschlüsselung
- Public-Key Sicherheit

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Sie werden im Rahmen einer Übung vertieft. Dabei sind zunächst im Selbststudium vorgegebene Übungsaufgaben zu bearbeiten. Nach der Korrektur der Übungsaufgaben durch die Betreuenden werden die Lösungen in einer gemeinsamen Sitzung besprochen (nun wieder Präsenzstudium). Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden. Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen.

Hinweise

Literatur zu den Parallelen und Verteilten Systemen:

- G. Bengel, C. Baun, M. Kunze und K.U. Stucky: Masterkurs Parallele und Verteilte Systeme: Grundlagen und Programmierung von Multicore-Prozessoren, Multiprozessoren, Cluster, Grid und Coud, 2. Aufl., Springer, 503 S., 2015.
- S. Hoffmann und R. Lienhart: OpenMP - Eine Einführung in die parallele Programmierung mit C/C++, Informatik im Fokus, Springer, 172 S., 2009.
- A.S. Tanenbaum und M. van Steen: Verteilte Systeme: Prinzipien und Paradigmen, 2. Aufl., Pearson Studium - IT, 768 S., 2007.
- T. Rauber und G. Rünger: Parallele Programmierung, 2. Aufl., Springer, 485 S., 2007.

Literatur zur Modernen Kryptographie:

- J. Buchmann: Einführung in die Kryptographie, Springer Verlag.
- A. Beutelspacher: Kryptologie, Vieweg Verlag.
- Beutelspacher, Schwenk, Wolfenstetter: Moderne Verfahren der Kryptographie, Vieweg Verlag.
- D. R. Stinson: Cryptography Theory and Practice, CRC Press.
- C. Eckert: IT-Sicherheit. Konzepte - Verfahren - Protokolle, Oldenbourg.
- A. J. Menezes, P. C. van Oorschot, S. A. Vanstone: Handbook of Applied Cryptography, CRC Press.

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Parallele und Verteilte Systeme (wöchentlich im Wintersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1)
• Moderne Kryptographie (wöchentlich im Sommersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Graphische Informationssysteme** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
5 und 6	jährlich	Wöchentlich im mit einer Lehrveranstaltung im Sommer- und zwei Veranstalltungen im Wintersemester	13.5	Präsenz (Vorlesung + Übung): **101.25** Selbststudium: **258.75** Prüfungsvorbereitung einschließlich der Prüfungen: **45** **Summe 405**	Deutsch und Englisch	Charles Wüthrich

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science	• Mathematik I, II • Software • HCI	Um eine möglichst zeitnahe Prüfung der Studierenden zu gewährleisten werden die einzelnen zum Teil aufeinander aufbauenden Lehrveranstaltungen in separaten Klausuren geprüft. Für die Lehrveranstaltungen Computergrafik und Computer Vision eine Klausur. Für die Lehrveranstaltung Visualisierung wird aus didaktischen Gründen eine mündliche Prüfung angeboten. Die regelmäßige Bearbeitung der Übungsaufgaben ist eine Voraussetzung für die Zulassung zu den Prüfungen. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Im Rahmen des Moduls werden grundlegende mathematische und algorithmische Methoden aus drei Bereichen des Visual Computing vermittelt. Computer Vision fokussiert auf das Erkennen und Interpretieren von Informationen, die i.a. in Bildern realer Umgebungen vorliegen. Die Computergrafik konzentriert sich auf Verfahren, die aus 3D-Modellen, manuell modelliert oder auch durch Verfahren der Computer Vision erzeugt, realitätsnahe oder auch abstrakte Bilder oder Bildfolgen generieren. Die Visualisierung beschäftigt sich mit Verfahren zur Darstellung und geeigneten Interaktionsformen für gemessene, simulierte oder gesammelte Daten.

