https://www.iwi.hs-karlsruhe.de/Intranetaccess/info/module/INFB/7/INFB1107

Modul Informatik 1, Bachelor-Studiengang Informatik (SPO 7)

Modulübersicht

EDV-Bezeichnung:
INFB1107

Verantwortliche(r):
Prof. Dr. Ulrich Bröckl

Umfang:
12 ECTS-Punkte / 10 SWS

Einordnung:
1. Semester

Inhaltliche Voraussetzungen: keine

Voraussetzungen nach SPO:
keine

Kompetenzen:

Die Studierenden erlernen die prinzipiellen Beschränkungen heutiger Computer bei der Lösung von wichtigen Problemen. Auf der Basis mathematisch exakter Beweise erfassen sie hierfür wichtige Gebiete der Theoretischen Informatik. Sie klassifizieren formale Sprachen mit Hilfe der sog. Chomsky-Hierarchie und erkennen dadurch ihre algorithmische Komplexität. Weiterhin erfassen die Studierenden die Berechnungskraft gängiger Rechnermodelle durch endliche Automaten und können mit exakten logischen Argumenten deren Grenzen aufzeigen. Diverse Probleme erkennen sie von vorneherein als durch Computer unlösbare Aufgabenstellungen. Die vorgestellten Ergebnisse können die Studierenden durch den sicheren Umgang mit verschiedenen Beweistechniken belegen.

Die Lehrveranstaltungen dieses Moduls vermitteln fachlichen Grundlagen der Softwareentwicklung und der Informatik. Die Studierenden lernen, kleine Probleme zu analysieren und sie mit Hilfe von Programmen zu lösen. Sie wenden bestehende Lösungskonzepte an, um komplexere Probleme in kleinere aufzuteilen. Darüberhinaus lernen Sie existierende Algorithmen zu bewerten und anzuwenden.

Prüfungsleistungen:
Klausur 120 Min. (benotet)

Lehrveranstaltung: Programmieren

EDV-Bezeichnung: INFB1117.a

Art/Modus: Vorlesung

Dozent/in:
Prof. Dr. Ulrich Bröckl

Lehrsprache:
deutsch

Umfang (ECTS): 5

Umfang (SWS): 4

Arbeitsaufwand: 150 Stunden (60 Stunden Präsenz, 90 Stunden eigenständige Arbeit)

Prüfungsleistungen: Modulprüfung

Inhalt:

Die Studierenden werden befähigt, die grundlegenden Java-Programmiersprachenkonstrukte, wie Variablen, Kontrollstrukturen, Methoden, Klassen, Objekte und Felder zum Lösen einfacher Probleme anzuwenden.

Die Hörer der Vorlesung erlernen Programmier- und Dokumentationskonventionen, um Java-Programme lesbar zu schreiben sowie mit Modultests anhand von JUnit zu testen.

Sie eignen sich die Grundelemente der Unified Modeling Language an und modellieren mit objekt-orientierter Analyse und Design kleinere Programme.

Die Studierenden erkennen rekursive Problemstrukturen und lösen Sie mit rekursiven Algorithmen.

Nach Vermittlung typische Such- und Sortierverfahren, werden sie befähigt, Algorithmen hinsichtlich ihres Resourcenverbrauchs zu analysieren und zu vergleichen.

Die Teilnehmer der Vorlesung wenden ihre Kenntnisse anhand von Übungsaufgaben an.

Empfohlene Literatur:

- Tafelmitschrift, Vorlesungsfolien
- Übungsaufgaben mit Lösungen
- Java-Programme und deren Dokumentation als Javadoc
- Weitere Java-Übungsaufgaben mit Lösungen zur Vertiefung.
- Joachim Goll, Cornelia Heinisch, "Java als erste Programmiersprache: Ein professioneller Einstieg in die Objektorientierung mit Java", Springer Vieweg, 7. Auflage, 2014.
- James Gosling, Bill Joy, Guy Steele, Gilad Bracha, Alex Buckley, "The Java Language Specification, Java SE 8 Edition", Oracle America, 8. Auflage, März 2015

Anmerkungen:

Lösen einfacher Aufgaben während der Vorlesung.

Lehrveranstaltung: Programmieren Übung

EDV-Bezeichnung: INFB1127

Art/Modus: Übung

Dozent/in:
Prof. Dr. Ulrich Bröckl

Lehrsprache:
deutsch

Umfang (ECTS): 3

Umfang (SWS): 2

Arbeitsaufwand: 90 Stunden (30 Stunden Präsenz, 60 Stunden eigenständige Arbeit)

Prüfungsleistungen: Übung 1 Semester (nicht benotet)

Inhalt:

Mit wöchentlichen Übungsaufgaben vertiefen die Studierenden am Rechner die praktischen Inhalte der Vorlesung.

