

Praktische Informatik 3: Funktionale Programmierung								Modulnummer:	
<i>Practical Computer Science 3</i>									
Bachelor Pflicht/Wahl <input checked="" type="checkbox"/> Wahlpflicht <input type="checkbox"/> Wahl <input type="checkbox"/> Sonderfall <input type="checkbox"/>				Modulbereich: Pflicht					
Anzahl der SWS	V	UE	K	S	Prak.	Proj.	Σ	Kreditpunkte: 6	Turnus angeboten in jedem WiSe
	2	2	0	0	0	0	4		
Formale Voraussetzungen: -									
Inhaltliche Voraussetzungen: Praktische Informatik 2									
Vorgesehenes Semester: ab 1. Semester									
Sprache: Deutsch									
Ziele: <ul style="list-style-type: none"> • Konzepte und typische Merkmale des funktionalen Programmierens kennen, verstehen und anwenden können. • Datenstrukturen und Algorithmen in einer funktionalen Programmiersprache umsetzen und auf einfachere praktische Probleme anwenden können. • In Gruppen Probleme analysieren und gemeinsam Lösungsstrategien entwickeln und präsentieren können. Die Vorlesung Praktische Informatik 3 vermittelt essenzielles Grundwissen und Basisfähigkeiten, deren Beherrschung für nahezu jede vertiefte Beschäftigung mit Informatik Voraussetzung ist.									
Inhalte: <ol style="list-style-type: none"> 1. Grundlagen der funktionalen Programmierung: Rekursion – Definition von Funktionen durch rekursive Gleichungen und Mustervergleich (pattern matching) – Auswertung, Reduktion, Normalform – Funktionen höherer Ordnung, currying, Typkorrektheit und Typinferenz 2. Typen: Algebraische Datentypen – Typkonstruktoren – Typklassen – Polymorphie – Standarddatentypen (Listen, kartesische Produkte, Lifting) und Standardfunktionen darauf (fold, map, filter) – Listenkomprehension 3. Algorithmen und Datenstrukturen: Unendliche Listen (Ströme) – Bäume – Graphen – zyklische Datenstrukturen 4. Strukturierung und Spezifikation: Module – Schnittstellen (Interfaces) – Abstrakte Datentypen – Signaturen und Axiome 5. Theoretische Aspekte: Referentielle Transparenz – Lambda-Kalkül – Beweis durch Induktion 6. Fortgeschrittene Funktionale Programmierung: Funktionale I/O und zustandsbasierte Programme – Monaden Im Übungsbetrieb; Programmentwicklung in Haskell — Realisierung einzelner, überschaubarer Programmieraufgaben in kleinen Gruppen									
Unterlagen (Skripte, Literatur, Programme usw.): <ul style="list-style-type: none"> • Simon Thompson: Haskell - The Craft of Functional Programming, Addison-Wesley, 3. Auflage 2011. • Peter Pepper: Funktionale Programmierung. Springer-Verlag 1999. Weiteres Lehrmaterial ist auf der Webseite des Veranstaltung zu finden: <ul style="list-style-type: none"> • Folienkopien • Übungsaufgaben • Hinweise auf Quellen im WWW Das Haskell-System ghci ist frei verfügbare Software (für Linux, Windows und MacOS).									
Form der Prüfung: i. d. R. Bearbeitung von Übungsaufgaben und Klausur									
Arbeitsaufwand		Präsenz		56 h		Übungsbetrieb/Prüfungsvorbereitung		124 h	
		Summe		180 h					

Lehrende:
Dr. B. Hoffmann, Prof. Dr. C. Lüth

Verantwortlich:
Dr. B. Hoffmann