Spezielle Qualifikationsziele aus der Computergrafik

Das Ziel der Computergrafik besteht darin, mit Hilfe von Computern visuelle Darstellungen zu erzeugen. Die Studierenden kennen nach dieser Vorlesung die grundlegenden Verfahren und Komponenten, die den Stand der Technik darstellen. Diese beinhalten Hardwarekomponenten, Farbräume sowie grundlegende Rasterungsverfahren bis hin zu Verfahren zur Elimination verdeckter Flächen. Weiterhin verfügen sie über einen Überblick zu Modellierungsverfahren, Ansichtstransformationen, lokalen und globalen Beleuchtungsverfahren sowie grundlegende Ansätzen der computergestützten Animation. Durch die Durchführung eines studienbegleitenden Belegs sind die Studierenden in der Lage, den Stoff aus der Vorlesung in konkrete Programme umzusetzen. Sie kennen und beherrschen die wesentlichen Konstrukte aus OpenGL und Shading Languages. Studierende sind am Ende der Veranstaltung in der Lage, geeignete grafische Algorithmen für ein Problem auszuwählen, diese zu programmieren und komplexe grafische Anwendungen zu entwickeln.

Spezielle Qualifikationsziele aus dem Bereich Computer Vision:

Die Studierenden kennen die wesentlichen Grundlagen der algebraischen projektiven Geometrie und können dieses Wissen für die räumliche 3D-Rekonstruktion von Objekten aus Bildern einsetzen. Sie sind in der Lage die bei der Bildaufnahme herrschende Abbildungsgeometrie zu modellieren und für eine Oberflächenrekonstruktion zu invertierten. Die Studierenden können perspektivische Abbildungen mit Hilfe der direkten linearen Transformation bestimmen und beherrschen auch die optischen Abbildungseigenschaften von Linsen. Nach der Veranstaltung sind sie in der Lage grundlegende Rekonstruktionsverfahren zu beurteilen und auszuwählen. Die begleitenden Übungen versetzen sie in die Lage eigene Verfahren zu implementieren, um wichtige Kameraeigenschaften zu kalibrieren, die einzelnen Sensorpositionen und –ausrichtungen aus den Bildern zu bestimmen und räumliche Objektpunkte zu triangulieren.

Spezielle Qualifikationsziele aus der Visualisierung

- Die Studierenden sind in der Lage auf verschiedenen Ebenen zu abstrahieren:
  - Problemabstraktion: Übersetzung des Problems aus einer spezifischen Anwendungsdomäne in das Visualisierungsvokabular
  - Datenabstraktion: Was soll visualisiert werden?
  - Aufgabenabstraktion: Warum wollen Nutzer die Daten visualisiert haben?
- Sie kennen eine große Auswahl an grundlegenden Visualisierungstechniken und deren Aufbau
  - Visuelle Kodierung: Wie werden die Daten dargestellt
  - Interaktionstechniken: Wie kann die Sicht auf die Daten angepasst werden und wie können verschiedene Sichten kombiniert werden
- Die Studierenden sind in der Lage, Daten aus eine Anwendungsdomäne zu analysieren und abstrahieren sowie systematisch geeignete Visualisierungstechniken auszuwählen und mit passenden Interaktionstechniken zu kombinieren.
- Sie können ihr Wissen auch auf anspruchsvolle und komplexere Probleme übertragen und dabei die Skalierbarkeit der ausgewählten Techniken beurteilen.
- Die Übungen versetzen Studierende in die Lage, grundlegende Visualisierungstechniken selbst zu entwickeln, implementieren und testen.

Lehrinhalte

- Computergrafik
  - Grundlagen
  - Licht, Farbe Farbräume
  - 3D Modellierung
    - Polygone, Splines, Patches, Rotation, Translation, Szenengraphen
  - Lokale Beleuchtung
    - Reflexion, ambiente Beleuchtung, diffuse Beleuchtung, spekulare Beleuchtung
  - Texturen und Schatten
  - Grafikpipeline, Vertexshader, Fragmentshader
  - Rasterisierung
  - Clipping
  - Hidden Surface Removal
  - Globale Beleuchtung: Raytracing
  - Radiosity
  - Aliasing und Antialiasing
  - Vision,Light und Displays
  - Computer Animation

- Computer Vision
  - Projektive Geometrie
    - Perspektivische Abbildungen mit homogenen Koordinaten
    - Bestimmung linearer Transformationen
    - Robuste Parameterschätzung
  - Kameramodellierung
    - Kalibrierung, Sensororientierung und Triangulation
    - Optische Abbildung mit Linsen
    - Bündelausgleich für Mehrbild-Geometrien
  - Stereobildverarbeitung
    - Erstellung von Stereo-Normalbildern
    - Globale Strategien für die dichte Bildzuordnung

