

9 Programmierung II	
Semester	2
Dauer (Semester)	einsemestrig
Credit Points	5
Pflicht/ Wahlpflicht	Pflicht
Häufigkeit des Angebotes/ Verwendbarkeit	Jedes Semester nach Bedarf der Partner-Hochschulen / Online-Bachelorstudiengang Regenerative Energien
Modulverantwortliche(r)	Prof. Dr. Carsten Link
Lerngebiet	Grundlagen der Informationstechnik
Teilnahmevoraussetzungen	Erfolgreicher Abschluss des Moduls Programmierung I wird empfohlen
Lernergebnisse	<p>Die Studierenden demonstrieren, dass sie über ein mentales Modell von Informationsdarstellung und Programmablauf verfügen, indem sie einen Rechner (mit rekursiv absteigendem Parser) für Bool'sche Ausdrücke konstruieren.</p> <p>Die Studierenden können</p> <ul style="list-style-type: none"> • mit den Besonderheiten des Programmablaufs bei der Mikrocontrollerprogrammierung umgehen, indem sie in einem Programm asynchrone Ereignisse mit den adäquaten Sprachmitteln behandeln und die geeigneten Datenstrukturen auswählen. • Klassen entwerfen, um mit angepassten Typen Aufgabenstellungen besser (abstrakter) umsetzen zu können, indem sie eine oder mehrere Klassen deklarieren und definieren, welche in einem vorgegebenen Programm verwendet werden. • mittels der wesentlichen Konzepte der objektorientierten Programmierung Programme entwerfen, indem sie diese Konzepte bei der Implementierung einer Aufgabenstellung verwenden. • die verschiedenen Arten des Polymorphismus differenzieren, um mit dem jeweils passenden Code von konkreten Typen zu abstrahieren. • ein vorgegebenes Programm mit den passenden Arten polymorpher C++-Sprachmittel ausstatten und bezüglich der Lesbarkeit und Wartbarkeit verbessern. • wesentliche Teile der C++-Standardbibliothek anwenden; hierbei können sie für kleine Problemstellungen die richtigen Bibliotheksteile benennen. • ausgewählte Bibliotheksteilen bei der Implementierung einer Aufgabenstellung benutzen und einschätzen, welche die dazu besser geeigneten sind. • (einige wenige) Idiome und Patterns auf eigenen Code adaptieren.

	<ul style="list-style-type: none"> eine gegebene Problemstellung analysieren, in kleinere Teile zerlegen und mit den jeweils angemessenen C/C++-Sprachmitteln implementieren.
Prüfungsvorleistung	Einsendeaufgabe
Medien-/ Lernform	Multimedial aufbereitetes Online-Studienmodul zum Selbststudium mit zeitlich parallel laufender Online-Betreuung (E-Mail, Foren, Chat, Webkonferenzen, Einsendeaufgaben u. a.) sowie Präsenzphase
Arbeitsaufwand	Selbststudium: ca. 145 h Präsenzteilnahme: ca. 3 h Prüfung: 120 Minuten (Präsenzteilnahme ist freiwillig)
Präsenzart	In Online-Konferenz möglich
Prüfungsform	Klausur (120 min.)
Literatur	Stroustrup, Bjarne (2014): The C++ programming language. [C++ 11]. 4. ed., 2. print. Upper Saddle River, NJ: Addison-Wesley. Kirch, Ulla; Prinz, Peter (2013): C++ - das Übungsbuch. 4., überarb. Aufl. Heidelberg, München, Landsberg, Frechen, Hamburg: mitp.
weitere Hinweise	Dieses Modul wird auf Deutsch angeboten

Studieninhalte
<p>Berechnungen auf wichtigen Datentypen Elementare Datentypen, char-ASCII-Glyph, I/O, Bitschieberei; call stack, Rekursion; free store</p> <p>Nichtlinearer Programmablauf Reentranz; event-based programming vs. thread-based; interrupt service routines; locking, lock-free data structures</p> <p>Benutzerdefinierte Datentypen Komposition neuer benutzerdefinierter Datentypen, um Ausdrucksmächtigkeit zu erhöhen; operator overloading; einfache Klasse ohne Vererbung; Typumwandlung (implizit/explicit)</p> <p>Objektorientierte Programmierung Information hiding; subtyping; Interface vs. Implementation; aggregation vs. composition; Identität vs. Äquivalenz; Delphi-style OO (i.e. only free store objects, no assignment operator, only explicit ctors)</p> <p>Polymorphismus Subtyping polymorphism mit virtuellen Methoden; Einordnung des bekannten Polymorphismustypen ad-hoc polymorphism; Generic Polymorphism (templates)</p> <p>C++-Standardbibliothek Std Container; Std Algorithmen; moderner C++-Progammierstil</p> <p>Idiome und Muster Constructional: virtual ctor, factory, ...; Resource Acquisition Is Initialization (RAII); Design-by-contract (DBC)</p> <p>Eleganz und diesbezügliche Hindernisse</p>

Klassendesign; Schwierigkeiten bei der Kombination von Sprach-Features; Objekte und Pointer (copy ctor, operator=, ...); Vererbung und operator overloading; Vererbung und container