Sie benutzen eine integrierte Java-Entwicklungsumgebung, um damit Programme zu erstellen, zu testen und zu ändern. In den darauf folgenden Übungen programmieren die Studierenden einfache Berechnungen mit Java unter Verwendung von Variablen, Ausdrücke und Kontrollstrukturen. Später entwickeln die Studenten einfache objekt-orientierte Programme am Rechner. Am Ende lösen Sie rekursive Probleme und implementieren teilweise aus der Vorlesung bekannte Such- und Sortierverfahren.

Zusätzlich lernen die Studierenden anhand einer umfassenden Programmieraufgabe gesamthaft, ein Programm zu entwerfen, zu implementieren und zu testen. Diese Aufgabe ist von der Komplexität her ein kleines textbasiertes Spiel, wie Tic-Tac-Toe, das zwei Personen gegeneinander am Bildschirm spielen können.

Empfohlene Literatur:

- Übungsaufgaben
- Programme mit Lösungen
- Online-Dokumentation im der Lernplattform ILIAS

Anmerkungen:

Praktische Übungen
Lehrveranstaltung: Theoretische Informatik 1

EDV-Bezeichnung: INFB1117.b
Art/Modus: Vorlesung
Dozent/in:
Prof. Dr. Heiko Körner
Lehrsprache:
deutsch
Umfang (ECTS): 4
Umfang (SWS): 4
Arbeitsaufwand: 120 Stunden (60 Stunden Präsenz, 60 Stunden eigenständige Arbeit)
Prüfungsleistungen: Modulprüfung

Inhalt:

Die Lehrveranstaltung führt in die Theorie der formalen Sprachen ein. Das Ziel ist die Vermittlung der Chomsky-Hierarchie als ein Stufenmodell unterschiedlich komplexer Sprachen. Weiterhin werden endliche Automaten als Repräsentanten heutiger Computer vorgestellt und ihre Beschränkungen aufgezeigt. Ein weiteres Lernziel ist die sichere Anwendung verschiedener Beweistechniken.

Die Lehrveranstaltung umfasst unter anderem die folgenden Gebiete der theoretischen Informatik: Aussagenlogik, formale Sprachen, Beweistechniken, das O-Kalkül, endliche Automaten, reguläre Sprachen und Ausdrücke, die Chomsky-Hierarchie, das Pumping-Lemma für reguläre und kontextfreie Sprachen sowie die Minimierung endlicher Automaten nach dem Satz von Myhill-Nerode. Weiterhin werden Kellerautomaten, der CYK-Algorithmus sowie Abgeschlossenheitseigenschaften von kontextfreien Sprachen besprochen.

Empfohlene Literatur:

- Tafelanschrieb

- Skript
- Musterlösungen für alle Übungsaufgaben
- D. W. Hoffmann: Theoretische Informatik, 3. Auflage. Hanser, 2015.
- D. Sipser: Introduction to the Theory of Computation, 3rd edition. Cengage Learning, Inc., 2012.

Anmerkungen:

Die Lehrveranstaltung findet als reine Vorlesung statt. Zahlreiche Übungsaufgaben vertiefen die vermittelten Gebiete und werden in evtl. zusätzlich angebotenen Tutorien diskutiert.

https://www.iwi.hs-karlsruhe.de/Intranetaccess/info/module/INFB/7/INFB2107

Modulübersicht

EDV-Bezeichnung:
INFB2107

Verantwortliche(r):
Prof. Dr. Christian Pape

Umfang:
9 ECTS-Punkte / 8 SWS

Einordnung:
2. Semester

Inhaltliche Voraussetzungen: Informatik 1

Voraussetzungen nach SPO:
keine

Kompetenzen:

Die Studierenden lernen objekt-orientierte Techniken kennen, um Software-Entwicklungs-Projekte eigenständig zu implementieren. Sie nutzen weitergehende Analyse-, Design- und Realisierungskompetenzen, um das Zusammenspiel und die architektonische Organisation vieler Klassen in einem Projekt anhand der Programmiersprache Java sowie der UML zu formulieren. Weiterhin können sie abschätzen, in welcher Situation bestimmte komplexe Datentypen eingesetzt werden, wie diese funktionieren und welchen Laufzeitaufwand sie besitzen. Die Studierenden erkennen, dass das erlernte Wissen erforderlich ist, um einfache mobile Apps zu schreiben. In der Übung wenden Sie Ihre erlangten Kenntnisse anhand verschiedener Aufgaben an.