- Visualisierung
  - Informationsvisualisierung
    - Munzners what-why-how Analyse-Framework
    - Grundlegende Interaktionstechniken
    - Visualisierungstechniken für tabellarische Daten, zeitabhängige Daten, Bäume, Graphen, kartographische Daten und Text
  - Wissenschaftliche Visualisierung
    - Datentypen und mathematische Grundlagen
    - Isolinen und Isoflächen
    - Direkte Volumendarstellung mittels Ray Casting
    - Multi-Resolution Verfahren für Volumendaten

- Strömungsvisualisierung

Lehr- und Lernmethoden / Didaktisches Konzept

Die Lehrinhalte werden in Vorlesungen präsentiert und im Rahmen von Übungen vertieft (Präsenzstudium). Übungsaufgaben sind im Selbststudium zu bearbeiten. Nach der Korrektur und Bewertung der Übungsaufgaben durch die Betreuenden werden ausgewählte Lösungen in der folgenden Übung besprochen. Die regelmäßige Bearbeitung der Übungsaufgaben und die aktive Teilnahme an den Übungen dient hierbei unteranderem der Selbstkontrolle der Studierenden und ist zum für die kontinuierliche Vertiefung der Lehrinhalte sehr zu empfehlen.

Hinweise

Literatur zu Computergrafik:

- F. Cohen, J. Wallace: Radiosity and Realistic Image Synthesis, Academic Press, 1993, ISBN 978-0121782702
- A. Glassner: Principles of Digital Image Synthesis, Morgan Kaufman, 1995, ISBN 978-1558602762
- A. Watt: 3D Computer Graphics, Addison-Wesley, 1999
- Akenine-Möller, Haines, Hoffman: Real-Time Rendering, A K Peters, 2008
- Foley, Van Dam, Feiner, Hughes: Computer Graphics. Principles and practice, Addison Wesley, 2009, ISBN 978-0201357172
- Shirley, Marschner: Fundamentals of Computer Graphics, A K Peters, 2009

Literatur zu Computer Vision:

- R. Hartley und A. Zisserman: Multiple View Geometry in Computer Vision, 2. Aufl., Cambridge University Press, 655 S., 2003
- O. Faugeras und Q.-T. Luong: The Geometry from Multiple Images, MIT Press, 646 S., 2004
- Y. Ma, S. Soatto, J. Kosecka und S. Sastry: An Invitation to 3D-Vision, 2. Aufl., Springer, 526 S., 2005
- R. Szeliski: Computer Vision - Algorithms and Applications, Springer, 812 S., 2010

Literatur zu Visualisierung:

- R. Spence: Information Visualization: An Introduction (3rd Edition)
- T. Munzner: Visualization Analysis and Design

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Computergrafik (wöchentlich im Wintersemester) • Computer Vision (wöchentlich im Wintersemester) • Visualisierung (wöchentlich im Sommersemester)	• 4.5 ECTS-Punkte (SWS: V2+Ü1) • 4.5 ECTS-Punkte (SWS: V2+Ü1) • 4.5 ECTS-Punkte (SWS: V2+Ü1)

Titel **Medien** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
1 und 2	jährlich	Wöchentlich im mit jeweils einer Lehrveranstaltung im Sommer- und einer im Wintersemester	6	Präsenz (Vorlesung): **45** Selbststudium: **105** Prüfungsvorbereitung einschließlich der Prüfungen: **30** **Summe 180**	Deutsch	

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul für Medieninformatik, Bachelor of Science		Je Lehrveranstaltung eine Klausur. In Sonderfällen kann aus didaktischen Gründen stattdessen auch eine mündliche Prüfung angeboten werden. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den enthaltenen Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Im Rahmen des Moduls werden betriebswirtschaftliche Grundkompetenzen vermittelt. Empfohlen wird der Besuch einer Veranstaltung über Medienwirtschaft sowie der Veranstaltung Medienrecht für Informatiker.

