

# Informatik Technischer Systeme

Version vom 21.01.2021 für die Prüfungs-und Studienordnung vom 10.01.2019

Beschlossen im Fakultätsrat Technik und Informatik am 17.12.2020

## Pflichtfächer

1. Semester
  - [Diskrete Mathematik](#)
  - [Grundlagen der Technischen Informatik](#)
  - [Programmiermethodik I](#)
  - [Programmietechnik](#)
  - [Mess- und Sensortechnik](#)
2. Semester
  - [Automatentheorie und Formale Sprachen](#)
  - [Programmiermethodik II](#)
  - [Datenbanken](#)
  - [Grundlagen der systemnahen Programmierung](#)
  - [Analysis und linear Algebra](#)
3. Semester
  - [Signalverarbeitung und Stochastik](#)
  - [Algorithmen und Datenstrukturen](#)
  - [Software Engineering I](#)
  - [Betriebssysteme](#)
  - [Intelligente Sensorsysteme](#)
4. Semester
  - [Mustererkennung und Maschine Learning](#)
  - [Embedded System Engineering](#)
  - [Rechnernetze](#)
5. Semester
  - [Verteilte Systeme](#)
  - [Betriebswirtschaft](#)
  - [Projekt](#)
  - [Seminar](#)
6. Semester
  - [Cyber-physische Systeme](#)
  - [Bachelorarbeit](#)

## Wahlpflichtfächer

- [Wahlpflichtfach I, II und III](#)
- [Gesellschaftswissenschaften](#)

## Prüfungsformen

Entsprechend § 14 APSO-INGI, jeweils in der geltenden Fassung, werden die Prüfungsformen für das anschließende Modulhandbuch wie folgt definiert:

### 1. Fallstudie (FS)

Die Fallstudie ist eine schriftliche Arbeit mit begründeter Lösung. In einer Fallstudie werden einzeln oder in Gruppen durch die Anwendung wissenschaftlicher Methoden und Erkenntnisse Praxisprobleme erfasst, analysiert und gelöst. Die Bearbeitung erfolgt veranstaltungsbegleitend. Die Bearbeitungszeit endet spätestens mit dem Ablauf der Lehrveranstaltung in dem jeweiligen Semester. Die Bearbeitungsdauer kann in den studiengangsspezifischen Prüfungs- und Studienordnungen näher geregelt werden.

### 2. Hausarbeit (H)

Eine Hausarbeit ist eine nicht unter Aufsicht anzufertigende schriftliche Ausarbeitung, durch die die oder der Studierende die selbstständige Bearbeitung eines gestellten Themas nachweist. Die Bearbeitungszeit einer Hausarbeit beläuft sich auf bis zu drei Monate. Handelt es sich bei der Hausarbeit um eine Prüfungsleistung, dann kann in der studiengangsspezifischen Prüfungs- und Studienordnung bestimmt werden, ob nach Abgabe der schriftlichen Ausarbeitung innerhalb einer Frist von in der Regel einem Monat ein Kolloquium zu halten ist. Die Dauer des Kolloquiums beträgt mindestens 15, höchstens 45 Minuten.

### 3. Klausur (K)

Eine Klausur ist eine unter Aufsicht anzufertigende schriftliche Arbeit, in der die Studierenden ohne Hilfsmittel oder unter Benutzung der zugelassenen Hilfsmittel die gestellten Aufgaben allein und selbstständig bearbeiten. Die Dauer einer Klausur beträgt mindestens 60, höchstens 240 Minuten.

### 4. Kolloquium (KO)

Ist bei einzelnen Prüfungsarten, der Bachelor- oder Masterarbeit ein Kolloquium vorgesehen, so handelt es sich dabei um ein Prüfungsgespräch, in dem die Studierenden in freier Rede darlegen müssen, dass sie den Prüfungsstoff beherrschen. Das Kolloquium ist ein Prüfungsgespräch von mindestens 15 und höchstens 45 Minuten Dauer, welches auch dazu dient, festzustellen, ob es sich bei der zu erbringenden Leistung um eine selbstständig erbrachte Leistung handelt. Kolloquien können als Einzelprüfung oder als Gruppenprüfung durchgeführt werden. Bei Gruppenprüfungen ist die Gruppengröße bei der Festlegung der Prüfungsdauer angemessen zu berücksichtigen.

### 5. Konstruktionsarbeit (KN)

Eine Konstruktionsarbeit ist eine schriftliche Arbeit, durch die anhand fachpraktischer Aufgaben die konstruktiven Fähigkeiten unter Beweis zu stellen sind. Die Bearbeitungszeit beträgt höchstens drei Monate.

### 6. Laborabschluss (LA)

Ein Laborabschluss ist erfolgreich erbracht, wenn die Studierenden die von der Prüferin oder dem Prüfer festgelegten experimentellen Arbeiten innerhalb des Semesters erfolgreich durchgeführt haben und ihre Kenntnisse durch versuchsbegleitende Kolloquien und/oder anhand von Protokollen und/oder durch schriftliche Aufgabenlösungen nachgewiesen haben. Die Dauer des Kolloquiums beträgt mindestens 15, höchstens 45 Minuten. Die schriftlichen Ausarbeitungen sind innerhalb einer von der Prüferin bzw. dem Prüfer festgesetzten Frist abzugeben. Diese Frist endet spätestens mit Ablauf des jeweiligen Semesters, in dem die zugeordnete Lehrveranstaltungsart durchgeführt wird.

### 7. Laborprüfung (LR)

Eine Laborprüfung besteht aus einem Laborabschluss und am Ende der Lehrveranstaltung aus einer abschließenden Überprüfung der Leistung. Bei dieser Überprüfung sollen die Studierenden eine experimentelle Aufgabe allein und selbständig lösen. Die Dauer der Überprüfung beträgt mindestens 60, höchstens 240 Minuten.

### 8. Mündliche Prüfung (M)

Eine mündliche Prüfung ist ein Prüfungsgespräch, in dem die Studierenden darlegen müssen, dass sie den Prüfungsstoff beherrschen. Sie dauert in der Regel mindestens 15 und höchstens 45 Minuten. Mündliche Prüfungen können als Einzelprüfung oder als Gruppenprüfung durchgeführt werden. Eine mündliche Prüfung ist von einer oder einem Prüfenden und Beisitzenden nach § 13 Absatz 4 abzunehmen. Die mündliche Prüfung kann anstatt von einer Prüferin oder einem Prüfer auch von mindestens zwei Prüfenden abgenommen werden (Kollegialprüfung); dabei ist die oder der Studierende in den einzelnen Prüfungsfächern verantwortlich jeweils nur von einer Prüferin oder einem Prüfer zu prüfen. Die in der mündlichen Prüfung erbrachte Leistung wird sowohl bei einer Prüfung durch mehrere Prüfer, als auch bei einer Prüfung durch eine Prüferin oder einen Prüfer und eine Beisitzerin oder einen Beisitzer nur von der oder dem Prüfenden bewertet und benotet. Die verantwortliche Prüferin oder der verantwortliche Prüfer hört die anderen Prüferinnen oder Prüfer bzw. die Beisitzerin oder Beisitzer vor der Festsetzung der Note an. Die wesentlichen Gegenstände und Ergebnisse der mündlichen Prüfung sind in einem Protokoll festzuhalten. Es wird von den Prüfenden und der oder dem Beisitzenden unterzeichnet und bleibt bei den Prüfungsakten.

### 9. Projekt (Pj)

Ein Projekt ist eine zu bearbeitende fachübergreifende Aufgabe aus dem jeweiligen Berufsfeld des Studiengangs. Die Ergebnisse des Projektes sind zu dokumentieren. Die Bearbeitungszeit beträgt zwischen 6 bis 26 Wochen und wird mit einem Kolloquium abgeschlossen. In der jeweiligen studiengangsspezifischen Prüfungs- und Studienordnung können zusätzliche Bedingungen zu Form, Inhalt und Ziel des Projektes und eine andere Form des Abschlusses als durch ein Kolloquium festgelegt werden.

### 10. Referat (R)

Ein Referat ist ein Vortrag über 15 bis 45 Minuten Dauer anhand einer selbst gefertigten schriftlichen Ausarbeitung. An das Referat schließt sich unter Führung einer Diskussionsleitung ein Gespräch an. Das Referat soll in freien Formulierungen gehalten werden. Die bei dem Vortrag vorgestellten Präsentationen bzw. Grafiken sind dem Prüfer in schriftlicher oder elektronischer Form zu übergeben. In der zusätzlichen schriftlichen Ausarbeitung, die dem Prüfer zu übergeben ist, sind die wichtigsten Ergebnisse zusammenzufassen.

### 11. Test (T)

Der Test ist eine schriftliche Arbeit, in dem die Studierenden nachweisen, dass sie Aufgaben zu einem klar umgrenzten Thema unter Klausurbedingungen bearbeiten können. Die Dauer eines Tests beträgt mindestens 15, höchstens 90 Minuten. In

studiengangsspezifischen Prüfungs- und Studienordnungen kann bestimmt werden, dass die Einzelergebnisse der Tests mit in die Bewertung der Klausuren einbezogen werden.

