Technische Universität München

Modulbeschreibung

IN0001: Einführung in die Informatik 1

Fakultät für Informatik

Modulniveau:	Sprache:	Semesterdauer:	Häufigkeit:
Bachelor	Deutsch	Einsemestrig	Wintersemester

Credits*:	Gesamt-stunden:	Eigenstudiums-stunden:	Präsenz-stunden:
6	180	120	60

* Die Zahl der Credits kann in Einzelfällen studiengangsspezifisch variieren. Es gilt der im Transcript of Records oder Leistungsnachweis ausgewiesene Wert.

Beschreibung der Studien-/Prüfungsleistungen:
Prüfungsart: Klausur (120 Minuten)

Die Prüfungsleistung wird in Form einer Klausur von 120 Minuten erbracht. Wissensfragen überprüfen die Vertrautheit mit Konzepten der Informatik und der Programmierung, kleine Programmieraufgaben überprüfen die Fähigkeit, mit maßgeschneiderten Algorithmen Probleme zu lösen und verteilte Anwendungen zu realisieren.

Wiederholungsmöglichkeit:
Im Folgesemester: Nein
Am Semesterende: Ja

(Empfohlene) Voraussetzungen:
Praktikum: Grundlagen der Programmierung (IN0002) sollte gleichzeitig besucht werden

Inhalt:
In dem Modul IN0001 werden beispielhaft folgende Inhalte behandelt:
- Einführung
++ Grundlegende Begriffe: Problem - Algorithmus - Programm
++ Imperative Programmkonstrukte
- Syntax und Semantik
++ Syntax von Programmiersprachen: reguläre Ausdrücke und kontextfreie Grammatiken
++ Semantik von Programmen: Kontrollfluss-Diagramme
- Grundlegende Datenstrukturen I:
++ Zahlen, Strings, Felder
++ Sortieren durch Einfügen
- Rekursion
++ Binäre Suche
++ Rekursionsarten
- Grundlegende Datenstrukturen II:
++ Objekte, Klassen, Methoden
++ Listen, Keller und Schlangen
- Objektorientierte Programmierung
++ Vererbung
++ abstrakte Klassen und Interfaces

++ Polymorphie
- Programmieren im Großen (Ausblick)
- Nebenläufige Programmierung und Threads

Lernergebnisse:
Nach der erfolgreichen Teilnahme an diesem Modul verstehen die Teilnehmer die wesentlichen Konzepte der Informatik auf einem grundlegenden, praxis-orientierten, aber wissenschaftlichen Niveau.
Konzepte dieser Art sind etwa: Algorithmen, Syntax und Semantik, sowie Effizienz im Hinblick auf Speicherverbrauch oder Zeit.
Die Teilnehmer sind dann in der Lage, in Java oder einer ähnlichen objektorientierten Sprache überschaubare algorithmische Probleme zu lösen und einfache verteilte und nebenläufige Anwendungen zu programmieren. Sie verstehen die diesen Programmiersprachen zugrundeliegenden Konzepte und Modelle und sind deshalb in der Lage, andere zuweisungs- und objektorientierte Programmiersprachen eigenständig zu erlernen.

Lehr- und Lernmethoden:
Vorlesung, kombiniert mit eigenem experimentellen Erarbeiten der Beispiele am Rechner und Erschließen weiterführender Literatur zur Klärung von technischen Detailfragen.

Medienform:
Folienpräsentation, Tafelanschrieb, Online-Programmierung, Animationen, Vorlesungsaufzeichnung

Literatur:
Heinisch, Müller-Hofmann, Goll: Java als erste Programmiersprache, Teubner, 2007
Deitel, Harvey / Deitel, Paul: How to program Java Prentice-Hall, 2002
Flanagan, David: Java in a Nutshell O'Reilly, 2002
Bishop, Judith: Java gently Prentice-Hall, 2001
Eckel, Bruce: Thinking in Java Prentice-Hall, 2002

