03-IBAP-UB	Übersetzerbau
	Compiler Construction

Lehrform *(teaching format)* / **SWS** *(hours per week)*: 3VL + 1UE

Kreditpunkte *(credit points)*: 6

Turnus *(frequency)*: i.d.R. jedes SoSe

Inhaltliche Voraussetzungen *(content-related prior knowledge/skills)*: Theoretische Informatik 1+2

Sprache *(language)*: Deutsch

Lehrende *(teaching staff)*: Dr. Thomas Röfer

Studiengang *(degree program)*	Module	Semester
Informatik (Bachelor VF)	IBAP	ab 4. Sem.
Informatik (Bachelor KF)	KINF-A1/A2	ab 4. Sem.
Informatik (Master)	*General Studies*	ab 1. Sem.
(Industr.)Mathematics (Master)	Anwendungsfach Informatik	

Lernergebnisse:

- Prinzipien der Strukturierung von Übersetzern und Interpretern verstehen und anwenden können.
- Konzepte und Methoden der lexikalischen, syntaktischen und kontextuellen (statisch semantischen) Analyse verstehen, anwenden, auf die Implementierung konkreter Sprachen übertragen, beurteilen und bewerten können.
- Prinzipien der Übersetzung von imperativen und objektorientierten Programmiersprachen in Maschinencode verstehen, auf die Implementierung konkreter Konzepte übertragen und die Qualität des Codes beurteilen können.
- Prinzipien der Codeerzeugung (Registerzuteilung, Instruktionsauswahl, globale und lokale Optimierung) verstehen können.
- selbstständig und in kleinen Teams Wissen und Verständnis erwerben und darstellen können.

Learning Outcome:

- Being able to understand and apply the principles of the structure of compilers and interpreters.
- Being able to understand, apply (for the implementation of concrete languages), and assess the concepts and methods of lexical, syntactical, and contextual (statically semantic) analysis.
- Being able to understand the principles of compiling imperative and object-oriented programming languages into machine code (applied to the implementation of concrete concepts) and assessing the quality of code.
- Being able to understand the principles of code generation (register assignment, instruction selection, global and local optimisation).
- Being able to acquire knowledge and understanding and the ability to present it individually and in small teams.

Inhalte:

- Implementierung von Programmiersprachen mit Interpretern und Übersetzern.
- Strukturierung von Übersetzern: Plattform(un)abhängigkeit, Bootstrap, Phasen.

- Lexikalische Analyse: reguläre Definitionen, endliche Automaten, Symboltabellen, Benutzung von lex/flex.
- Syntaxanalyse: kontextfreie Grammatiken, ab- und aufsteigendes Parsieren, Baumaufbau, Fehlerbehandlung, Benutzung von yacc/bison.
- Kontext-Analyse: Attributgrammatiken, Auswerter, Vereinbarungstabellen.
- Transformation von imperativen und objektorientierten Programmen in abstrakten Maschinencode.
- Grundzüge der Codeerzeugung für konkrete Maschinen: globale Optimierung, Registerzuteilung, Instruktionsauswahl, lokale Optimierung.

In der Übung Anwendung der in der Vorlesung erworbenen Kenntnisse und Fähigkeiten auf spezifische Konstrukte von Programmiersprachen.

Contents:

- Implementation of programming languages with interpreters and compilers.
- Structure of compilers: platform (in)dependence, bootstrapping, phases.
- Lexical analysis: regular definitions, finate state machines, symbol tables, usage of lex/flex.
- Syntax analysis: context free grammars, top-down and bottom-up parsing, parse tree, error handling, usage of yacc/bison.
- Context analysis: attribute grammars, evaluaters, declaration tables.
- Transformation of imperative and object-oriented programs into abstract machine code.
- Basics of code generation for real machines: global optimization, register assignment, instruction selection, local optimization.

In the assignments, the acquired knowledge and abilities are applied to specific constructs of programming languages.

Hinweise *(remarks)*: In der Tabelle sind nur die primären/spezifischsten Module aufgelistet, denen diese Veranstaltung zugeordnet ist.