#### 12. Übungstestat (ÜT)

Ein Übungstestat ist erfolgreich abgeschlossen, wenn die Studierenden die von der Prüferin oder dem Prüfer festgelegten theoretischen Aufgaben durch schriftliche Aufgabenlösungen erfolgreich erbracht sowie ihre Kenntnisse durch Kolloquien oder Referate nachgewiesen haben. Die Dauer des Kolloquiums beträgt mindestens 15, höchstens 45 Minuten. Die schriftlichen Ausarbeitungen sind innerhalb einer von der Prüferin bzw. dem Prüfer festgesetzten Frist abzugeben. Diese Frist endet spätestens mit Ablauf des jeweiligen Semesters, in dem die zugeordnete Lehrveranstaltungsart (Übung) durchgeführt wird.

#### 13. Übungsabschluss (ÜA)

Für die erfolgreiche Ablegung eines Übungsabschlusses ist die kontinuierliche aktive Teilnahme der Studierenden erforderlich. Es kann die schriftliche Ausarbeitung oder eine sonstige Vorstellung einzelner Übungsaufgaben vorgesehen werden.

Modulbezeichnung	<b>Diskrete Mathematik</b>	Kürzel	DM / DMÜ
Lehrveranstaltung(en)	SeU: Diskrete Mathematik Übung: Diskrete Mathematik	Fach-semester	1
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Stephan Pareigis	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Inhalte des Mathematik-Vorkurses	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• können wichtige mathematische Strukturen sicher verwenden</li> <li>• beherrschen die logischen und algebraischen Grundlagen der theoretischen Informatik</li> <li>• können Definitionsprinzipien und Beweistechniken in unterschiedlichen Bereichen und an typischen Beispielen anwenden</li> <li>• können die Methoden der linearen Algebra anwenden</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Mathematische Grundlagen: Mengen, Relationen, Abbildungen, Funktionen und deren Operatoren Boolesche Algebra, Aussagenlogik</li> <li>• Mathematische Techniken: Grundlegende Beweisstrategien, Vollständige Induktion</li> <li>• Mathematische Strukturen: Lösung von linearen Gleichungssystemen, Vektoren, Matrizen, Determinanten</li> <li>• Vertiefung in eines oder mehrere der folgenden Themen: <ul style="list-style-type: none"> <li>- Konvergenz und Grenzwerte von Folgen und Reihen</li> <li>- Schaltalgebra</li> <li>- Graphentheorie</li> </ul> </li> <li>• Kombinatorik, Diskrete Stochastik</li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Rechnerpräsentation, Vorrechnung von Beispielaufgaben, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium</p> <p>Übung: selbstständiges Lösen von Übungsaufgaben</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur</p> <p>Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben</p> <p>Voraussetzung (PVL): Übungstest (erfolgreiche Bearbeitung von Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>• P. Hartmann: Mathematik für Informatiker</li> </ul>		

Modulbezeichnung	<b>Grundlagen der Technischen Informatik</b>	Kürzel	GT / GTP
Lehrveranstaltung(en)	SeU: Grundlagen der Technischen Informatik Übung: Grundlagen der Technischen Informatik	Fachsemester	1
Arbeitsaufwand	36 Std. SeU, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Franz Korf	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	keine	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	Die Studierenden <ul style="list-style-type: none"> <li>• können grundlegenden Begriffe der Informatik benennen</li> <li>• können grundlegende Rechnerarchitekturkonzepte wie zum Beispiel die von-Neumann-Architektur beschreiben</li> <li>• verstehen die Grundkonzepte der Instruction-Set-Architektur eines Prozessors</li> <li>• können einfache Assembler-Programme für einen ausgewählten Prozessors erstellen</li> </ul>		
Inhalte	Elementare Begriffe und Definitionen der Informatik wie zum Beispiel: <ul style="list-style-type: none"> <li>• Informatik, Information, Daten, Algorithmus, Programm, Theoretische Informatik, Praktische Informatik, Technische Informatik, Hochsprachen,</li> <li>• Assembler, Syntax und Semantik von Programmiersprachen</li> <li>• Darstellung von Daten im Computer</li> <li>• Rechnerarchitekturgrundlagen auf Instruction Set Architecture Ebene</li> <li>• Elemente eines Rechners, von Neumann, Harvard</li> <li>• Grundlegende Konzepte der Instruction Set Architecture einer ausgewählten Mikroprozessorfamilie</li> <li>• Abbildung von Daten- und Kontrollstrukturen prozeduraler Hochsprachen in maschinennahe Implementierungen</li> </ul>		
Lehr- und Lernformen	SeU: Seminaristischer Unterricht, Tafel, Präsentation, freiwillige Übungsaufgaben Übung: Programmieren in 2-er Gruppen		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben Voraussetzung (PVL): Übungstestat (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"> <li>• A. S. Tanenbaum, James Goodman: Computerarchitektur</li> <li>• J. L. Hennessy, D. A. Patterson: Computer Architecture</li> <li>• W. Hohl: ARM Assembly Language: Fundamentals and Techniques</li> <li>• Bernd Becker, Rolf Drechsler, Paul Molitor (Autor): Technische Informatik - Eine Einführung. Pearson Studium (IT)</li> </ul>		

Modulbezeichnung	<b>Programmiermethodik I</b>	Kürzel	PM1
Lehrveranstaltung(en)	SeU: Programmiermethodik I	Fachsemester	1
Arbeitsaufwand	48 Std. SeU, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Michael Schäfers	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	keine	SWS	4
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• sind mit einem Programmierparadigma so gut vertraut, dass sie darin abstrakt beschriebene Programmierprobleme effizient mit korrektem ausführbarem Quelltext lösen können,</li> <li>• kennen die wesentlichen Eigenschaften der Basistypen (Zahlen, Zeichen, Wahrheitswerte, Zeichenketten) einer Programmiersprache und können diese zielgerichtet einsetzen,</li> <li>• können die Kernabstraktionen (Methoden/Prozeduren/Funktionen, Klassen) einer Programmiersprache geeignet parametrisieren,</li> <li>• kennen den Unterschied zwischen Wertsemantik und Referenzsemantik und können Referenzsemantik aktiv in eigenen Programmtexten anwenden,</li> <li>• kennen geeignete Darstellungen für Sammlungen von Werten/Objekten in einer Programmiersprache und können diese problemangemessen einsetzen,</li> <li>• und können eine abstrakte Problembeschreibung in einen programmierbaren Algorithmus übertragen.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Abstraktionsmechanismen <ul style="list-style-type: none"> <li>- Funktionale Abstraktion</li> <li>- Datenabstraktion (ADT)</li> <li>- Kontrollabstraktion (z.B. Iteratoren, Streams)</li> </ul> </li> <li>• Objektorientierung <ul style="list-style-type: none"> <li>- Polymorphie: Overloading, Overriding, Dynamische Bindung</li> </ul> </li> <li>• Ausgewählte Elemente objektorientierter Bibliotheken, z.B.: <ul style="list-style-type: none"> <li>- Collections</li> <li>- Streams, Channels</li> </ul> </li> <li>• Typisierungskonzepte <ul style="list-style-type: none"> <li>- Dynamische vs. statische Typisierung</li> </ul> </li> </ul>		
Lehr- und Lernformen	SeU: Tafelarbeit, Präsentation		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur</p> <p>Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben</p>		
Literatur	<ul style="list-style-type: none"> <li>• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben</li> </ul>		

Modulbezeichnung	<b>Programmiertechnik</b>	Kürzel	PT / PTP
Lehrveranstaltung(en)	SeU: Programmierertechnik Praktikum: Programmierertechnik	Fachsemester	1
Arbeitsaufwand	24 Std. SeU, 24 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Michael Schäfers	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	keine	SWS	2+2
Verwendbarkeit	für den Studiengang „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Zu erwerbende Kompetenzen/ Lernergebnisse	Die Studierenden <ul style="list-style-type: none"> <li>• beherrschen handwerklichen Programmierfertigkeiten, elementaren Programmierertechniken sowie eine modernen Entwicklungsumgebung mit den zugehörigen Werkzeugen (z.B. Editor, Debugger, Testautomation, Version Control System,... )</li> <li>• können eine technische Realisierung von Systemen im Kleinen durchführen</li> </ul>		
Inhalte	Elementare Programmierertechniken und Syntax einer modernen Programmiersprache: <ul style="list-style-type: none"> <li>• primitive Datentypen</li> <li>• Unicode</li> <li>• Arrays</li> <li>• Referenztypen</li> <li>• Sequenz</li> <li>• Selektion</li> <li>• Iteration</li> </ul>		
Lehr- und Lernformen	SeU: Tafelarbeit, Präsentation, Beispielaufgaben, freiwillige Übungsaufgaben, Tutorium Praktikum: Bearbeitung der Aufgaben in 2-er Gruppen, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	<ul style="list-style-type: none"> <li>• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben</li> </ul>		