Modulverantwortliche(r):
Seidl, Helmut; Prof. Dr.: helmut.seidl@tum.de

Lehrveranstaltungen (Lehrform, SWS) Dozent(in):

240952338 Einführung in die Informatik 1 (IN0001) (4SWS VO, WS 2020/21) [BF]
Seidl H, Erhard J, Hagerer G

Weitere Informationen zum Modul und seiner Zuordnung zum Curriculum:
https://campus.tum.de/tumonline/wbModHb.wbShowMHBReadOnly?pKnotenNr=455630

Generiert am: 19.01.2021 13:05

Technische Universität München

Modulbeschreibung

IN0002: Praktikum: Grundlagen der Programmierung

Fakultät für Informatik

Modulniveau:	Sprache:	Semesterdauer:	Häufigkeit:
Bachelor	Deutsch	Einsemestrig	Wintersemester

Credits*:	Gesamt-stunden:	Eigenstudiums-stunden:	Präsenz-stunden:
6	180	120	60

* Die Zahl der Credits kann in Einzelfällen studiengangsspezifisch variieren. Es gilt der im Transcript of Records oder Leistungsnachweis ausgewiesene Wert.

Beschreibung der Studien-/Prüfungsleistungen:
Prüfungsart: Übungsleistung

Auf 7 bis 14 Übungsblättern werden Aufgaben bzw. Programmieraufgaben gestellt, die von den Teilnehmern in schriftlicher oder elektronischer Form gelöst und abgegeben werden. Damit weisen die Teilnehmer nach, dass sie in einer objekt-orientierten Programmiersprache wie Java im Kleinen programmieren können und dass sie grundlegende Konzepte der Informatik verstanden haben und in eigenständigen Lösungen bzw. Programmen anwenden können.
Um den eigenen Anteil an den Lösungen nachweisen zu können, müssen die Teilnehmer dabei jederzeit in der Lage sein, ihre Lösungen auch mündlich zu präsentieren. Vor Beginn des Praktikums wird bekannt gegeben, wie die einzelnen Übungsbestandteile zur Ermittlung der Note gewichtet werden.

Wiederholungsmöglichkeit:
Im Folgesemester: Nein
Am Semesterende: Ja

(Empfohlene) Voraussetzungen:
Einführung in die Informatik 1 (IN0001) sollte gleichzeitig besucht werden

Inhalt:
Begleitend zum Modul IN0001 behandelt das Praktikum
Aufgaben, die u.a. die kreative Verwendung von:
- grundlegenden Datenstrukturen
- Rekursion
- Objekten, Klassen und Methoden
- Listen, Schlangen und Bäumen
- höheren Konzepten der objektorientierten Programmierung
- Nebenläufigkeit

zur Problemlösung einüben.

Lernergebnisse:
Nach erfolreicher Teilnahme an diesem Modul beherrschen die Studierenden die Programmiersprache Java oder eine

ähnliche objekt-orientierte Programmiersprache und das Programmieren im Kleinen. Sie können Programme eigenständig entwickeln und dabei wesentliche Konzepte der Informatik auf einem grundlegenden, praxis-orientierten, aber wissenschaftlichen Niveau anwenden, wie sie im Modul IN0001 gelehrt werden.

Lehr- und Lernmethoden:
Etwa ein Viertel des Moduls besteht aus der Bearbeitung von Übungsaufgaben zum begleitenden Modul IN0001. Diese Übungsaufgaben vertiefen das Verständnis fundamentaler Konzepte der Informatik.
In dem restlichen Teil dieses Moduls entwickeln die Teilnehmer kleinere Beispielanwendungen unter Anleitung, um ihre Fähigkeiten zur Programmierung in einer objektorientierten Programmiersprache zu entwickeln.