Prüfungsleistungen:
Einzelprüfungen

Lehrveranstaltung: Algorithmen und Datenstrukturen

EDV-Bezeichnung: INFB2117

Art/Modus: Vorlesung

Dozent/in:
Prof. Dr. Christian Pape

Lehrsprache:
deutsch

Umfang (ECTS): 4

Umfang (SWS): 4

Arbeitsaufwand: 120 Stunden (60 Stunden Präsenz, 60 Stunden eigenständige Arbeit)

Prüfungsleistungen: Klausur 120 Min. (benotet)

Inhalt:

Die Vorlesung gliedert sich in mehrere Teile, die inhaltlich aufeinander aufbauen:

1. Im wichtigsten und umfangreichsten ersten Teil erwerben die Studierenden Grundbegriffe und Denkweisen der objekt-orientierten Programmierung anhand der Programmiersprache Java. Dazu gehören: Sprachelemente von Java, Datenabstraktion und Kapselung, Vererbung, Polymorphie, generische Programmierung, Fehlerbehandlung und Laufzeit-Typinformationen.
2. Darauf aufbauend werden die Modellierung von Klassendiagrammen mittels UML vermittelt und der Zusammenhang zwischen der grafischen Beschreibung und deren Implementierung in Java gezeigt.

3. Im dritten Teil wenden die Teilnehmer das erworbene Wissen an, um einfache mobile Anwendungen mit grafischen Oberflächen für Android zu erstellen. Sie sehen, wie dort objekt-orientierte Techniken eingesetzt werden.
4. Der vierte Teil der Vorlesung konzentriert sich auf die Funktionsweisen wichtiger Datenstrukturen wie Listen, Hashtabellen, Bäume und Graphen sowie grundlegende Algorithmen auf Basis der Datenstrukturen. Die Studierenden lernen nicht nur, wie die Datenstrukturen aufgebaut sind, sie sollen auch anhand des Laufzeitverhaltens Datenstrukturen für bestimmten Aufgaben auswählen können.
5. Im abschließende fünften Teil beschäftigt sich die Vorlesung mit der Modularisierung von Anwendungen mit Hilfe von Spring.

Empfohlene Literatur:

- PowerPoint-Präsentationen
- Programmbeispiele
- Skript
- Christian Ullenboom, Java ist auch eine Insel, Galileo Computing
- R. C. Martin, Clean Code, mitp
- B. Lahres, G. Raýman, Objektorientierte Programmierung, Galileo Computing
- G. Popp, Konfigurationsmanagement mit Subversion, Maven und Redmine, dpunkt
- M. Jeckle, C. Rupp, J. Hahn, B. Zengler, S. Queins, UML 2 - glasklar, Hanser-Verlag
- G. Saake, K. Sattler, Datenstrukturen und Algorithmen: Eine Einführung mit Java, dpunkt
- T. Künneth, Android 5: Apps entwickeln mit Android Studio, Rheinwerk Computing

Anmerkungen:

Vor- und Nacharbeit der Vorlesungsinhalte, Klausurvorbereitung
Lehrveranstaltung: Algorithmen und Datenstrukturen Übung

EDV-Bezeichnung: INFB2137

Art/Modus: Übung

Dozenten:
Prof. Dr. Christian Pape
Dr. Martin Holzer

Lehrsprache:
deutsch

Umfang (ECTS): 2

Umfang (SWS): 2

Arbeitsaufwand: 60 Stunden (30 Stunden Präsenz, 30 Stunden eigenständige Arbeit);

Prüfungsleistungen: Übung 1 Semester (nicht benotet)

Inhalt:

Die Studierenden vertiefen das in der Vorlesung erworbene Wissen, indem sie Übungsaufgaben in Java lösen und kleinere Problemstellungen in UML mit Klassendiagrammen modellieren. Dazu verwenden sie jeweils Standard-Entwicklungsumgebungen.

Empfohlene Literatur:

- Skript
- Übungsaufgaben
- Musterlösungen (außer für die Pflichtaufgaben)

Anmerkungen:
Lehrveranstaltung: Theoretische Informatik 2

EDV-Bezeichnung: INFB2127
Art/Modus: Vorlesung
Dozent/in:
Prof. Dr. Heiko Körner
Lehrsprache:
deutsch
Umfang (ECTS): 3
Umfang (SWS): 2
Arbeitsaufwand: 90 Stunden (30 Stunden Präsenz, 60 Stunden eigenständige Arbeit);
Prüfungsleistungen: Klausur 60 Min. (benotet)
Inhalt:

Kern dieser Vorlesung ist die Vermittlung der Grenzen von heutigen Computern, die selbst bei unendlich viel vorhandenem Speicherplatz auftreten. Themen sind vor allem die Berechen- und Unentscheidbarkeit diverser Probleme. Ebenso wird eine Einführung in die Theorie hartnäckiger Probleme gegeben.