Modulbezeichnung	<b>Mess- und Sensortechnik</b>	Kürzel	MS/ MSP
Lehrveranstaltung(en)	SeU: Mess-und Sensortechnik Übung: Mess-und Sensortechnik	Fach-semester	1
Arbeitsaufwand	36 Std. SeU, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Andreas Meisel	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Schulmathematik und -physik	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• können Messschaltungen mit resistiven Sensoren dimensionieren, aufbauen und auf elektrischer Ebene mit einem Computer verbinden. Dabei setzen sie grundlegende Methoden des Gleichstrom-kreises ein, wie die Knoten- und Maschenregel, das ohmsche Gesetz, Strom- und Spannungsteiler sowie das Ersatzquellenverfahren. Darüber hinaus können Sie Grundkonzepte der elektr. Messtechnik einsetzen, wie die Erweiterung des Messbereichs und die Ausschlagbrücke. Mit einem Multimeter können Sie die Schaltung kontrollieren (U-/I-/R) und die Ergebnisse interpretieren.</li> <li>• können Messschaltungen mit kapazitiven und induktiven Sensoren dimensionieren, aufbauen und auf elektrischer Ebene mit einem Computer verbinden. Dabei setzen Sie die komplexe Wechselstromrechnung ein. Mit Multimeter und Oszilloskop können Sie die Schaltung kontrollieren (<math>U\sim/I\sim</math>) und die Ergebnisse interpretieren.</li> <li>• können einfache Programme zur Messsignalaufnahme und -auswertung entwickeln, z.B. eine getaktete Messwertaufnahme (Timer, Polling), die Bestimmung von Maximal- oder Minimalwerten, die Überwachung von Schwellwerten oder eine grafische Ausgabe.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Atomistische Grundlagen der Elektrotechnik, Strom, Spannung, Widerstand, ideale Quellen, Knoten- und Maschenregel, ohmsches Gesetz, Spannungs- und Stromteiler, Ersatzquellenverfahren</li> <li>• Potentiometer, resistive Sensoren (z.B. Länge, Druck, Licht, Temperatur, Kraft) und Ausschlagbrücke</li> <li>• Ladung, elektr. Feld, Influenz, Kapazität, Plattenkondensator, kapazitive Sensoren (z.B. Druck, Schall)</li> <li>• Durchflutung, magn. Fluss, magn. Widerstand, Induktivität, induktive Sensoren (z.B. Weg, Abstand)</li> <li>• Wechselstrom (Amplitude, Frequenz, Phase), komplexe Beschreibung von Wechselgrößen, Wechselstromverhalten von Kapazitäten und Induktivitäten</li> <li>• Messung von Wechselgrößen, Wechselstrom-basierte Sensoren (Metalldetektoren, Differential-sensoren wie Druckmessdose, Präzisionstaster oder Beschleunigungssensoren)</li> </ul>		
Lehr- und Lernformen	<p>SeU: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben</p> <p>Übung: Elektrische Grundlagenversuche mit Programmieren in 2-er Gruppen</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur</p> <p>Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p> <p>Voraussetzung (PVL): Übungstestat (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben		

Modulbezeichnung	<b>Automatentheorie und Formale Sprachen</b>	Kürzel	AF / AFÜ
Lehrveranstaltung(en)	SeU: Automatentheorie und Formale Sprachen Übung: Automatentheorie und Formale Sprachen	Fachsemester	2
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Bettina Buth	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Modul Diskrete Mathematik oder äquivalente Kenntnisse Anmerkung: benötigt werden Kenntnisse von formalen Beweisen, Prädikatenlogik, speziell Induktion	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	Die Studierenden <ul style="list-style-type: none"> <li>• können formale Beweise erläutern und selber durchführen</li> <li>• können formale Modelle in Form von Automaten, Regulären Ausdrücken und Grammatiken erstellen</li> <li>• können Zusammenhänge zwischen Automatenmodellen, regulären Sprachen und Grammatiken herstellen und Modelle ineinander überführen</li> <li>• können formale Spezifikationen auf Problemstellungen der realen Welt anwenden</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Automaten: Die Grundlagen und Methoden</li> <li>• Endliche Automaten</li> <li>• Reguläre Ausdrücke und Sprachen</li> <li>• Eigenschaften regulärer Sprachen</li> <li>• Kontextfreie Grammatiken und Sprachen</li> <li>• Eigenschaften kontextfreier Sprachen</li> <li>• Vertiefung in eine der folgenden Richtungen: Kellerautomaten, zelluläre Automaten, zeitbehaftete Automaten, Modellierung mit formalen Methoden</li> </ul>		
Lehr- und Lernformen	SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben, Gruppenarbeit Übung: selbstständiges Lösen der Übungsaufgaben		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Übungstestat (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"> <li>• J.E. Hopcroft, R. Motwani; J.D. Ullman: Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie, Addison-Wesley</li> <li>• M.Kreuzer, S. Kühling: Logik für Informatiker. Pearson Studium</li> </ul>		

Modulbezeichnung	<b>Programmiermethodik II</b>	Kürzel	PM2 / PMP2
Lehrveranstaltung(en)	SeU: Programmiermethodik II Praktikum: Programmiermethodik II	Fach-semester	2
Arbeitsaufwand	36 Std. SeU, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Michael Schäfers	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	keine	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• beherrschen fortgeschrittene Fertigkeiten und Programmiertechniken,</li> <li>• können Systeme im Kleinen objektorientiert modellieren, auch mit Hilfe der UML und</li> <li>• beherrschen die technische Realisierung objektorientierter und interaktiver Systeme.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Ausgewählte Elemente objektorientierter Bibliotheken, z.B.: <ul style="list-style-type: none"> <li>- GUI-Frameworks</li> <li>- Generics</li> </ul> </li> <li>• Metasprachliche Konzepte: <ul style="list-style-type: none"> <li>- Annotationen</li> <li>- Reflection</li> </ul> </li> <li>• Vertiefungen <ul style="list-style-type: none"> <li>- Typ- vs. Implementierungshierarchie</li> <li>- elementare Entwurfsmuster</li> <li>- Modellierungen (anhand UML)</li> <li>- nebenläufige bzw. asynchrone Programmierung</li> <li>- systematische Fehlerbehandlung</li> </ul> </li> <li>• Korrektheit <ul style="list-style-type: none"> <li>- Design by Contract (Assertions, Invarianten, Teststrategien)</li> </ul> </li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben, evtl. Tutorium Praktikum: Bearbeitung von Aufgaben in Zweiergruppen, Begutachtung der Lösungen, Gesprächsführung</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben Prüfungsvorleistung (PVL): Laborabschluss (erfolgreich durchgeführtes Praktikum)</p>		
Literatur	<ul style="list-style-type: none"> <li>• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben</li> </ul>		

Modulbezeichnung	<b>Datenbanken</b>	Kürzel	DB / DBP
Lehrveranstaltung(en)	SeU: Datenbanken Praktikum: Datenbanken	Fachsemester	2
Arbeitsaufwand	36 Std. SeU, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Thomas Clemen	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Programmieren I (PR1, PRP1) , Mathematische Grundlagen (MG)	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden:</p> <ul style="list-style-type: none"> <li>• kennen die Einsatzgebiete und Grenzen von relationalen Datenbanksystemen,</li> <li>• beherrschen den Prozess des Datenbankentwurfs,</li> <li>• kennen die theoretischen Grundlagen von relationalen Datenbanksystemen,</li> <li>• können einfache Datenbankanwendungen entwickeln,</li> <li>• beherrschen die relationale Anfragesprache SQL im Rahmen des Standards und</li> <li>• verstehen aktuelle gesellschaftliche Herausforderungen.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Grundkonzepte relationaler Datenbanksysteme</li> <li>• der logische Entwurf und die Überführung in das technische Design</li> <li>• Implementierung und Befüllung von Datenbanksystemen</li> <li>• Anfragen und Transaktionen</li> <li>• programmiersprachliche Schnittstellen</li> <li>• Alternativen zum relationalen Modell</li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in Kleingruppen</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>• A. Kemper, A. Eickler: Datenbanksysteme – Eine Einführung, Oldenbourg</li> <li>• R.A. Elmasri, S.B. Navathe: Grundlagen von Datenbanksystemen, Pearson</li> <li>• W. Gerken: Datenbanksysteme für Dummies. Wiley.</li> </ul>		

Modulbezeichnung	<b>Grundlagen der systemnahen Programmierung</b>	Kürzel	GS / GSP
Lehrveranstaltung(en)	SeU: Grundlagen der systemnahen Programmierung Praktikum: Grundlagen der systemnahen Programmierung	Fach-semester	2
Arbeitsaufwand	24 Std. SeU, 24 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Franz Korf	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Grundlagen der Technischen Informatik	SWS	2+2
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	Die Studierenden: <ul style="list-style-type: none"> <li>• können prozedurale Programme in C erstellen,</li> <li>• verstehen und benutzen Zugriffe auf Hardwareschnittstellen,</li> <li>• können Anwendungen zum Ansteuern einfacher Sensoren und Aktoren erstellen und</li> <li>• verstehen die Schnittstelle zwischen einer Hochsprache und einem Assembler.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Methoden und Techniken zur prozeduralen - und maschinennahen Programmierung</li> <li>• für die technische Informatik relevante Konzepte einer hardwarenahen Programmiersprache, wie zum Beispiel C</li> <li>• Speicherverwaltung auf Hochsprachen- und Maschinenebene</li> <li>• C Projekte: Verwaltung, Modultechik, Bibliotheken, Standardbibliotheken</li> <li>• Interfaces zur Verzahnung von Hochsprachen und Assembler</li> <li>• elementare Zeitmessungen</li> <li>• grundlegende Muster zur Programmierung eingebetteter Systeme (z.B. Direct Digital Control Programmiermuster)</li> </ul>		
Lehr- und Lernformen	SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben Praktikum: Programmieren in 2er Gruppen		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"> <li>• A. S. Tanenbaum, James Goodman: Computerarchitektur</li> <li>• Groll, U. Bröckl, M. Dausmann: C als erste Programmiersprache</li> <li>• B.W. Kernighan, D.M. Ritchie: Programmieren in C</li> <li>• W. Hohl: ARM Assembly Language: Fundamentals and Techniques</li> <li>• Michael Pont: Patterns for Time-Triggered Embedded Systems - Building Reliable Applications with the 8051 Family of Microcontrollers (with CD-ROM).</li> </ul>		