Medienform:
Projektor, Folien, Tafel, Softwareentwicklungsumgebungen

Literatur:
Siehe Modul IN0001

Modulverantwortliche(r):
Seidl, Helmut; Prof. Dr.: helmut.seidl@tum.de

Lehrveranstaltungen (Lehrform, SWS) Dozent(in):

0000000261 Praktikum: Grundlagen der Programmierung (IN0002), Do (4SWS PR, WS 2020/21) [BF]
Seidl H [L], Erhard J, Hagerer G

0000005339 Praktikum: Grundlagen der Programmierung (IN0002), Di, Fr (4SWS PR, WS 2020/21) [BF]
Seidl H [L], Erhard J, Hagerer G

2409129840 Praktikum: Grundlagen der Programmierung (IN0002), Mo, Fr (4SWS PR, WS 2020/21) [BF]
Seidl H [L], Erhard J, Hagerer G

240919840 Praktikum: Grundlagen der Programmierung (IN0002), Di, Mi (4SWS PR, WS 2020/21) [BF]
Seidl H [L], Erhard J, Hagerer G

Weitere Informationen zum Modul und seiner Zuordnung zum Curriculum:
https://campus.tum.de/tumonline/wbModHb.wbShowMHBReadOnly?pKnotenNr=452806

Generiert am: 19.01.2021 13:06

Technische Universität München

Modulbeschreibung

IN0004: Einführung in die Rechnerarchitektur

Fakultät für Informatik

Modulniveau:	Sprache:	Semesterdauer:	Häufigkeit:
Bachelor	Deutsch	Einsemestrig	Wintersemester

Credits*:	Gesamt-stunden:	Eigenstudiums-stunden:	Präsenz-stunden:
8	240	150	90

* Die Zahl der Credits kann in Einzelfällen studiengangsspezifisch variieren. Es gilt der im Transcript of Records oder Leistungsnachweis ausgewiesene Wert.

Beschreibung der Studien-/Prüfungsleistungen:
Die Prüfungsleistung wird in Form einer Klausur von 120 Minuten erbracht. Anhand von einfachen Beispielaufgaben zu maschinennaher Assembler Programmierung, zur Mikroprogrammierung und zum Schaltungsentwurf soll die Beherrschung der praktischen Konzepte der Rechnerarchitektur nachgewiesen werden. Weiterhin soll durch Beantwortung von Fragen nachgewiesen werden, dass auch the theoretischen Grundkonzepte der Rechnerarchitektur beherrscht werden. Als Bearbeitungshilfen werden Merkblätter zur Verfügung gestellt, ansonsten sind keine Hilfsmittel erlaubt.

Wiederholungsmöglichkeit:
Im Folgesemester: Nein
Am Semesterende: Ja

(Empfohlene) Voraussetzungen:
keine

Inhalt:
- Funktion und Aufbau von Rechnersystemen aus technischer Sicht: Von-Neumann-Rechner, Maschinenbefehlszyklus, Hardware-Software-Schnittstelle
- Die Instruction Set Architecture (ISA): Funktionsweise und maschinennahes Assembler Programmieren, Aufrufkonventionen
- Implementierung von Maschinenbefehlen durch Mikroprogrammierung
- Schaltungen, Schaltwerke, Schaltwerksentwurf mittels einer formalen Sprache am Beispiel von VHDL
- Einführung in die Rechnerarchitektur: Mikroprozessorarchitekturen und Systemarchitekturen, parallele und verteilte Systeme, Speichersysteme und E/A

Lernergebnisse:
Nach Teilnahme an diesem Modul sind die Studierenden in der Lage, Rechnersysteme als geschichtete, abstrakte Maschinen zu verstehen. Sie haben einen ersten Einblick in das Fachgebiet der Rechnerarchitektur gewonnen und beherrschen die nachfolgend genannten, einzelnen Fähigkeiten:
Sie haben die wesentlichen Konzepte von maschinennaher Programmierung, Mikroprogrammierung und Schaltungsentwurf erlernt und können diese anwenden. Sie haben den Maschinenbefehlszyklus auf Basis der Vorgänge in der Hardware auf Registertransferebene verstanden und sind in der Lage Rechnerarchitekturen zu klassifizieren. Sie haben die Grundsätze moderner Rechnerarchitekturen erlernt.

