HAW Kiel
Hochschule für Angewandte Wissenschaften Kiel
Kiel University of Applied Sciences

BI141 - Grundlagen funktionaler Programmierung
BI141 - Introduction to functional programming

Allgemeine Informationen	
Modulkürzel oder Nummer	BI141
Eindeutige Bezeichnung	GrundlFunktP-01-BA-M
Modulverantwortlich(e)	Oenings, Hendrik (hendrik.oenings@haw-kiel.de) Angres, Julius (julius.angres@haw-kiel.de)
Lehrperson(en)	Angres, Julius (julius.angres@haw-kiel.de)
Wird angeboten zum	Wintersemester 2024/25
Moduldauer	1 Fachsemester
Angebotsfrequenz	Regelmäßig
Angebotsturnus	In der Regel im Wintersemester
Lehrsprache	Deutsch
Empfohlen für internationale Studierende	Nein
Ist als Wahlmodul auch für andere Studiengänge freigegeben (ggf. Interdisziplinäres Modulangebot - IDL)	Ja

Studiengänge und Art des Moduls (gemäß Prüfungsordnung)
Studiengang: B.Eng. - E - Elektrotechnik (PO 2017, V3) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - E - Elektrotechnik (PO 2023, V4) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - Ming - Medieningenieur/-in (PO 2018, V1 + PO 2021, V2) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - Wing - Wirtschaftsingenieurwesen - Elektrotechnik (PO 2017, V1) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Eng. - Wing - Wirtschaftsingenieurwesen - Elektrotechnik (PO 2023, V2) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Sc. - INF - Informatik (PO 2021,V1) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Sc. - INI - Informationstechnologie (PO 2017, V1) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Sc. - WINF - Wirtschaftsinformatik (6 Sem.) Modulart: Wahlmodul Fachsemester: 5
Studiengang: B.Sc. - WINF 7 Sem. - Wirtschaftsinformatik (7 Sem.) Modulart: Wahlmodul Fachsemester: 5

Kompetenzen / Lernergebnisse
Kompetenzbereiche: Wissen und Verstehen; Einsatz, Anwendung und Erzeugung von Wissen; Kommunikation und Kooperation; Wissenschaftliches Selbstverständnis/Professionalität.
Die Studierenden - kennen die Unterschiede zwischen imperativer und deklarativer Programmierung - kennen die Vorteile deklarativer Programmierung gegenüber imperativer Programmierung - können grundlegende Programme in einer funktionalen Programmiersprache erstellen - wissen, wie klassische imperative Programmierkonzepte (Datenstrukturen, Schleifen) in funktionalen bzw. funktionallogischen Programmiersprachen abgebildet werden können - können Ein- und Ausgabeoperationen umsetzen und verstehen, wie der Verzicht auf Seiteneffekte funktioniert - verstehen das Konzept der "Lazy Evaluation" und können Algorithmen unter Ausnutzung dieses Konzepts kompakt angeben - kennen die Grundlagen logischer Programmierung und das Konzept des Nichtdeterminismus in der Programmierung
Die Studierenden - können in Anwendungen der multiparadigmatischen Programmiersprache Python Elemente der funktionalen Programmierung erkennen und zielgerichtet zur Verbesserung der Codequalität einsetzen - können Algorithmen unabhängig von der konkreten Implementierung und Auswertung verstehen und beschreiben
Die Studierenden - lernen, Aufgaben in kleinen Gruppen zu bearbeiten - können sich über Programme, Konzepte und Lösungen austauschen und diese vor einer Gruppe präsentieren - können Programme übersichtlich und kompakt darstellen und dokumentieren
Die Studierenden - können Programmiersprachen mit funktionalen Elementen erlernen und anwenden - können Programmiersprachen mit logischen Elementen erlernen und anwenden - sind in der Lage, Aufgabenstellungen sowohl selbstständig als auch im Team umzusetzen

Angaben zum Inhalt	
Lehrinhalte	Abgrenzung Programmierparadigmen: deklarativ vs. imperativ; Funktionale Programmierung; Motivation Vergleich zwischen "strict evaluation" und "lazy evaluation"; Vorteile von "lazy evaluation" Vergleich zwischen imperativen und funktionalen Umsetzungen bekannter Algorithmen (z.B. Quicksort) Programmiersprache(n) Haskell / Curry: - Geschichte, Verbreitung, Compiler/Laufzeitumgebungen (GHC, GHCi) - Funktionen (Definition, Rekursive Funktionen, Funktionen höherer Ordnung, Lambdaausdrücke, Komposition) - Datentypen (vordefinierte Typen, Deklaration, Polymorphie, Typklassen) - Datenstrukturen (Listen, Bäume, unendliche Datenstrukturen) - Ein-/Ausgabe und Behandlung von Seiteneffekten - Nichtdeterministische Funktionen - Logische Constraints Programmiersprache Python: - Nutzung der Funktionen map, filter, reduce - Nutzung von Lambdaausdrücken und Funktionen höherer Ordnung - Vermeidung von Seiteneffekten, Wiederverwendbarkeit
Literatur	wird in der Veranstaltung bekanntgegeben

Lehrformen der Lehrveranstaltungen	
Lehrform	**SWS**
Übung	2
Lehrvortrag	2

Arbeitsaufwand	
Anzahl der SWS	4 SWS
Leistungspunkte	5,00 Leistungspunkte
Präsenzzeit	48 Stunden
Selbststudium	102 Stunden

Modulprüfungsleistung	
Voraussetzung für die Teilnahme an der Prüfung gemäß PO	Keine
BI141 - Klausur	Prüfungsform: Klausur Dauer: 60 Minuten Gewichtung: 100% wird angerechnet gem. § 11 Absatz 2 PVO: Nein Benotet: Ja

Sonstiges	
Empfohlene Voraussetzungen	- Grundlegendes Verständnis imperativer Programmiersprachen - Grundkenntnisse der informatischen Logik - Verständnis von Algorithmen und Rekursion - Verständnis von Datenstrukturen (Bäume, verkettete Listen) - Grundkenntnisse in der Programmiersprache Python