Spezielle Qualifikationsziele aus der Medienwirtschaft:

- Entwicklung betriebswirtschaftlicher Sichtweisen
- Kennenlernen und Einüben wichtiger betriebswirtschaftlicher Konzepte

Spezielle Qualifikationsziele aus dem Medienrecht für Informatiker:

- Die Studierenden lernen, sich in den für Informatiker wichtigsten Rechtsgebieten des Medienbereichs mit ihren rechtlichen Grundlagen zu orientieren.
- Sie entwickeln ein Grundverständnis für zentrale Begriffe des Rechts und kennen die wichtigsten Vertragsformen rechtsverbindlicher Dokumente.

Lehrinhalte

Medienwirtschaft:

- Einführung
- Unternehmensführung
  - Grundlagen
  - Planung und Entscheidung
  - Organisation
  - Personalwirtschaft
  - Controlling
- Konstitutive Entscheidungen
- Investitionsplanung und Investitionsrechnung
- Finanzierung

Medienrecht:

- Grundlagen
  - Einführung in die Rechtsordnung

- Das Grundgesetz: Funktion und Aufbau
- Grundrechte mit besonderer Bedeutung für die Medien
- Das Zivilrecht und seine Bedeutung für die Medien

- Persönlichkeitsrecht
  - Das allgemeine Persönlichkeitsrecht
  - Fallgruppen
  - Die identifizierende Berichterstattung
  - Vermögensrechtliche Bestandteile des allgemeinen Persönlichkeitsrechts
  - Zivilrechtliche Ansprüche
- Urheberrecht
  - Einführung
  - Entstehung des Urheberrechtsschutzes
  - Einzelne Werkarten
  - Sonstige Werke
  - Der Urheber
  - Dauer des Urheberrechts
  - Die Rechte des Urhebers
  - Urhebervertragsrecht (Nutzungsrechte)
  - Nichtübertragbarkeit des Urheberrechts
  - Schranken des Urheberrechts
  - Verwandte Schutzrechte (Leistungsschutzrechte)
  - Schutz des Filmherstellers (§§ 88 - 94 UrhG)
  - Durchsetzung von Ansprüchen
- Internetrecht
  - Rechtliche Entwicklung
  - Begriff des Telemediendienstes
  - Zulassungs- und Anmeldefreiheit
  - Informationspflichten
  - Die Verantwortlichkeit der Anbieter
  - Recht der Domains

Lehr- und Lernmethoden / Didaktisches Konzept
Die Lehrinhalte werden in einer Vorlesung präsentiert (Präsenzstudium). Zur Unterstützung des Selbststudiums kann ein Tutorium angeboten werden.

Hinweise

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)
• Medienwirtschaft (wöchentlich im Winterrsemester) • Medienrecht für Inforamtiker (wöchentlich im Sommersemester)	• 3.0 ECTS-Punkte (SWS: V2) • 3.0 ECTS-Punkte (SWS: V2)

Titel **Wahlmodul** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
5 und 6	Jedes Semester		15	**450**	Deutsch, ggf. andere Sprache	Prüfungsausschuss

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Wahlmodul für Medieninformatik, Bachelor of Science	Zulassung zum Studium Abhängig von der gewählten Veranstaltung können weitere Zulassungsvoraussetzungen gelten.	Abhängig von der gewählten Veranstaltung. Die Endnote des Moduls wird berechnet anhand des gewichteten Mittelwerts der Noten in den gewählten Kursen, gewichtet anhand ECTS.

Qualifikationsziele

Das Wahlmodul ermöglicht den Studierenden wahlweise

1. den Besuch von Lehrveranstaltungen der Medieninformatik, um die Kompetenzen als Medieninformatiker zu vertiefen,
2. den Besuch von Lehrveranstaltungen, die von Professoren anderer Fakultäten bzw. von Professoren anderer Bereiche der Fakultät Medien angeboten werden, um die Kompetenzen zu verbreitern
3. und den Besuch von Englisch-Veranstaltungen, um die Berufsqualifikation zu verbessern bzw. um sich auf den Besuch eines englischsprachigen Master-Studiums vorzubereiten.