Lehr- und Lernmethoden:
Mit Hilfe einer Folienpräsentation mit Animationen stellt die Vorlesung die Grundbegriffe und der Methoden Rechnerarchitektur vor und erläutert sie an Beispielen. Eine begleitende Zentralübung sowie begleitende Tutorübungen vertiefen anhand geeigneter Aufgaben das Verständnis der Inhalte der Vorlesung und zeigen die Anwendung der verschiedenen Methoden mit Hilfe von überschaubaren. Hausaufgaben ermöglichen Studierenden die Themen im Selbststudium zu vertiefen. Lösungen zu den Aufgaben werden in der Zentralübung und den Tutorgruppen besprochen. Die Präsentation der eigenen Lösung in der begleitenden Tutorübung verbessert die Kommunikationsfähigkeiten.

Medienform:
Folien von Vorlesung und Zentralübung, schriftliche Übungsblätter, Übungsaufgabensammlung, weitere Arbeitsmaterialien.

Literatur:
- Andrew S. Tanenbaum, Todd Austin: Rechnerarchitektur: Von der digitalen Logik zum Parallelrechner
- David A. Patterson, John L. Hennessy, Computer Organization and Design: The Hardware/Software Interface
- Intel386 TM DX MICROPROCESSOR 32-BIT CHMOS MICROPROCESSOR WITH INTEGRATED MEMORY MANAGEMENT
- Beschreibung der mikroprogrammierbaren Maschine

Modulverantwortliche(r):
Schulz, Martin; Prof. Dr. rer. nat.: martin.w.j.schulz@tum.de

Lehrveranstaltungen (Lehrform, SWS) Dozent(in):

0000000149 Übungen zu Einführung in die Rechnerarchitektur - Gruppen Mo, Di, Mi (IN0004) (2SWS UE, WS 2020/21) [GM]
Schulz M [L], Huseynli F, Schreiber M

0000005120 Übungen zu Einführung in die Rechnerarchitektur - Gruppen Do, Fr (IN0004) (2SWS UE, WS 2020/21) [GM]
Schulz M [L], Huseynli F, Schreiber M

240941859 Einführung in die Rechnerarchitektur (IN0004) (4SWS VO, WS 2020/21) [BF]
Schulz M (Huseynli F, Schreiber M)

Weitere Informationen zum Modul und seiner Zuordnung zum Curriculum:
https://campus.tum.de/tumonline/wbModHb.wbShowMHBReadOnly?pKnotenNr=452812

Generiert am: 19.01.2021 13:07

Technische Universität München

Modulbeschreibung

IN0007: Grundlagen: Algorithmen und Datenstrukturen

Fakultät für Informatik

Modulniveau:	Sprache:	Semesterdauer:	Häufigkeit:
Bachelor	Deutsch	Einsemestrig	Sommersemester

Credits*:	Gesamt-stunden:	Eigenstudiums-stunden:	Präsenz-stunden:
6	180	105	75

* Die Zahl der Credits kann in Einzelfällen studiengangsspezifisch variieren. Es gilt der im Transcript of Records oder Leistungsnachweis ausgewiesene Wert.

Beschreibung der Studien-/Prüfungsleistungen:
Die Prüfungsleistung wird in Form einer 90-minütigen Klausur erbracht. In dieser weisen Studierende anhand der gestellten Aufgaben nach, dass sie über fundamentale Kenntnisse im Bereich der Algorithmen und Datenstrukturen verfügen und diese erfolgreich bei der Lösung von Problemen anwenden können. Ferner demonstrieren Studierende beim Lösen der gestellten Aufgaben, dass sie die im Modul behandelten Datenstrukturen und grundlegenden algorithmischen Methoden beherrschen. Die Studierenden weisen nach, dass sie in begrenzter Zeit grundlegende algorithmische Probleme erkennen und analysieren können sowie Wege zu einer effizienten Lösung finden können.

