Modulbezeichnung:	**Theorie der Programmierung (ThProg)** (Theory of Programming)	**7.5 ECTS**
Modulverantwortliche/r:	Lutz Schröder	
Lehrende:	Tadeusz Litak, Lutz Schröder, Daniel Gorin	

Startsemester: SS 2014	Dauer: 1 Semester	Turnus: jährlich (SS)
Präsenzzeit: 90 Std.	Eigenstudium: 135 Std.	Sprache: Deutsch

Lehrveranstaltungen:

Theorie der Programmierung (SS 2014, Vorlesung, 4 SWS, Lutz Schröder)
Übungen zu Theorie der Programmierung (SS 2014, Übung, 2 SWS, Daniel Gorin)
Intensivübung zu Theorie der Programmierung (SS 2014, optional, Übung, 2 SWS, Daniel Gorin)

Inhalt:

- Termersetzungssysteme, Normalisierung, Konfluenz
- Getypter und ungetypter Lambda-Kalkül
- Semantik von Programmiersprachen, Anfänge der Bereichstheorie
- Datentypen, Kodatentypen, Induktion und Koinduktion, Rekursionund Korekursion
- Programmverifikation, Floyd-Hoare-Kalkül
- Reguläre Sprachen und endliche Automaten
- Beschriftete Transitionssysteme, Bisimulation und Temporallogik

Lernziele und Kompetenzen:

Fachkompetenz

Wissen

Die Studierenden geben elementare Definitionen und Fakten zu den behandelten Formalismen wieder.

Verstehen

Die Studierenden

- erläutern Grundbegriffe der Syntax und Semantik von Formalismen und setzen diese zueinander in Bezug
- beschreiben und erklären grundlegende Algorithmen zu logischem Schließen und Normalisierung
- beschreiben wichtige Konstruktionen von Modellen, Automaten und Sprachen

Anwenden

Die Studierenden

- verfassen formale Spezifikationen sequentieller und nebenläufiger Programme
- verifizieren einfache Programme gegenüber ihrer Spezifikation durchAnwendung der relevanten Kalküle
- setzen formale Sprachen mit entsprechenden Automaten inBeziehung
- führen einfache Beweise über Programme mittels Induktion und Koinduktion

Analysieren

Die Studierenden

- wählen für gegebene Verifikationsprobleme geeignete Formalismen aus
- erstellen einfache Meta-Analysen formaler Systeme, etwa Konfluenzprüfung von Termersetzungssystemen
- führen einfache Meta-Beweise über Formalismen mittels Induktion und Koinduktion

Lern- bzw. Methodenkompetenz

Die Studierenden beherrschen das grundsätzliche Konzept des Beweises als hauptsächliche Methode des Erkenntnisgewinns in der theoretischen Informatik. Sie überblicken abstrakte Begriffsarchitekturen.

Sozialkompetenz

Die Studierenden lösen abstrakte Probleme in kollaborativer Gruppenarbeit.

Literatur:

- Glynn Winskel, Formal Semantics of Programming Languages, MIT Press, 1993

- Michael Huth, Mark Ryan, Logic in Computer Science, Cambridge University Press, 2. Auflage 2004
- Henk Barendregt, The lambda-Calculus: Its Syntax and Semantics, North Holland, 1984
- John E. Hopcroft, Jeffrey D. Ullman and Rajeev Motwani, Introduction to Automata Theory, Languages, and Computation, 3rd ed., Prentice Hall, 2006
- Franz Baader, Tobias Nipkow, Term Rewriting and All That, Cambridge University Press, 1999

Verwendbarkeit des Moduls / Einpassung in den Musterstudienplan:

Das Modul ist im Kontext der folgenden Studienfächer/Vertiefungsrichtungen verwendbar:

[1] **Informatik (Bachelor of Science): 4. Semester**

(Po-Vers. 2009w | weitere Pflichtmodule | Theorie der Programmierung)

Dieses Modul ist daneben auch in den Studienfächern "Mathematik (Bachelor of Science)" verwendbar.

Studien-/Prüfungsleistungen:

Theorie des Programmierens (Klausur) (Prüfungsnummer: 31211)

Prüfungsleistung, Klausur, Dauer (in Minuten): 90

Anteil an der Berechnung der Modulnote: 100%

weitere Erläuterungen:

Die Rahmen der Übungen gestellten Übungsaufgaben können abgegeben werden und werden in diesem Fall bewertet. Auf Basis des Ergebnisses dieser Bewertungen können bis zu 15% Bonuspunkte erworben werden, die zu dem Ergebnis einer bestandenen Klausur hinzugerechnet werden.

Erstablegung: SS 2014, 1. Wdh.: WS 2014/2015

1. Prüfer: Lutz Schröder