Im Zusammenhang mit einem Studienaufenthalt im Ausland können auch Veranstaltungen zum Erwerb der jeweiligen Landessprache im Rahmen des Wahlmoduls belegt werden.

Lehrinhalte

(abhängig von den gewählten Veranstaltungen)

Lehr- und Lernmethoden / Didaktisches Konzept

(abhängig von den gewählten Veranstaltungen)

Hinweise

Grundsätzlich können Studierende frei aus den folgenden Veranstaltungen wählen:

1. Bachelorveranstaltungen des Bereichs Medieninformatik, soweit sie nicht an anderer Stelle angerechnet werden.

   Handelt es sich bei der Veranstaltung um ein Projekt, haben Studierende, die dieses Projekt im Rahmen ihres Erst- bzw. Zweitprojekts belegen wollen, Vorrang vor Studierenden, die sich das Projekt im Wahlmodul anrechnen lassen wollen. Insbesondere gibt es keinen Anspruch darauf, das im Rahmen des Wahlmoduls ein Projekt belegen zu können.

   Der Prüfungsausschuss kann auf Antrag auch die Anrechnung einzelner Master-Veranstaltungen aus dem Bereich der Medieninformatik im Wahlmodul erlauben.
2. Fachfremde Veranstaltungen, die von Professoren oder Dozenten anderer Fakultäten bzw. anderer Bereiche der Fakultät Medien angeboten werden.

Als fachfremd gilt eine Veranstaltung, deren Inhalte bzw. Lernziele nicht Informatik-typisch sind (im Gegensatz, z.B., zum Programmieren, zum Schreiben von Skripten in einer ausführbaren Sprache oder zu der Beschäftigung mit den Details binärer Kommunikationsprotokolle). Es obliegt dem Prüfungsausschuss, zu entscheiden, ob eine Veranstaltung fachfremd ist, oder nicht. Bei Veranstaltungen, die nicht fachfremd sind, kann der Prüfungsausschuss die Anrechnung der Leistungspunkte begrenzen oder ganz versagen.

3. Sprachkurse in Englisch, die an der Bauhaus-Universität angeboten werden.

Über die Anrechnung anderer Veranstaltungen im Wahlmodul entscheidet der Prüfungsausschuss. Unter anderem sollen fachfremde Veranstaltungen, die formal nicht von Professoren oder Dozenten angeboten werden, sondern von Universitätsmitarbeitern, trotzdem angerechnet werden können, wenn sich ein Professor oder Dozent als zuständig für die fachliche Qualität der Veranstaltung und eine angemessene Bewertung der Studierenden erklärt.

Im Zusammenhang mit einem Studienaufenthalt an einer ausländischen Hochschule können im Wahlmodul auch

- Veranstaltungen (auch fachfremde) der gastgebenden Universität und
- Veranstaltungen zum Erwerb der jeweiligen Landessprache

angerechnet werden.

Die Anrechnung von Sprachkursen im Rahmen des Wahlmoduls (Englisch und, ggf., die Landessprache für einen Studienaufenthalt) ist auf insgesamt maximal 6 Leistungspunkte beschränkt.

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)

Titel	**Erstprojekt / Zweitprojekt**	Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
4 bzw. 5	jedes Semester	regelmäßige Termine über das gesamte Semester hinweg	15	Präsenz: **50** (davon Treffen mit Betreuenden **45**, öffentliche Präsentation des Projektes: **5)** Selbststudium: **400** **Summe 450**	Deutsch, ggf. Englisch	Jeweilige Professur

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Zwei Pflichtmodule für Medieninformatik, Bachelor of Science	Abhängig vom Projektthema Kenntnisse aus einzelnen Pflichtmodulen des 1.-3. Semesters (Erstprojekt) bzw. des 1.-4. Semesters (Zweitprojekt)	• Regelmäßige Zwischenpräsentationen (mündlich) • ggf. Zwischenberichte (schriftlich) • hochschul- oder allgemein öffentliche Präsentation des Projektes (mündlich) • Dokumentation der Ergebnisse, wissenschaftlicher Report oder Software-Quelltext (schriftlich)

Qualifikationsziele

Abhängig vom Projektthema lernen die Studierenden, wie man bei anspruchsvollen Fragestellungen nach geeigneten Lösungsansätzen recherchiert, deren Vor- und Nachteile versteht, und die Ergebnisse derartiger Recherchen dokumentiert, oder die Studierenden machen praktische Erfahrungen mit dem Entwurf, der Implementation und der Evaluation von Software- und ggf. Hardwaresystemen und Benutzerinterfaces, oder sie lernen, wie man wissenschafliche Experimente und insbesondere Benutzerstudien plant, ausführt und auswertet.