Die Lehrveranstaltung umfasst unter anderen die folgenden Gebiete der theoretischen Informatik: Elementare Berechnungsmodelle wie Turingmaschinen und WHILE-Programme, die Church-Turing-These, Unentscheidbarkeit, die Theorie der NP-Vollständigkeit und Zero-Knowledge-Beweise.

Für diese Lehrveranstaltung sind elementare Vorkenntnisse zur theoretischen Informatik notwendig (regulären Sprachen, endliche Automaten, O-Kalkül, usw.). Diese Kenntnisse können z.B. in der Vorlesung Theoretische Informatik I erworben werden.

Empfohlene Literatur:

- Tafelanschrieb
- Skript
- Zu allen Übungsaufgaben werden Musterlösungen angeboten.
- D. W. Hoffmann: Theoretische Informatik, 3. Auflage. Hanser, 2015.
- M. Sipser: Introduction to the Theory of Computation, 3rd edition. Cengage Learning, Inc., 2012.

Anmerkungen:

Die Lehrveranstaltung findet als reine Vorlesung statt. Zahlreiche Übungsaufgaben vertiefen die vermittelten Gebiete und werden in evtl. zusätzlich angebotenen Tutorien diskutiert.

https://www.iwi.hs-karlsruhe.de/Intranetaccess/info/module/INFB/7/INFB2207

Modulübersicht

EDV-Bezeichnung:
INFB2207

Verantwortliche(r):
Prof. Dr. Martin Sulzmann

Umfang:
5 ECTS-Punkte / 4 SWS

Einordnung:
2. Semester

Inhaltliche Voraussetzungen: Informatik 1

Voraussetzungen nach SPO:
keine

Kompetenzen:

Die Studierenden bekommen einen Einblick in die Programmiersprachen C/C++. Die Studierenden sind in der Lage die verschiedenen Sprachmerkmale (imperative, objekt-orientiert) einzuordnen und je nach Anwendungsfall geeignet einzusetzen. Anhand einer Reihe von Übungsaufgaben wird das erlernte Wissen praktisch erprobt.

Prüfungsleistungen:
Einzelprüfungen

Lehrveranstaltung: Softwareprojekt

EDV-Bezeichnung: INFB2217

Art/Modus: Vorlesung

Dozent/in:
Prof. Dr. Martin Sulzmann

Lehrsprache:
deutsch

Umfang (ECTS): 3

Umfang (SWS): 2

Arbeitsaufwand: 150 Stunden (60 Stunden Präsenz, 90 Stunden eigenständige Arbeit)

Prüfungsleistungen: Klausur 90 Min. (benotet)

Inhalt:

Gegenstand der Vorlesung ist die Einführung in die Programmiersprachen C/C++. Folgende Themen werden betrachtet.

1. Systemnahe Programmierung in C mit Hilfe von Bitoperationen und Speichermanipulation via Zeigern.
2. Manuelle Speicherverwaltung in C
3. Objekt-orientierte Programmierung in C++ mit Vergleich zu Java.
4. Komplexere Programmieraufgabe unter Ausnutzung der STL.

Empfohlene Literatur:

- Projektbeschreibung mit genauer Anleitung
- Skript zu C/C++ und der benötigten API
- zusätzliche Übungsaufgaben mit Musterlösungen
- Ulrich Breymann, C++ - Einführung und professionelle Programmierung, Hanser-Verlag

Anmerkungen:
Lehrveranstaltung: Softwareprojekt Übung

EDV-Bezeichnung: INFB2227

Art/Modus: Übung

Dozenten:
Prof. Dr. Martin Sulzmann
Dipl. Inf. (FH) Oktavian Gniot
Prof. Dr. Heiko Körner

Lehrsprache:
deutsch

Umfang (ECTS): 2

Umfang (SWS): 2

Arbeitsaufwand: 60 Stunden (30 Stunden Präsenz, 30 Stunden eigenständige Arbeit)

Prüfungsleistungen: Übung 1 Semester (nicht benotet)

Inhalt:

Diese Übung ergänzt die Vorlesung und ermöglicht den Studierenden, das theoretisch erlangte Wissen in kleinen Aufgaben zu C und C++ anzuwenden.

Empfohlene Literatur:

- Übungsaufgaben auf der Lehrplattform Ilias

Anmerkungen: