Modulcode (1.)	Modulbezeichnung (2.)		Zuordnung (3.)
BAI2040	Objektorientierte Programmierung (OOP)		
	Studiengang (4.)	Bachelor Angewandte Informatik/ Bachelor Angewandte Informatik DUAL	
	Fakultät (5.)	Gebäudetechnik und Informatik	

Modulverantwortlich (6.)	Prof. Dr.-Ing. Jörg Sahm
Modulart (7.)	Pflichtmodul
Angebotshäufigkeit (8.)	SS
Regelbelegung / Empf. Semester (9.)	BA2
Credits (ECTS) (10.)	5 CP
Leistungsnachweis (11.)	PL (N)
Unterrichtssprache (12.)	Deutsch
Voraussetzungen für dieses Modul (13.)	BAI104: Grundkonzepte der Programmierung
Modul ist Voraussetzung für (14.)	BAI3010: Programmierung Java 1 BAI4010: Programmierung Java 2 BAI6010: Programmierung mobiler Endgeräte BAI6120: Grafische Datenverarbeitung 1 BAI7130: Grafische Datenverarbeitung 2 BAI5530: Einführung Künstliche Intelligenz
Moduldauer (15.)	1 Semester
Notwendige Anmeldung (16.)	-
Verwendbarkeit des Moduls (17.)	Sämtliche Fächer, in denen objektorientierte Programmierkompetenzen benötigt werden

Lehrveranstaltung (18.)		Dozent/in (19.)	Art (20.)	Teilnehmer (maximal) (21.)	Anzahl Gruppen (22.)	SWS (23.)	Workload	
							Präsenz (24.)	Selbst-studium (25.)
1	Objektorientierte Programmierung	Sahm	V	100	1	2	30	15
2	Objektorientierte Programmierung	Sahm	Ü	25	4	2	30	50
					Summe	**4**	**60**	**65**
					Workload für das Modul (26.)		**125**	

Qualifikationsziele (27.)	Die Studierenden können… • die grundlegenden Prinzipien der OOP benennen und mit eigenen Worten und Beispielprogrammen beschreiben • den Zusammenhang zwischen Klassen zur Compilezeit und Instanzen zur Laufzeit darstellen • aus einer verbalen Aufgabenstellung ein sinnvolles System von Klassen ableiten, passende Schnittstellen entwerfen und die dazugehörigen Methoden implementieren • den Lebenszyklus von Objekten konsistent gestalten • die Vor- und Nachteile einfacher und mehrfacher Vererbung sowie deren interne Umsetzung erklären • die Vor- und Nachteile virtueller bzw. abstrakter Methoden sowie deren interne Umsetzung erklären • Aggregations- und Kompositionsbeziehungen zwischen Klassen in geeigneter Weise implementieren
Inhalte (28.)	• Codestyle und seine Bedeutung • Realität als Vorbild eines Softwareentwurfs • Begriff der Abstraktion • Klassen und Instanzen • Membervariablen und Klassenvariablen • Konstruktoren und Destruktor • Kapselung • Begriff der Schnittstelle • Vererbung und Polymorphismus • Virtuelle und abstrakte Methoden • Problematik einfacher Vererbung (Bubble-Up-Prinzip) • Problematik multipler Vererbung • Aggregation und Komposition als Alternative zu Vererbung
Vorleistungen und Modulprüfung (29.)	Vorleistungen: • keine Modulprüfung: • 100% Klausur über 120 min im Prüfungszeitraum
Literatur (30.)	• S. B. Lippman, J. Lajoie, B. E. Moo: C++ Primer • S. B. Lippman: Inside the C++ Object Model • B. Stroustrup: The C++ Programming Language • B. Stroustrup: Programming: Principles and Practice Using C++ • S. Meyers: Effective C++