Modulbezeichnung	<b>Analysis und lineare Algebra</b>	Kürzel	AA /AAÜ
Lehrveranstaltung(en)	SeU: Analysis und lineare Algebra Praktikum: Analysis und lineare Algebra	Fachsemester	2
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Stephan Pareigis	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Diskrete Mathematik	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• beherrschen die analytischen Rechentechnik mit elementaren Funktionen einer Variablen</li> <li>• können mit Parametern, Transformationen und graphischen Darstellungen umgehen</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Kombinatorik, Teilbarkeit, Restklassen</li> <li>• Gruppen, Vektorräume, Homomorphismen</li> <li>• Skalarprodukt, Orthogonalität, Norm und Metrik</li> <li>• Differenzialrechnung, Integralrechnung, Fourierreihen.</li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Präsentation, Vorrechnen von Beispielaufgaben, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium</p> <p>Praktikum: selbstständige Lösung von Übungsaufgaben</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur</p> <p>Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p> <p>Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>• Peter Hartmann: Mathematik für Informatiker. Beispielhaft: Vieweg</li> </ul>		

Modulbezeichnung	<b>Signalverarbeitung und Stochastik</b>	Kürzel	SV / SVÜ
Lehrveranstaltung(en)	SeU: Signalverarbeitung und Stochastik Praktikum: Signalverarbeitung und Stochastik	Fachsemester	3
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Andreas Meisel	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Analysis und lineare Algebra	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• können auf der Basis von Elementarsignalen und einfachen Grundoperationen (Gewichtung, Verschiebung, Dehnung) periodische und abgetastete Funktionen beschreiben.</li> <li>• können mit Hilfe der Fouriertransformation (von Elementarsignalen) und den Theoremen der FT die Frequenzeigenschaften von Impulsen, periodischen und abgetasteten Signalen beschreiben.</li> <li>• können den Einfluss von Frequenzbegrenzungen (Filter: TP, HP, BP) und Zeitbegrenzungen (Fensterfunktionen) auf die Zeit- und Frequenzeigenschaften der Signale beschreiben.</li> <li>• können Abtastsignale mit Hilfe der z-Transformation beschreiben und sowohl die Differenzgleichungen als auch die Übertragungsfunktionen digitaler Filter angeben.</li> <li>• können mit Hilfe von Werkzeugen digitale Filter (FIR, IIR) entwerfen und diese auf Zielsystemen (z.B. eingebetteten Systemen) unter Benutzung von Hardwaretimern und Interrupts realisieren.</li> <li>• können Verfahren zur Bestimmung von statistischen Signalkenngrößen (Erwartungswert, Varianz, Momente) und Ähnlichkeitsmaßen (Kreuzkorrelation, Autokorrelation) realisieren und die Ergebnisse interpretieren.</li> <li>• können ausgewählte Merkmale (zerocrossings, Momente, usw.) aus verschiedenen Anwendungsbereichen (Audioprocessing, medizinische Signalverarbeitung, usw.) aus Signalen extrahieren.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Elementarsignale, Grundoperationen (Verschieben, Dehnen, Abtastung, ..), Faltungsalgebra</li> <li>• Fouriertransformation, Faltungssatz, Diracstoß und Diracstoßfolgen</li> <li>• Abtastung, Periodisierung, ideale Filter, diskrete FT</li> <li>• Abtastsysteme, z-Transformation, Differenzgleichungen, Übertragungsfunktion, FIR- und IIR-Filter</li> <li>• Diskrete Fouriertransformation, Bandbegrenzung (Filterung), Zeitbegrenzung (Fensterung)</li> <li>• Entwurf und Realisierung digitaler Filter, schmalbandige Filter, Görtzel-Algorithmus</li> <li>• Realisierung von Abtastsystemen (Filter, Audioeffekte, ...) mit Hardwaretimern und Interrupts unter Nutzung einer hardwarenahen Programmiersprache (z.B. C).</li> <li>• Grundbegriffe der Statistik: Wahrscheinlichkeitsbegriff, Zufallsvariablen, Verteilungs- und Verteilungsdichtefunktion, stat. Unabhängigkeit, bedingte Wahrscheinlichkeit, Satz von Bayes</li> <li>• Signale als Zufallsprozesse: Erwartungswert, Varianz, Schätzung stat. Signalkenngrößen</li> <li>• Ausgewählte Signalmerkmale aus verschiedenen Anwendungsbereichen</li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Präsentation, Matlab Beispiele, freiwillige Übungsaufgaben Praktikum: Bearbeitung von Aufgaben in 2-er Gruppen mit Abnahmegespräch</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p> <p>Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		

Modulbezeichnung	<b>Signalverarbeitung und Stochastik</b>	Kürzel	SV / SVÜ
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"><li>• Hoffmann und Quint: Signalverarbeitung mit MATLAB und Simulink. Oldenbourg Verlag</li><li>• D.Ch. von Grüningen: Digitale Signalverarbeitung. Fachbuchverlag Leipzig</li></ul>		

Modulbezeichnung	<b>Algorithmen und Datenstrukturen</b>	Kürzel	AD /ADP
Lehrveranstaltung(en)	SeU: Algorithmen und Datenstrukturen Praktikum: Algorithmen und Datenstrukturen	Fachsemester	3
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Stephan Pareigis	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Diskrete Mathematik, Programmieren I + II	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>haben Kenntnisse zur Bewertung und selbstständigen Entwicklung von Algorithmen und beherrschen die dazu erforderlichen Datenstrukturen.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>Abstrakte Datentypen Abstrakte Datentypen, Signatur, Vor- und Nachbedingungen, Lineare Listen, Stack, Queue</li> <li>Algorithmen und Komplexität Komplexität, Aufwandsfunktion, Asymptotischer Aufwand, Landau-Notation, Darstellung in logarithmischen Skalen, rekursive Verfahren</li> <li>Sortierverfahren Nicht-rekursive Sortierverfahren und deren Komplexität, Rekursive Sortierverfahren (Quicksort und Mergesort) und deren Komplexität</li> <li>Bäume und Graphen Implementationsmöglichkeiten, Binäre Suchbäume und Komplexität, Graphen und kürzeste Wege (Dijkstra)</li> <li>Hashfunktionen offene Adressierung und separate chaining, Kollisionsvermeidungsstrategien</li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Vorrechnen von Beispielaufgaben, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium</p> <p>Praktikum: selbstständige Lösung von Praktikumsaufgaben</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur</p> <p>Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p> <p>Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>Ottman, Widmayer: Algorithmen und Datenstrukturen. Spektrum Verlag</li> </ul>		

Modulbezeichnung	<b>Software Engineering I</b>	Kürzel	SE1 / SEP1
Lehrveranstaltung(en)	SeU: Software Engineering I Praktikum: Software Engineering I	Fachsemester	3
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Bettina Buth	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Programmiermethodik I, Programmiertechnik I, Programmiermethodik II	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• können die Zusammenhänge zwischen Programmierung und Software Engineering erläutern</li> <li>• können Fachbegriffe des Bereichs Software Engineering erläutern</li> <li>• können vorgegebene Modelle (insbesondere Prozessmodelle und Modelle der UML Notationsfamilie) im Rahmen einer Softwareentwicklung interpretieren und analysieren, anwenden und in Code umsetzen.</li> <li>• können die Besonderheiten des Software Engineering bei technischen Anwendungen benennen und erläutern</li> <li>• können programmiernahe Techniken und Werkzeuge im Rahmen des Software Engineering einsetzen, speziell für die Anforderungsanalyse, die Modellierung und den Test.</li> <li>• können statische Analysen ebenso wie grundlegende dynamische Testtechniken anwenden</li> <li>• können sich eigenständig neue Techniken und Methoden und deren Anwendung an konkreten Beispielen aneignen</li> <li>• können fachliche Zusammenhänge schriftlich ausdrücken, speziell Dokumentation von Anforderungen, Architekturen, Tests erstellen und bewerten</li> <li>• können englische Dokumentationen interpretieren</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Grundkonzepte und Ziele des Software Engineering, insbesondere Lebenszyklusmodelle, Anforderungserhebung- und Analyse, Architekturen, Qualitätssicherung,</li> <li>• Modellierungstechniken, speziell auch im Hinblick auf technische Systeme,</li> <li>• Spezielle Entwicklungstechniken, wie z. B. Refactoring, Code Analyse, Continuous Integration, DSLs, Code Generierung,</li> <li>• Requirements Engineering, speziell Anforderungserhebung und -analyse</li> <li>• Prozessmodelle, speziell traditionelle Lebenszyklusmodelle im Vergleich zu agilen Entwicklungsansätzen</li> <li>• Im Praktikum: Arbeiten mit aktuellen Software-Entwicklungsumgebungen speziell zu Modellierung, Test, Refactoring, Requirements Engineering – auch: Beherrschen von Werkzeugketten (Konfiguration, Einsatz)</li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Präsentation, Übungsaufgaben, Gruppenarbeit Praktikum: Aufgabenbearbeitung in 2er-4er Teams, Diskussionsforen mit Betreuer</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		