Aufbauend auf den bereits im Rahmen der für das Projektthema relevanten Pflichtmodule erworbenen fachlichen Kompetenzen können die Studierenden zeigen, dass sie in der Lage sind, Sachverhalte zu verstehen, Methoden der Medieninformatik anzuwenden und auf verwandte Probleme zu übertragen. In der Projektgruppe sollen sie Probleme analysieren und Lösungen bewerten. Außerdem wird der Projektgruppe die Gelegenheit geboten, gemeindam mit den Betreuern/innen an der Schöpfung neuer wissenschaftlicher Erkenntnisse bzw. an der Erstellung neuer praktischer Lösungen mitzuwirken.

Neben der Vermittlung fachlicher Kompetenzen, ensprechend dem jeweiligen Projektthema, dienen die Projekte vor allem der Vermittlung von

- Sozialen Kompetenzen und
- Selbstkompetenz

Unabhängig vom Projektthema lernen die Studierenden, wie man als Team zusammenarbeitet, auch autonom, ohne Beisein von Betreuern, wie man Gruppensitzungen protokolliert, wie man Ergebnisse mündlich vorträgt und diese später schriftlich fixiert. Die Studierenden verstehen die Bedeutung von Projektmanagement und Organisation für komplexe Projekte.

Eine Konsequenz des “learning by doing” Ansatzes des projektbezogenen Studiums ist, dass Studierende ihre Arbeits- und Kommunikationsweise, ihre Selbstdisziplin und ihre Erwartungen an andere Projektteilnehmer im Laufe des Projektes hinterfragen und ihre eigenen Fehler nachträglich erkennen können. Das Zweitprojekt soll dazu dienen, Lehren aus dem Erstprojekt zu ziehen und die dort bereits erworbenen Kompetenzen entsprechend zu vertiefen.

Lehrinhalte

Die konkreten Lehrinhalte sind abhängig vom Projektthema, beinhalten jedoch stets mehrere der folgenden Punkte:

- Literaturrecherche zu einem Thema aus der Wissenschaft oder der Praxis der Medieninformatik
- Spezifikation eines Systems der Medieninformatik
- Implementation eines Systems in Soft- oder Hardware.
- Evaluation eines Systems
- Planen Durchführen und Auswerten von Benutzerstudien
- Schreiben eines wissenschaftlichen Berichts

Lehr- und Lernmethoden / Didaktisches Konzept
Typischerweise treffen sich die Projektgruppen einmal die Woche mit den Betreuenden, um Zwischenergebnisse zu präsentieren und Feedback für die weitere Arbeit zu erhalten. In manchen Fällen finden zu Projektbeginn, in Rahmen einer Einarbeitungsphase, mehrere Treffen pro Woche statt, dafür können die Treffen im weiteren Verlauf des Semesters entsprechend seltener sein. Immer liegt der Schwerpunkt der Arbeit auf dem Selbststudium. Das projektbezogene Studium ist ein zentraler didaktischer Bestandteil der Lehre an der Bauhaus-Universität. Unter Anleitung der zuständigen Professuren werden die Studierenden mit komplexen wissenschaftlichen oder praktischen Problemen der Medieninformatik konfrontiert, die sie so selbstständig wie möglich lösen sollen. Projektteilnehmer tauschen sich sehr regelmäßig untereinander aus, treffen sich regelmäßig mit den Betreuenden, diskutieren ihre Zwischenergebnisse innerhalb der Projektgruppe und mit den Betreuern, präsentieren ihr Projekt wenigstens einmal hochschulöffentlich oder allgemein öffentlich, und verfassen einen Abschlussbericht.
Hinweise
Um im 4. Semester ein Mobilitätsfenster zu schaffen, kann das Erstprojekt durch beliebige Module/Veranstaltungen aus der Informatik oder Medieninformatik ersetzt werden, wenn diese im Rahmen eines Auslandsstudiums an einer ausländischen Hochschule erworben wurden. Es wird davon ausgegangen, dass die Aufnahme eines Auslandsstudiums dazu beiträgt, unter anderem • Sozialen Kompetenzen und • Selbstkompetenz zu erwerben, die dann, im Zweitprojekt, an der Bauhaus-Universität weiter vertieft werden sollen.

