

| Modul <b>FMI-IN0186</b> Advanced Functional Programming           |   |
|---|---|
| Modulcode   | FMI-IN0186  |
| Modultitel (deutsch)  | Advanced Functional Programming   |
| Modultitel (englisch)   | Advanced Functional Programming   |
| Modul-Verantwortliche/r   | Clemens Grelck  |
| Voraussetzung für die Zulassung zum Modul                         | keine   |
| Empfohlene bzw. erwartete Vorkenntnisse                           | Kenntnisse der Informatik im Umfang der ersten beiden Studiensemester   |
| Art des Moduls (Pflicht-, Wahlpflicht- oder Wahlmodul)            | <ul style="list-style-type: none"> <li>- 079 B.Sc. Informatik: Wahlpflichtmodul (SWS; Mathematik/Informatik)</li> <li>- 079 Lehramt Gymnasium Informatik: Wahlpflichtmodul (SWS)</li> <li>- 079 Lehramt Regelschule Informatik: Wahlpflichtmodul (SWS)</li> <li>- 079 B.A. Informatik (EF): Wahlpflichtmodul</li> <li>- 221 B.Sc. Bioinformatik: Wahlpflichtmodul (Informatik)</li> <li>- 679 B.Sc. Angewandte Informatik: Wahlpflichtmodul (SWS)</li> </ul>  |
| Häufigkeit des Angebots (Modulturnus)                             | Jedes 2. Semester (ab Sommersemester)   |
| Dauer des Moduls  | 1 Semester  |
| Zusammensetzung des Moduls / Lehrformen (V, Ü, S, Praktikum, ...) | 2 SWS Vorlesung<br>2 SWS Übung  |
| Leistungspunkte (ECTS credits)                                    | 6 LP  |
| Arbeitsaufwand (work load) in:                                    | 180 h   |
| - Präsenzstunden  | 60 h  |
| - Selbststudium (einschl. Prüfungsvorbereitungen)                 | 120 h   |
| Inhalte   | <p>Das Paradigma der Funktionalen Programmierung ist *der* andere Ansatz zur Programmierung von Rechensystemen im Gegensatz zur imperativen und ggf. objektorientierten Programmierung. Dabei wird der funktionalen Programmierung im allgemeinen eine höhere Programmierproduktivität zugeschrieben, und sie eignet sich aufgrund der konzeptuellen Abwesenheit von Zuständen grundsätzlich auch besser für die Programmierung paralleler Systeme. Auf Basis allgemeiner Grundkenntnisse von Informatik und Programmierung vermittelt das Modul die theoretischen Grundlagen und praktische Kenntnisse moderner getypter funktionaler Programmierung.</p> <p>Das funktionale Paradigma hat eine Vielzahl von Programmiersprachen hervorgebracht. Das Modul konzentriert sich jedoch auf zwei funktionale Programmiersprachen: die „strikte“ Sprache OCaml und die „lazy“ Sprache Haskell. Bei ihnen handelt es sich um die prominentesten und am weitesten verbreiteten Vertreter der beiden Schulen funktionaler Programmierung, strikt und lazy.</p> |

|   |  |
|---|--|
|   | <p>Die eher praktischen Teile der Veranstaltung werden durch eine solide Einführung in die darunter liegende Theorie, insbesondere den Lambda-Kalkül, ergänzt. Dabei zeigen wir den Zusammenhang zwischen theoretischen Modellen der Berechenbarkeit und dem praktischen Entwurf von Programmiersprachen auf und diskutieren die grundlegenden Zusammenhänge zwischen Turing-Maschine und imperativ/objekt-orientierter Programmierung einerseits und Lambda-Kalkül und funktionaler Programmierung andererseits. Gleichzeitig lernen wir, worin die fundamentalen Unterschiede zwischen strikter und lazy Ausführung funktionaler Programme liegen und welche Auswirkungen die Wahl auf die Semantik hat.</p> <p>Neben den theoretischen Kenntnissen und praktischen Fertigkeiten der funktionalen Programmierung vermittelt das Modul grundlegende Einsichten in Programmierkonzepte und die Kunst des Programmierens, welche auch bei der Verwendung imperativer und objektorientierter Programmiermodelle hilfreich sind. Dies gilt umso mehr, da zahlreiche klassische Merkmale funktionaler Programmierung zunehmend auch von modernen Versionen imperativer Programmiersprachen unterstützt werden.</p> |
| Lern- und Qualifikationsziele                                     | <ul style="list-style-type: none"> <li>• Die Studierenden sammeln umfassende fortgeschrittene Kenntnisse und Fähigkeiten auf dem Gebiet der getypten funktionalen Programmierung.</li> <li>• Die Studierenden beherrschen die Grundlagen der funktionalen Programmiersprachen OCaml und Haskell sowie des Lambda-Kalküls als gemeinsamer theoretischer Grundlage.</li> <li>• Die Studierenden kennen die Unterschiede zwischen imperativer und funktionaler Programmierung sowie die Unterschiede und Gemeinsamkeiten zwischen strikter und lazy funktionaler Programmierung sowohl aus theoretischer als auch aus praktischer Perspektive.</li> <li>• Die Studierenden trainieren ihr konzeptionelles, analytisches und logisches Denken.</li> </ul>  |
| Voraussetzung für die Zulassung zur Modulprüfung                  | Erfolgreiche Bearbeitung von Übungsaufgaben  |
| Voraussetzung für die Vergabe von Leistungspunkten (Prüfungsform) | Mündliche Prüfung (100%) oder schriftliche Prüfung (100%)<br>Die Festlegung erfolgt zu Beginn des Moduls   |
| Zusätzliche Informationen zum Modul                               |  |
| Unterrichtssprache  | Deutsch, bei Bedarf Englisch<br>Sämtliche Materialien sind in englischer Sprache verfasst.   |