Modulbezeichnung	<b>Software Engineering I</b>	Kürzel	SE1 / SEP1
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"><li>• Sommerville: Software Engineering. Pearson Education</li><li>• Spillner, Linz: Basiswissen Softwaretest. dpunkt Verlag</li><li>• Hammerschall, Beneken: Software Requirements</li></ul>		

Modulbezeichnung	<b>Betriebssysteme</b>	Kürzel	BS / BSP
Lehrveranstaltung(en)	SeU: Betriebssysteme Praktikum: Betriebssysteme	Fachsemester	3
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Franz Korf	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Programmiermethodik I, Programmiertechnik, Grundlagen der Technischen Informatik	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>verstehen die Architektur, die Konzepte und die Funktionsweise moderner Betriebssysteme sowie des Zusammenspiels von Hard- und Software,</li> <li>verstehen die Konzepte zur Implementierung systemnaher Software</li> <li>können das Verhalten von Computersystemen analysieren und beschreiben</li> <li>können Grundkonzepte der nebenläufigen Programmierung anwenden</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>Architekturen und Betriebsarten</li> <li>Prozess- und Thread-Konzept, Scheduling</li> <li>Synchronisation, Interprozesskommunikation, Deadlocks</li> <li>Hauptspeicherverwaltung, Virtueller Speicher</li> <li>Verwaltung externer Geräte</li> <li>Dateisysteme</li> <li>Schutzmechanismen, Sicherheitsaspekte</li> <li>Exemplarische Betrachtung aktueller Betriebssysteme</li> </ul>		
Lehr- und Lernformen	<p>SeU: Seminaristischer Unterricht, Tafelarbeit, Präsentation, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>Andrew S. Tanenbaum, Modern Operating Systems, Pearson Studium Verlag</li> <li>Silberschatz, Galvin, Gagne: Operating System Concepts with Java. John Wiley &amp; Sons</li> <li>Eduard Glatz: Betriebssysteme - Grundlagen, Konzepte, Systemprogrammierung. dpunkt Verlag</li> <li>Williams Stallings: Operating Systems</li> </ul>		

Modulbezeichnung	<b>Intelligente Sensorsysteme</b>	Kürzel	ISS /ISSP
Lehrveranstaltung(en)	SeU: Intelligente Sensorsysteme Praktikum: Intelligente Sensorsysteme	Fach-semester	3
Arbeitsaufwand	36 Std. SeU, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Tim Tiedemann	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Programmiermethodik I, Programmiermethodik II, Mess- und Sensortechnik	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• können Sensorsysteme zur Erfassung physikalischer Größen entsprechend dem geforderten Einsatzbereich, den jeweiligen Datenblättern und Herstellerdokumentationen bewerten und auswählen,</li> <li>• sind in der Lage, die physikalischen Prinzipien, die Sensoren (etwa für Annäherung, Entfernung, Druck, Temperatur, Durchfluss, Beschleunigung, Drehung und Kraft) zugrunde liegen, zu beschreiben,</li> <li>• können die wesentlichen Grundlagen und Randbedingungen für den Einsatz komplexer Sensoren (wie beispielsweise Kamera, Mikrofon, Laserscanner) zusammenfassen,</li> <li>• können intelligente Sensoren mit Hilfe der Herstellerdokumentation über gängige Sensor-Bussysteme, wie z.B. I2C, CAN oder Ethernet, an Rechnersysteme anbinden,</li> <li>• können die Daten intelligenter Sensoren mit Hilfe der Herstellerdokumentation auf Rechnersystemen erfassen und Daten verschiedener Sensorsysteme kombinieren,</li> <li>• können beispielhaft mittels verschiedener Algorithmen einfache Rohsensordaten zu abstrakteren/komplexeren Informationen (sog. Features) aufbereiten/vorverarbeiten.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Physikalische Messprinzipien beispielsweise für Annäherung, Entfernung, Temperatur, Druck, Durchfluss, Beschleunigung, Drehung und Kraft. Wirkungsweise von z.B. Mikrofonen, Kameras, Laserscannern, Radarsensoren.</li> <li>• Konkrete Beispiele typischer Sensoren, insbesondere typischer intelligenter Sensoren.</li> <li>• Exemplarische Erörterung von Datenblättern.</li> <li>• Elektrischer Anschluss von Sensoren. Dazu – soweit für das Verständnis der Sensoren benötigt – Eigenschaften und Anwendungen von Dioden, Transistoren oder Operationsverstärkern.</li> <li>• Vorverarbeitung: Abtastung, Glättung, Mittelwertbildung, grafische Darstellung.</li> <li>• Generierung und Auswahl von Features aus Rohdaten, z.B. mittels maschineller Lernverfahren, Bildverarbeitungs- oder statistischer Methoden. Sensordatenfusion.</li> <li>• Schnittstellen/Bussysteme von intelligenten Sensoren: I2C, Ethernet, u.a. Rechnersysteme für den Anschluss von Sensoren, z.B. Industrie-PCs, on-board units (OBU), aber auch Arduino, Raspberry Pi oder Beaglebone Black.</li> <li>• Zeitverhalten von Rechnersystemen zur Auswertung von Sensordaten.</li> <li>• Programmtechnische Konfiguration und Programmierung von Sensoren.</li> <li>• Anwendungsbeispiele wie KFZ-Sensorik, industrielle Steuerungs- und Automatisierungstechnik, Smart Gadgets.</li> </ul>		
Lehr- und Lernformen	SeU: Seminaristischer Unterricht, Tafelarbeit, freiwillige Übungsaufgaben Praktikum: Sensor-Anwendungen mit Programmieren in 2-er-Gruppen		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit		

Modulbezeichnung	<b>Intelligente Sensorsysteme</b>	Kürzel	ISS /ISSP
	Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL):            Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"> <li>• Meijer et al. (Eds): Smart Sensor Systems. Wiley</li> <li>• Kimmo Karvinen et. al.: Sensoren. dpunkt-Verlag</li> </ul>		

Modulbezeichnung	<b>Mustererkennung und Machine Learning</b>	Kürzel	MM/MMP
Lehrveranstaltung(en)	SeU: Mustererkennung und Machine Learning Praktikum: Mustererkennung und Machine Learning	Semester	4
Arbeitsaufwand	36 Std. SeU, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Andreas Meisel	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Analysis und lineare Algebra, Signalverarbeitung und Stochastik	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• können mit ein- oder mehrdimensionale Zeitsignalen (z.B. von einem Sensorarray), Bildern oder auch Bildsequenzen datentechnisch umgehen (Datenstrukturen, effiziente Verarbeitung).</li> <li>• können zu einem gegebenen Mustererkennungsproblem (MEP) mit Hilfe von Faltungsoperatoren, Korrelation, spektralen Merkmalen, statistischen Merkmalen und ausgewählten höheren Merkmalen geeignete Merkmalsextraktionsverfahren auswählen und realisieren.</li> <li>• können zu einem MEP ein passendes Klassifikatorkonzept angeben. Dabei berücksichtigen sie, dass dieses analytisch (z.B. Fourierdeskriptoren), regelbasiert (scharf oder fuzzy) oder lernbasiert (statistisch, neuronal) sein kann und wägen die unterschiedlichen Vor- und Nachteile gegeneinander ab.</li> <li>• können passend zum Klassifikationskonzept einen geeigneten Klassifikator auswählen, konfigurieren und realisieren. Sie können lernende Systeme trainieren, testen und in die Zielanwendung integrieren.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Stufen der Mustererkennung (Vorverarbeitung, Merkmalsextraktion, Klassifikation)</li> <li>• analytische, regelbasierte und lernbasierte Verfahren der Mustererkennung</li> <li>• Grundzüge der Wahrscheinlichkeitstheorie, bedingte Wahrscheinlichkeit, Bayes-Regel</li> <li>• ausgewählte Verfahren zur Merkmalsextraktion, Hauptkomponentenanalyse</li> <li>• Klassifikation durch Funktionsapproximation (Polynomklassifikator, radiale Basisfunktionen)</li> <li>• Curse of dimensionality (Problem hochdimensionaler Merkmalsräume), Gradientenabstieg</li> <li>• Clusteringverfahren (z.B. k-Means, Self-organizing-map)</li> <li>• Grundlagen neuronaler Netze (NN): Neuron, Aktivierungsfunktion, Multilayer-Perzeptron Backpropagation u. Erweiterungen, Training, Overfitting</li> <li>• Support Vektor Machines und Kernelmethoden</li> <li>• Moving-window-NN und Recurrent NN für Zeitsignale, Dynamic-time-Warping</li> <li>• Konzepte des Deep-Learning</li> <li>• Ausgewählte Themen: Convolutional Neural Networks, LSTM-Netze</li> </ul>		
Lehr- und Lernformen	SeU: Seminaristischer Unterricht, Tafelarbeit, Präsentation, Vorrechnung von Beispielaufgaben, Applets und Matlab-Skripte zur Veranschaulichung, freiwillige Übungsaufgaben Praktikum: selbstständige Lösung von Praktikumsaufgaben in 2-er Gruppen		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur</p> <p>Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p> <p>Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>• Duda, Hart, Stork: Pattern Classification. Wiley interscience</li> <li>• Bishop: Pattern Recognition and Machine Learning. Springer</li> </ul>		