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)

Titel **Bachelor-Modul** Modulnummer

Semester (optional)	Häufigkeit des Angebots	Dauer + Turnus	ECTS-Punkte	Studentischer Arbeitsaufwand (in Stunden)	Sprache	Verantwortlich
6	jedes Semester	4 Monate, jederzeit	15	Verfassen der Arbeit: **360**, davon • Präsenz (Treffen mit Betreuer(in)): **45** • Selbststudium: **315** Verteidigung der Arbeit: **90**, davon • Präsenz (Treffen mit Betreuer(in) und Halten des eigentlichen Vortrages): **5** • Selbststudium: **85** Summe: **450**	Deutsch, auf Antrag Englisch	Betreuerin / Betreuer

Modultyp / Verwendbarkeit	Voraussetzungen für die Teilnahme	Prüfungsleistung(en)
Pflichtmodul Medieninformatik Bachelor of Science	Zum Zeitpunkt der Anmeldung der Arbeit müssen die Studierenden in den anderen Modulen mindestens 120 Leistungspunkte nachgewiesen haben. Die Verteidigung ist öffentlich und die letzte Prüfungsleistung. Deshalb müssen die Studierenden zum Zeitpunkt der Verteidigung alle anderen Module erfolgreich abgeschlossen haben.	• Wissenschaftlicher Report (schriftlich, 80%) • Verteidigung der Arbeit (mündlich, 20%)

Qualifikationsziele

Eine Bachelor-Arbeit ist die Bearbeitung eines wissenschaftlichen Themas mit

- schriftlicher Ausarbeitung
- und anschließender mündlicher Präsentation.

Die Aufgabe einer Bachelorarbeit kann die Entwicklung von Software oder Hardware, die Durchführung einer empirischen Studie, eine formale Beweisführung oder eine Literaturrecherche umfassen.

Die schriftliche Ausarbeitung soll die Struktur einer wissenschaftlichen Veröffentlichung haben und muss den Grundsätzen guter wissenschaftlicher Praxis genügen, insbesondere bei der Verwendung von Referenzen und Zitaten sowie bei der Nutzung möglicherweise erhobener oder genutzter empirischer Daten. Es muss klar erkennbar sein, welche eigenen Beiträge der Student geleistet hat und welche er von anderen Quellen übernommen hat.

Im Rahmen der Bachelor-Arbeit soll der Studierende zeigen, dass er

- dass er grundlegende wissenschaftliche Methoden beherrscht,
- dass er die Grundsätze guter wissenschaftlicher Praxis kennt und ihnen entsprechend handelt
- und dass er innerhalb einer vorgegebenen Frist ein Thema der Informatik bzw. Medieninformatik mit wissenschaftlichen Methoden erarbeiten und für Fachkollegen verständlich darstellen und präsentieren kann.

Lehrinhalte

- Literaturrecherche zu einem Thema aus der Wissenschaft oder der Praxis der Medieninformatik
- Experimente, Tests, Implementationsarbeit oder Benutzerstudien
- Schriftliches Fixieren der Ergebnisse in Form einer wissenschaftlichen Arbeit

Lehr- und Lernmethoden / Didaktisches Konzept

Weitgehend Selbststudium, Feedback durch regelmäßige Betreuung.

Hinweise

Das Bachelor-Modul ist ein zentraler Bestandteil des Studiums, und das Verfassen der Arbeit ist der wichtigste Teil dieses Moduls. Die Gewichtung der Noten reflektiert diese Bedeutung.

Zum Modul gehörende Lehrveranstaltungen / Kurse (optional)	SWS / ECTS (optional)