Wiederholungsmöglichkeit:
Im Folgesemester: Nein
Am Semesterende: Ja

(Empfohlene) Voraussetzungen:
IN0001 Einführung in die Informatik 1, IN0015 Diskrete Strukturen

Inhalt:
Das Modul behandelt zunächst die Grundlagen der Analyse von Effizienz bzw. Komplexität. Es werden grundlegende Begriffe, Komplexitätsmaße, die Landau-Symbole sowie verschiedene Maschinenmodelle eingeführt. Danach studiert das Modul grundlegende Datenstrukturen und algorithmische Probleme.
- Datenstrukturen für Sequenzen: Untersucht werden dynamische Arrays, Listen, Stapel und Warteschlangen. Dabei wird jeweils die Komplexität der einzelnen Operationen hergeleitet.
- Hashing: Im Kern werden Hashing mit Verkettung, universelles Hashing sowie verschiedenen Sondierverfahren vorgestellt. Das Modul behandelt optional perfektes Hashing und hash-basierte Algorithmen, zum Beispiel für das Problem des Mengendurchschnitts.
- Sortieren: Das Modul wiederholt zunächst einfache Verfahren wie InsertionSort, SelectionSort und BubbleSort. Anschließend werden fortgeschrittene Verfahren wie MergeSort, HeapSort und QuickSort analysiert. Optional werden sortierbasierte Algorithmen, die untere Schranke für vergleichsbasiertes Sortieren, Rang-Selektion, RadixSort sowie externes Sortieren vorgestellt.
- Prioritätswarteschlangen: Das Modul untersucht binäre Heaps und Binomialheaps.
- Suchbäume: Das Modul behandelt binäre Suchbäume, AVL-Bäume und (a,b)-Bäume.
- Graphalgorithmen: Das Modul studiert verschiedene Graphrepräsentation, Traversierungstechniken per DFS/BFS, die Berechnung von Zweifachzusammenhangskomponenten und starken Zusammenhangskomponenten,

topologische Sortierung, die Berechnung von kürzesten Wegen und minimalen Spannbäumen. Optional werden Lösungsverfahren für das Traveling Salesman Problem (TSP) vorgestellt.
Im Stoffspektrum des Moduls sind optional Datenkompressionverfahren (Huffman, Lempel-Ziv) und einfache Algorithmen für das Problem des Pattern Matchings vorgesehen.

Lernergebnisse:
Die Teilnehmer beherrschen die oben genannten grundlegende Algorithmen und Datenstrukturen. Sie sind in der Lage, diese eigenständig in ihrer Komplexität zu analysieren und die entsprechenden Analysekonzepte auf verwandte algorithmische Probleme anzuwenden. Ferner sind die Teilnehmer in der Lage, die behandelten Algorithmen und Datenstrukturen einzusetzen, sie ggf. zu modifizieren und verschiedene Lösungen in ihrer Güte zu vergleichen.

Lehr- und Lernmethoden:
Das Modul besteht aus einer Vorlesung und einer begleitenden Übungsveranstaltung. Die Inhalte der Vorlesung werden im Vortrag und durch Präsentation vermittelt. Studierende werden insbesondere durch die Lösung von Übungsblättern zur inhaltlichen Auseinandersetzung mit den Themen angeregt. Die Lösung der Übungsaufgaben wird in der Übungsveranstaltung besprochen. Zusätzlich erhalten die Studierenden durch die Korrektur der Übungsblätter eine individuelle Rückmeldung über ihren Lernerfolg.