Modulbezeichnung	<b>Mustererkennung und Machine Learning</b>	Kürzel	MM/MMP
	<ul style="list-style-type: none"><li>• Haykin: Neural Networks and Learning Machines. Pearson international</li><li>• Goodfellow , Bengio , Courville: Deep Learning - Adaptive Computation and Machine Learning. MIT Press</li></ul>		

Modulbezeichnung	<b>Embedded System Engineering</b>	Kürzel	ESE / ESEP
Lehrveranstaltung(en)	SeU: Software Engineering II, Embedded Programming, System- und Echtzeitprogrammierung Praktikum: Embedded System Engineering	Fachsemester	4
Arbeitsaufwand	72 Std. SeU, 24 Std. Praktikum, 204 Std. Eigenarbeit/Selbststudium	Dauer	zwei Semester
Modulverantwortliche(r)	Prof. Dr. Franz Korf	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	12
Voraussetzungen	Empfohlen: Programmiermethodik I, Programmiermethodik II, Software-Engineering 1, Mess- und Sensortechnik, Grundlagen der systemnahen Programmierung, Betriebssysteme	SWS	6+2
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• können konkrete Projekte unter Anwendung von Projektmanagementmethoden planen und durchführen, insbesondere unter Berücksichtigung unterschiedlicher Entwicklungsprozesse</li> <li>• können spezielle Aspekte eingebetteter Systeme, wie z. B. Safety, Robustness oder Fehlertoleranz, beim Entwurf und Entwicklung berücksichtigen</li> <li>• können technische Systeme auf der Systemebene modellieren und eingebettete Software als Teilsystem modellbasiert entwerfen, realisieren und testen</li> <li>• können verteilte nebenläufige Echtzeitsysteme auf Basis objektorientierter Architekturen unter Verwendung spezialisierter implementierungsnaher Pattern entwerfen und implementieren</li> <li>• können Echtzeitsysteme entsprechend ihres Einsatzes und Anforderungen klassifizieren</li> <li>• können ausgewählter Aspekte eingebetteter Echtzeitsysteme im Zusammenhang mit einem Echtzeitbetriebssystem (z. B. Zeitmanagement, Interrupt Management, Kommunikation, I/O) konzipieren und umsetzen</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• SeU Embedded Programming (EP) <ul style="list-style-type: none"> <li>- C++-Sprachprinzipien und Eignung für Echtzeitprogrammierung</li> <li>- Kapselung von Systemaufrufen, plattformunabhängige Programmierung</li> <li>- Reaktor Pattern, Speicherverwaltung, Fabrikpattern, Functor Pattern</li> <li>- Kommunikationsprinzipien und Message Passing</li> <li>- Generische Programmierung mit Templates</li> <li>- Code-Instrumentalisierung, Logging, Debuggen von Echtzeitsystemen</li> </ul> </li> <li>• Vorlesung Software Engineering II (SE2) <ul style="list-style-type: none"> <li>- Einführung in das Projektmanagement, Projektplanung und Teamorganisation</li> <li>- Grundlagen des Konfigurations- und Versionsmanagements</li> <li>- Einführung in das Qualitätsmanagement</li> <li>- Spezielle Aspekte des Software Engineering bei eingebetteten Systemen</li> <li>- Spezialisierungen von Techniken der Modellierens und des Testens auf eingebettete Echtzeitsysteme</li> </ul> </li> <li>• Vorlesung System- und Echtzeitprogrammierung (SY) <ul style="list-style-type: none"> <li>- Grundlagen, Klassifizierung, Einsatz, Anforderungen von eingebetteten Systemen.</li> <li>- Vertiefung ausgewählter Aspekte eingebetteter Echtzeitsysteme unter Einsatz eines Echtzeitbetriebssystems (z. B.: Zeitmanagement, Interrupt Management, Kommunikation, I/O).</li> <li>- Scheduling-Techniken für Echtzeitanwendungen</li> <li>- Umsetzung der Aspekte anhand eines ausgewählten Echtzeitbetriebssystems (z. B. QNX)</li> </ul> </li> </ul>		

Modulbezeichnung	<b>Embedded System Engineering</b>	Kürzel	ESE / ESEP
	<ul style="list-style-type: none"> <li>• Praktikum                             <ul style="list-style-type: none"> <li>- Projektorientierte Software-Entwicklung für ein ES auf Basis eines ausgewählten Echtzeitbetriebssystems (z. B. QNX). Das Praktikum erfordert im besonderen Maße auch Techniken und Fähigkeiten der vorausgesetzten Module.</li> <li>- Teamorientierte Softwareentwicklung</li> </ul> </li> </ul>		
Lehr- und Lernformen	SeU: Seminaristischer Unterricht mit verschiedenen Medien Praktikum: Projektorientierte Entwicklung eines eingebetteten System in kleinen Teams (4-6 Personen)		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"> <li>• Sommerville: Software Engineering.</li> <li>• (GOF) Gamma, Erich et al.: Design Patterns. Addison Wesley</li> <li>• Douglas Schmidt, Stephen Huston: C++ Network Programming. Volume 1, Addison Wesley</li> <li>• (POSA 2) Schmidt, Stal, Rohnert, Buschmann: Pattern-Oriented Software Architecture. Wiley</li> <li>• Bruce Powell Douglas: Real-Time Design Patterns. Addison-Wesley</li> <li>• Andrei Alexandrescu: Modernes C++ Design. MITP</li> <li>• Kaley, Danny: The ANSI/ISO C++ Professional Programmer's Handbook. Que Corporation.</li> <li>• Meyers, Scott: Effective C++. Addison Wesley</li> <li>• Meyers, Scott: More Effective C++. Addison Wesley</li> <li>• A. Burns und A. Wellings: Real-time systems and programming languages : Ada 95, real-time Java and real-time POSIX. 3. ed, Pearson Addison-Wesley.</li> <li>• G. C. Buttazzo: Hard real-time computing systems: predictable scheduling algorithms and applications. Nr. 23 in Real-time systems series, Springer</li> <li>• H. Kopetz: Real-time systems - Design principles for distributed embedded applications, Kluwer Acad.</li> <li>• R. Krten: Getting started with QNX Neutrino : a guide for realtime programmers. QNX Software Systems.</li> <li>• W. Stallings: Operating systems : internals and design principles. Pearson.</li> </ul>		

Modulbezeichnung	<b>Rechnernetze</b>	Kürzel	RN / RNP
Lehrveranstaltung(en)	SeU: Rechnernetze Praktikum: Rechnernetze	Fachsemester	4
Arbeitsaufwand	36 Std. SeU, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Thomas Schmidt	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Programmieren, Betriebssysteme	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>verstehen die Konzepte und die Funktionsweisen von Rechnernetzen</li> <li>können auf der Socket-Schnittstelle basierende Client- / Server-Anwendungen erstellen</li> <li>können Methoden und Werkzeuge für die Konfiguration und Administration von Rechnernetzen anwenden</li> <li>können die Leistungsdaten von Rechnernetzen bewerten</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>Grundlagen der Datenübertragung</li> <li>Protokolle der Sicherungsschicht</li> <li>Protokolle und Dienste der Netzwerk- und Transportschicht, insbesondere die TCP/IP-Protokollsuite</li> <li>Einführung in wichtige Anwendungsschichtprotokolle</li> <li>Sicherheit in Netzwerken</li> <li>Einführung in Netzwerkmanagement</li> <li>Socket-Programmierung</li> </ul>		
Lehr- und Lernformen	<p>SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in 2er Gruppen</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>Andrew S. Tanenbaum, David Wetherall: Computer Networks</li> <li>Larry L. Peterson, Bruce S. Davie: Computer Networks – A Systems Approach</li> <li>James F. Kurose, Keith W. Ross: Computer Networking: A Top-Down Approach Featuring the Internet</li> <li>Ch. Meinel, H. Sack: Internetworking</li> </ul>		

Modulbezeichnung	<b>Verteilte Systeme</b>	Kürzel	VS / VSP
Lehrveranstaltung(en)	SeU: Verteilte Systeme Praktikum: Verteilte Systeme	Fachsemester	5
Arbeitsaufwand	36 Std. SeU, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Thomas Schmidt	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Empfohlen: Betriebssysteme, Rechnernetze, Software Engineering I, Software Engineering II, Algorithmen und Datenstrukturen	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>kennen und verstehen die Verteilten Systemen zugrunde liegenden Probleme sowie die einschlägigen Leistungskategorien zu ihrer Lösung</li> <li>beherrschen und verstehen einschlägige verteilte Programmiermodelle, können eine exemplarische Auswahl praktisch anwenden und können den zugehörigen Lösungsraum technisch beurteilen</li> <li>beherrschen und verstehen einschlägige Algorithmen zur Realisierung verteilter Anwendungsszenarien und können diese auf reale Probleme übertragen</li> <li>können eine System-Infrastruktur eines VS entwerfen und realisieren</li> <li>können eine Middleware eines VS entwerfen und realisieren</li> <li>können ein Konzept für replizierte Daten entwerfen und realisieren</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>Eine Einführung im Sinne einer Beschreibung der charakteristischen Eigenschaften verteilter Systeme</li> <li>Interprozesskommunikation zwischen verteilten Prozessen und einschlägige Programmiermodelle</li> <li>Namensdienste und exemplarische Anwendungen</li> <li>Zeit, Koordination und Übereinstimmung</li> <li>Wahlen, Wechselseitiger Ausschluss und Verteilte Transaktion</li> <li>Verteilte Dateisysteme und Replikation</li> <li>Ausgewählte Anwendungen verteilter replizierender Systeme</li> <li>Sicherheit in verteilten Systemen</li> </ul>		
Lehr- und Lernformen	SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben Praktikum: Bearbeitung von Aufgaben in Zweiergruppen mit abschließendem Abnahmegespräch		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur</p> <p>Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p> <p>Voraussetzung (PVL): Übungstestat oder Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<ul style="list-style-type: none"> <li>Jeweils in der aktuellen Auflage:</li> <li>G. Coulouris, J. Dollimore, T. Kindberg. Distributed Systems: Concepts and Design.</li> <li>A.S. Tanenbaum, M.v. Stehen. Distributed Systems: Principles and Paradigms</li> <li>Skripte der Dozenten</li> </ul>		

Modulbezeichnung	<b>Betriebswirtschaft</b>	Kürzel	BW / BWÜ
Lehrveranstaltung(en)	SeU: Betriebswirtschaft Praktikum: Betriebswirtschaft	Fach-semester	5
Arbeitsaufwand	36 Std. SeU, 12 Std. Übungen, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Martin Hübner	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	keine	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	Die Studierenden <ul style="list-style-type: none"> <li>• verstehen rechtliche, finanzielle und organisatorische Strukturen von Unternehmen,</li> <li>• verstehen die Bedeutung von wirtschaftlichen Vorgehensweisen und können entsprechend Controlling-Instrumente anwenden</li> <li>• können Kostenberechnungen selbstständig durchführen,</li> <li>• können Investitionsentscheidungen nach betriebswirtschaftlichen Kriterien treffen.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>• Das Unternehmen als System</li> <li>• Rechtsformen und Aufbauorganisation</li> <li>• Ablauforganisation und Methoden zu ihrer Beschreibung</li> <li>• Grundlagen der Finanzbuchhaltung (Buchführung und Jahresabschluss)</li> <li>• Kosten- und Leistungsrechnung</li> <li>• Finanzierung und Investitionsrechnung</li> </ul>		
Lehr- und Lernformen	SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben Praktikum: selbstständiges Lösung von Übungsaufgaben		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"> <li>• G. Wöhe: Einführung in die allgemeine BWL. Verlag Franz Vahlen</li> <li>• A. J. Schwab: Managementwissen für Ingenieure. Springer-Verlag</li> <li>• Dietmar Vahs, Jan Schäfer-Kunz: Einführung in die BWL. Schäffer-Poeschel Verlag</li> <li>• Siegfried Schmolke, Manfred Deitermann: Industrielles Rechnungswesen IKR. Winklers Verlag</li> </ul>		

Modulbezeichnung	<b>Projekt</b>	Kürzel	PRO
Lehrveranstaltung(en)	Pj: Projekt	Semester	5
Arbeitsaufwand	270 Std. Pj	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Martin Becke	Turnus	semesterweise
Art des Moduls	Wahlpflichtmodul	CP	9
Voraussetzungen	je nach Projekt	SWS	6
Verwendbarkeit	für die Studiengänge „Angewandte Informatik“, „Informatik Technischer Systeme“, „Wirtschaftsinformatik“, „European Computer Science“	Sprache	deutsch oder englisch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>haben die Fähigkeit zur Lösung informatikspezifischer Probleme unter Berücksichtigung begrenzter Ressourcen (Zeit, Mitarbeiter, Werkzeuge etc.), zur Spezifikation von Anforderungen, zur Modellierung von Systemen, zur Zielsetzung und Planung von Projekten, zur Sicherung der Qualität, zur Vor- und Nachkalkulation des Zeitaufwandes und zur verständlichen Dokumentation.</li> <li>haben die Fähigkeit zur Teamarbeit mit Entwicklern und (wenn möglich) Anwendern, speziell: zur Präsentation von Arbeitsergebnissen, zur Leitung und Moderation von Besprechungen, zur Lösung von Konflikten und zur Beurteilung von Arbeitsergebnissen.</li> </ul>		
Inhalte	<p>Die Studierenden wählen die Projekte aus einem Pool, der für die Informatik-Bachelorstudiengänge des Department Informatik gemeinsam angeboten wird.</p> <p>Die in den Projekten direkt benötigten spezifischen Kenntnisse – sowohl aus dem anwendungs- und berufsbezogenen als auch aus dem informatischen und mathematischen Bereich – werden in Blockveranstaltungen vermittelt bzw. erarbeitet. Soweit im Rahmen des jeweiligen Projekts machbar, sollte den Studierenden Gelegenheit gegeben werden, die Ermittlung fachlicher Anforderungen in Interviews mit „echten Kunden“ zu trainieren. Im Bereich Dokumentation sollte zur Schärfung des Problembewusstseins die Dokumentation von Vorgänger- oder Zuarbeitergruppen genutzt werden.</p> <p>Regelmäßige Projektsitzungen geben den Studierenden die Möglichkeit, die die oben genannten Fähigkeiten zur Teamarbeit durch Einübung zu erwerben. Dabei wird insbesondere die Qualitätssicherung durch Präsentation von Ergebnissen aus Analyse, Entwurf und Implementierung trainiert.</p>		
Lehr- und Lernformen	Pj		
Studien- und Prüfungsleistungen	Unbenotetes Projekt		
Literatur	abhängig vom Projektthema		

Modulbezeichnung	<b>Seminar</b>	Kürzel	SEM
Lehrveranstaltung(en)	S: Seminar	Semester	5
Arbeitsaufwand	90 Std. S	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Bettina Buth	Turnus	semesterweise
Art des Moduls	Wahlpflichtmodul	CP	3
Voraussetzungen	-	SWS	2
Verwendbarkeit	für die Studiengänge „Angewandte Informatik“, „Informatik Technischer Systeme“, „Wirtschaftsinformatik“, „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	<p>Die Studierenden</p> <ul style="list-style-type: none"> <li>• sind in der Lage, ein Thema selbständig zu erarbeiten</li> <li>• beherrschen die Methodik zur sachgerechten Literaturrecherche</li> <li>• sind in der Lage, einen eigenen Standpunkt herauszuarbeiten</li> <li>• können überzeugend argumentieren</li> <li>• können eine professionelle Präsentation erstellen</li> <li>• sind in der Lage, eine themenzentrierte konstruktive Diskussion zu führen</li> </ul>		
Inhalte	<p>Es werden Themen zu einem gegebenen Themengebiet oder nach eigener Wahl und Absprache mit dem Betreuer bearbeitet.</p> <p>Die Vorbereitung erfolgt unter individueller Betreuung durch eine(n) der veranstaltenden Professorinnen oder Professoren.</p> <p>Dabei wird auf methodische inhaltliche Arbeit ebenso geachtet wie auf eine gute didaktische Aufbereitung und eine professionelle Präsentation. Dazu gehört auch ein Probevortrag vor der betreuenden Professorin bzw. dem betreuenden Professor.</p>		
Lehr- und Lernformen	S: Kritik an Inhalt und Form durch Betreuer und Studierende		
Studien- und Prüfungsleistungen	Unbenotetes Referat		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>• Martin Hartmann, Rüdiger Funk, Horst Nietmann: Präsentieren; Beltz</li> <li>• Josef W. Seifert: Visualisieren, Präsentieren, Moderieren; Gabal</li> <li>• Christian W. Dawson: Computerprojekte im Klartext; Pearson Studium</li> <li>• Fachliteratur abhängig vom Thema</li> </ul>		

Modulbezeichnung	<b>Cyber-physische Systeme</b>	Kürzel	CPS / CPSP
Lehrveranstaltung(en)	SeU: Cyber-physische Systeme Praktikum: Cyber-physische Systeme	Fachsemester	6
Arbeitsaufwand	36 Std. SeU, 12 Std. Übungen, 132 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Jan Sudeikat	Turnus	semesterweise
Art des Moduls	Pflichtmodul	CP	6
Voraussetzungen	Programmieren, Rechnerstrukturen, Maschinennahe Programmierung, Verteilte Systeme	SWS	3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch
Lernziele und Kompetenzen	Die Studierenden <ul style="list-style-type: none"> <li>verstehen die Anforderungen einer Verknüpfung von realen Objekten und Prozessen mit informationsverarbeitenden Objekten und Prozessen</li> <li>können reale und informationsverarbeitende Objekte in lokalen, wie auch globalen Kommunikationsnetze verbinden</li> <li>verstehen die Kopplung von physikalischen und virtuellen Objekten</li> </ul>		
Inhalte	Die Veranstaltungen fokussiert sich auf domainübergreifende und offene Systeme. <ul style="list-style-type: none"> <li>Einführung von CPS als neues Kommunikationsparadigma, das auf den verteilten Systemen aufbaut.</li> <li>Adaption geänderter oder sich ändernder Umgebungsbedingungen</li> <li>virtuelle Abbildung von physikalischen Interaktionen</li> <li>unterschiedliche QoS Anforderungen innerhalb eines Systems.</li> <li>Besprechung der Modellierung zur Umsetzung z.B. aus den Bereichen Car-to-X, Smart Grid, Produkt- oder Produktionssystemen</li> <li>Modellierungs-, Spezifikations- und Verifikationstechniken für offene Systeme</li> </ul>		
Lehr- und Lernformen	SeU: Tafelarbeit, Rechnerpräsentation, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in 2er Gruppen		
Studien- und Prüfungsleistungen	Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben. Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)		
Literatur	Jeweils in der aktuellen Auflage: <ul style="list-style-type: none"> <li>Rajeev Alur: Principles of Cyber-Physical Systems. MIT Press</li> <li>Christian Manzei , Linus Schlepner: Industrie 4.0 im internationalen Kontext - Kernkonzepte, Ergebnisse, Trends. VDE-Verlag</li> <li>Dr. William Stallings: Foundations of Modern Networking: SDN, NFV, QoE, IoT and Cloud. Addison-Wesley</li> </ul>		

Modulbezeichnung	<b>Bachelorarbeit</b>	Kürzel	BA / BAK
Lehrveranstaltung(en)	Bachelorarbeit	Semester	6
Arbeitsaufwand	360 Std. Bachelorarbeit, 90 Std. Kolloquiumsvorbereitung und -durchführung	Dauer	6 Monate
Modulverantwortliche(r)	Prof. Dr. Martin Becke	Turnus	regelmäßig
Art des Moduls	Pflichtmodul	CP	12 + 3
Voraussetzungen	Die Bachelorarbeit kann angemeldet werden, wenn alle bis auf drei Module erfolgreich abgelegt worden sind. Der Umfang der noch nicht erfolgreich abgelegten Module darf 18 Leistungspunkte nicht übersteigen.	SWS	-
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch oder englisch
Lernziele und Kompetenzen	<p>In der Bachelorarbeit zeigen die Studierenden, dass sie in der Lage sind, ein Problem aus den wissenschaftlichen, anwendungsorientierten oder beruflichen Tätigkeitsfeldern dieses Studiengangs selbständig unter Anwendung wissenschaftlicher Methoden und Erkenntnisse zu bearbeiten und dabei in die fächerübergreifenden Zusammenhänge einzuordnen.</p> <p>Folgende Kompetenzen werden erworben:</p> <ul style="list-style-type: none"> <li>• Kompetenz, sich in das Thema einzuarbeiten, es einzuordnen, einzugrenzen, kritisch zu bewerten und weiter zu entwickeln</li> <li>• Kompetenz, das Thema anschaulich und formal angemessen in einem bestimmten Umfang schriftlich darzustellen</li> <li>• Kompetenz, die wesentlichen Ergebnisse der Arbeit fachgerecht und anschaulich in einem Vortrag einer vorgegebenen Dauer zu präsentieren</li> <li>• Kompetenz, aktiv zu fachlichen Diskussionen beizutragen</li> </ul>		
Inhalte	<p>Die Bachelorarbeit ist eine theoretische, empirische und/oder experimentelle Abschlussarbeit mit schriftlicher Ausarbeitung, die individuell durchgeführt wird.</p> <p>Typischerweise umfasst sie auch eine Entwurfs- und Implementierungsleistung des Studierenden, die allein aber nicht ausreichend ist.</p>		
Lehr- und Lernformen	-		
Studien- und Prüfungsleistungen	Benotete Bachelorarbeit und Kolloquium (KO)		
Literatur	abhängig vom Thema		

Modulbezeichnung	<b>WP-Kürzel-Kurztitel (diese Beschreibung gilt für alle drei Wahlpflichtmodule WP I, WP II und WP III)</b>	Kürzel	WP / WPP
Lehrveranstaltung(en)	SeU: WP-Kürzel-Kurztitel Praktikum: WP-Kürzel-Kurztitel	Fach-semester	4-6
Arbeitsaufwand	36 Std. SeU, 36 Std. Praktikum, 108 Std. Eigenarbeit/Selbststudium (bei 2+2)	Dauer	ein Semester
Modulverantwortliche(r)	Anbieter/-in des Moduls	Turnus	nach Angebot und Wahl
Art des Moduls	Wahlpflichtmodul	CP	6
Voraussetzungen	Falls notwendig angeben: „Empfohlen: <empfohlene Module>“	SWS	2+2 oder 3+1
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch oder englisch
Lernziele und Kompetenzen	<p>Je nach Wahlpflichtmodul.</p> <p>Die Beschreibung der aktuellen Angebote ist auf den Informatikseiten zu finden.</p>		
Inhalte	<p>Die Studierenden wählen die Wahlpflichtmodule aus einem Pool, der für die Informatik-Bachelorstudiengänge des Department Informatik gemeinsam angeboten wird. Die Beschreibung des aktuellen Angebots ist auf den Webseiten des Departments Informatik zu finden.</p>		
Lehr- und Lernformen	<p>SeU: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, Beispiele und Demos Praktikum: Bearbeitung von Aufgaben in Zweiergruppen mit abschließendem Abnahmegespräch</p>		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform: benotete Klausur Alternative Prüfungsformen: benotete mündliche Prüfung oder benotetes Referat oder benotete Hausarbeit</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p> <p>Voraussetzung (PVL): Laborabschluss (erfolgreiche Bearbeitung der Aufgaben)</p>		
Literatur	<p>Jeweils in der aktuellen Auflage:</p> <ul style="list-style-type: none"> <li>• ...</li> </ul>		

Modulbezeichnung	<b>Gesellschaftswissenschaften</b>	Kürzel	GW
Lehrveranstaltung(en)	SeU:      GW-Kürzel-Kurztitel	Fach- semester	6
Arbeitsaufwand	36 Std. SeU, 54 Std. Eigenarbeit/Selbststudium	Dauer	ein Semester
Modulverantwortliche(r)	Prof. Dr. Julia Padberg	Turnus	nach Angebot und Wahl
Art des Moduls	Wahlpflichtmodul	CP	3
Voraussetzungen	Falls notwendig angeben: „Empfohlen: <empfohlene Module>“	SWS	2
Verwendbarkeit	Für die Studiengänge „Informatik Technischer Systeme“ und „European Computer Science“	Sprache	deutsch oder englisch
Lernziele und Kompetenzen	Die Studierenden sind in der Lage <ul style="list-style-type: none"> <li>aktuelle Fragestellungen gesellschaftlicher Natur im Umfeld der Informatik debattieren und sich weiterführende Literatur beschaffen und interpretieren</li> <li>die entsprechenden Methoden die das wissenschaftliche Arbeiten, die berufliche Praxis oder die persönliche Entwicklung betreffen, anwenden und erfolgreich nutzen.</li> </ul>		
Inhalte	<ul style="list-style-type: none"> <li>Die Studierenden wählen das gesellschaftswissenschaftliche Fach aus einem Pool, der für die Informatik-Bachelorstudiengänge des Department Informatik gemeinsam angeboten wird. Die gesellschaftswissenschaftlichen Fächer dürfen sich nicht mit den Inhalten der Pflichtmodule überschneiden.</li> <li>Themengebiete mit gesellschaftswissenschaftlicher Relevanz und einem Bezug zur Informatik können Inhalte sein. Darüber hinaus können GW-Fächer Themen, die das wissenschaftliche Arbeiten, die berufliche Praxis oder die persönliche Entwicklung betreffen, beinhalten.</li> <li>Die entsprechenden Methoden und Kompetenzen sollen didaktisch aufbereitet und professionell präsentiert werden.</li> <li>Aktuelle Inhalte können unter folgender Adresse eingesehen werden: <a href="https://www.haw-hamburg.de/fileadmin/TI/FSB/Informatik/">https://www.haw-hamburg.de/fileadmin/TI/FSB/Informatik/</a></li> </ul>		
Lehr- und Lernformen	SeU:      Seminaristischer Unterricht, praktische Übungsaufgaben		
Studien- und Prüfungsleistungen	<p>Regelhafte Prüfungsform:    unbenotete Klausur</p> <p>Alternative Prüfungsformen: nicht benotete mündliche Prüfung oder nicht benotetes Referat oder nicht benotete Hausarbeit oder nicht benoteter Übungsabschluss</p> <p>Bei mehr als einer möglichen Prüfungsform im Modul wird die zu erbringende Prüfungsform von dem verantwortlichen Lehrenden zu Beginn der Lehrveranstaltung bekannt gegeben.</p>		
Literatur	Literatur zum gegebenen Themengebiet; Skripte der Dozent*innen		