Medienform:
Folien, Tafelarbeit, Übungsblätter

Literatur:
Kurt Mehlhorn, Peter Sanders: Algorithms and Data Structures - The Basic Toolbox. Springer, 2008.
Vertiefendes und ergänzendes Material zur Vorlesung findet sich in folgenden Büchern:
- Volker Heun: Grundlegende Algorithmen - Einführung in den Entwurf und die Analyse effizienter Algorithmen. 2. Auflage, Vieweg, 2003.
- Michael T. Goodrich, Roberto Tamassia. Algorithm Design - Foundations, Analysis, and Internet Examples. John Wiley & Sons, 2002.
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. Introduction to Algorithms, 3rd edition, MIT Press, 2009. Deutsche Übersetzung: Algorithmen - Eine Einführung. 3. Auflage, Oldenbourg Verlag, 2010.
- Jon Kleinberg, Eva Tardos. Algorithm Design. Pearson Education, 2005.
- Uwe Schöning. Algorithmik. Spektrum Akademischer Verlag, 2001.
- Robert Sedgewick, Kevin Wayne: Algorithms. 4th edition, Addison-Wesley, 2011.
- Robert Sedgewick. Algorithms in Java, Parts 1-4. 3rd edition, Addison-Wesley, 2002. Deutsche Übersetzung: Algorithmen in Java, Teil 1-4. 3. Auflage, Pearson Education, 2003.

Modulverantwortliche(r):
Albers, Susanne; Prof. Dr. rer. nat.: susanne.albers@tum.de

Lehrveranstaltungen (Lehrform, SWS) Dozent(in):

0000000868 Übungen zu Grundlagen: Algorithmen und Datenstrukturen (IN0007), Mo, Di (2SWS UE, SS 2020/21) [GP]
Lasser T [L], Anneser C, Lasser T, Radke B, Wolf S

0000003907 Übungen zu Grundlagen: Algorithmen und Datenstrukturen (IN0007), Mi, Do, Fr (2SWS UE, SS 2020/21) [GP]
Lasser T [L], Anneser C, Lasser T, Radke B, Wolf S

0000020868 Übungen zu Grundlagen: Algorithmen und Datenstrukturen (IN0007), Mi, Do, Fr (2SWS UE, SS 2020/21) [GP]
Lasser T [L], Anneser C, Lasser T, Radke B, Wolf S

821085727 Grundlagen: Algorithmen und Datenstrukturen (IN0007) (3SWS VO, SS 2020/21) [GP]
Lasser T

Weitere Informationen zum Modul und seiner Zuordnung zum Curriculum:
https://campus.tum.de/tumonline/wbModHb.wbShowMHBReadOnly?pKnotenNr=452818

Generiert am: 19.01.2021 13:08

Technische Universität München

Modulbeschreibung

IN0006: Einführung in die Softwaretechnik

Fakultät für Informatik

Modulniveau:	Sprache:	Semesterdauer:	Häufigkeit:
Bachelor	Deutsch/Englisch	Einsemestrig	Sommersemester

Credits*:	Gesamt-stunden:	Eigenstudiums-stunden:	Präsenz-stunden:
6	180	105	75

* Die Zahl der Credits kann in Einzelfällen studiengangsspezifisch variieren. Es gilt der im Transcript of Records oder Leistungsnachweis ausgewiesene Wert.

Beschreibung der Studien-/Prüfungsleistungen:
Prüfungsart: Klausur
Die Modulprüfung besteht aus einer 90-minütigen Klausur, in der die Studierenden Konzepte und Methoden der verschiedenen Phasen des Software-Engineering erklären und zur Lösung kleiner Probleme anwenden. Des Weiteren wird durch Modellierungsaufgaben die Fähigkeit zur systematischen Analyse und Bewertung fachlicher Anforderungen überprüft.

Wiederholungsmöglichkeit:
Im Folgesemester: Nein
Am Semesterende: Ja

(Empfohlene) Voraussetzungen:
IN0002 Praktikum: Grundlagen der Programmierung

Inhalt:
Software Engineering ist die Etablierung und systematische Anwendung von Prinzipien, Methoden und Werkzeugen für die arbeitsteilige, ingenieurmäßige Entwicklung und Anwendung von umfangreichen, komplexen Softwaresystemen. Es beschäftigt sich mit der Herstellung und Entwicklung von Software, der Organisation und Modellierung von Datenstrukturen und Objekten, und dem Betrieb von Softwaresystemen. Themen der Vorlesung sind damit unter anderem:
- Modellierung mit UML
- Vorgehensmodelle in der Software Entwicklung (linear, iterativ, agil)
- Anforderungsermittlung und -analyse (funktionales Modell, dynamisches Modell und Objektmodell)
- Systementwurf (Spezifikation, Software Architektur, Architekturmuster und Entwurfsziele)
- Objektentwurf und Implementierung (Wiederverwendung, Entwurfsmuster und Schnittstellen Spezifikation)
- Testen (Komponententest, Integrationstest und Systemtest)
- Konfigurationsmanagement, Build Management und Release Management
- Softwarewartung und Evolution
- Projektorganisation und Kommunikation

Lernergebnisse:
Nach der erfolgreichen Teilnahme an dem Modul kennen die Studierenden die Konzepte und Methoden für die verschiedenen Phasen eines Projekts, z.B. Modellierung des Problems, Wiederverwendung von Klassen und Komponenten, und Auslieferung der Software. Sie sind in der Lage für konkrete Probleme die geeigneten Konzepte

und Methoden auszuwählen und anzuwenden.

Die Studierenden kennen die wichtigsten Begriffe und Vorgehensweisen der Softwaretechnik und können gegebene Probleme daraufhin analysieren und bewerten. Darüber hinaus haben sie die Fähigkeit konkrete Problemstellungen in der Softwaretechnik, z.B. mit Hilfe von Entwurfsmustern, lösen.

Lehr- und Lernmethoden:
Mit Hilfe einer Folienpräsentation mit Animationen stellt die interaktive Vorlesung die Grundbegriffe und Methoden des Software Engineerings vor und erläutert sie an Beispielen. Kleine Übungen, z.B. Quiz-, Modellierungs- und Programmieraufgaben, mit individuellem Feedback helfen den Studierenden zu erkennen, ob sie die Grundbegriffe und Methoden verstanden haben.
Begleitende Tutorübungen vertiefen anhand geeigneter Gruppenaufgaben das Verständnis der Inhalte der Vorlesung und zeigen die Anwendung der verschiedenen Methoden mit Hilfe von überschaubaren Problemstellungen in den verschiedenen Phasen des Software Engineerings. Hausaufgaben ermöglichen Studierenden die Themen im Selbststudium zu vertiefen. Die Präsentation der eigenen Lösung in der begleitenden Tutorübung verbessert die Kommunikationsfähigkeiten, die im Software Engineering essentiell sind. Individuelles Feedback zu den Hausaufgaben erlaubt den Studierenden den Lernfortschritt zu messen und ihre Fähigkeiten zu verbessern.

Medienform:
Vortrag mit digitalen Folien, Livestream, Online Übungsaufgaben (Programmierung, Modellierung, Quiz) mit individuellem Feedback, Diskussionsforum und Kommunikationsplattform zum Austausch zwischen Dozenten, Tutoren und Studierenden

Literatur:
B. Bruegge, A. Dutoit: Object-Oriented Software Engineering: Using UML, Design Patterns and Java, 3rd Edition, Pearson Education, 2010
I. Sommerville, Software Engineering, 9th edition, Addison Wesley, 2010

Modulverantwortliche(r):
Matthes, Florian; Prof. Dr. rer. nat.: matthes@tum.de

Lehrveranstaltungen (Lehrform, SWS) Dozent(in):

0000004167 Übungen zu Einführung in die Softwaretechnik (IN0006) 30 Gruppen (2SWS UE, SS 2020/21) [GP]
Krusche S [L], Krusche S, Bernius J

0000004341 Übungen zu Einführung in die Softwaretechnik (IN0006) 25 Gruppen (2SWS UE, SS 2020/21) [GP]
Krusche S [L], Krusche S, Bernius J

0000005028 Übungen zu Einführung in die Softwaretechnik (IN0006) 1 Gruppe (2SWS UE, SS 2020/21) [GP]
Krusche S [L], Krusche S, Bernius J

0000024167 Übungen zu Einführung in die Softwaretechnik (IN0006) 27 Gruppen (2SWS UE, SS 2020/21) [GP]
Krusche S [L], Krusche S, Bernius J

821022409 Einführung in die Softwaretechnik (IN0006) (3SWS VO, SS 2020/21) [GP]
Krusche S [L], Krusche S, Bhatotia P, Bernius J

Weitere Informationen zum Modul und seiner Zuordnung zum Curriculum:
https://campus.tum.de/tumonline/wbModHb.wbShowMHBReadOnly?pKnotenNr=452816

Generiert am: 19.01.2021 13:09

Technische Universität München

Modulbeschreibung

ED0338: Diskrete Mathematik für Berufliche Bildung

Fakultät TUM School of Education

Modulniveau:	Sprache:	Semesterdauer:	Häufigkeit:
Bachelor	Deutsch	Einsemestrig	Wintersemester

Credits*:	Gesamt-stunden:	Eigenstudiums-stunden:	Präsenz-stunden:
4	120	45	75

* Die Zahl der Credits kann in Einzelfällen studiengangsspezifisch variieren. Es gilt der im Transcript of Records oder Leistungsnachweis ausgewiesene Wert.

Beschreibung der Studien-/Prüfungsleistungen:
Die Studierenden werden mittels einer schriftlichen Prüfung von 120 Minuten bewertet. Dabei wird überprüft, ob die Studierenden die mathematischen Begriffe aus Mengentheorie, Relationen, Logik, Graphen und den weiteren, in der Vorlesung behandelten Bereichen korrekt anwenden können.

Wiederholungsmöglichkeit:
Im Folgesemester: Nein
Am Semesterende: Ja

(Empfohlene) Voraussetzungen:
keine

Inhalt:
Die Vorlesung ist eine Einführung in die Begriffe und Bereiche der Diskreten Mathematik für Informatiker. Sie gliedert sich in fünf Teilen:
- Grundbegriffe von Mengen, Relationen, Funktionen und Graphen
- Grundlagen der Aussagenlogik und Logik erster Stufe
- Grundlagen der Kombinatorik
- Algebraische Grundlagen: (Halb-)Gruppen, Ringe und Körper

Lernergebnisse:
Nach erfolgreichem Abschluss des Moduls sind die Teilnehmer in der Lage die Grundbegriffe sowie die Grundlagen von logischen, algebraischen und algorithmischen Kalkülen anzuwenden. Sie können kombinatorische Problemstellungen lösen. Außerdem sind sie in der Lage Probleme mit Methoden der Graphentheorie zu modellieren und zu lösen. Sie können die Effizienz von Lösungsmethoden und Algorithmen bewerten.

Lehr- und Lernmethoden:
In der Vorlesung wechseln sich Vorträge, Präsentationen und Übungsphasen in Einzel- und Gruppenarbeit ab. Das sofortige Einüben des Gelernten an geeigneten Aufgaben wird besonders betont.

Medienform:
Folienpräsentation, Tafelanschrieb, Überungsblätter

Literatur:
A. Steger: Diskrete Strukturen, Band 1: Kombinatorik, Graphentheorie, Algebra, Springer, 2001
M. Aigner: Diskrete Mathematik, Vieweg, 1999 (3rd Edition)
Schöning, Uwe: Logik für Informatiker, Spektrum-Verlag, 2000 (5. Auflage)

Modulverantwortliche(r):
Hubwieser, Peter; Prof. Dr. rer. nat. habil.: peter.hubwieser@tum.de

Lehrveranstaltungen (Lehrform, SWS) Dozent(in):
Vorlesung/Übung:
Diskrete Mathematik für Berufliche Bildung (5 SWS)
N.N.

Weitere Informationen zum Modul und seiner Zuordnung zum Curriculum:
https://campus.tum.de/tumonline/wbModHb.wbShowMHBReadOnly?pKnotenNr=1252993

Generiert am: 19.01.2021 13:12