FACHHOCHSCHULE
WIENER NEUSTADT
University of Applied Sciences – Austria

Computer Science
Bachelor Study Programme

# Qualifications, Modules and Courses

Michael Szvetits
*Head of Programme*

Version 1.3

# Contents

# 1 Qualifications

| **Taxonomy** | **Learning Outcome**<br>After completing the curriculum, students … |
|---|---|
| LO.01 | know and are able to apply the essential theories, principles, standards and methods of computer science and software engineering. |
| LO.02 | know and are able to apply various aspects and dimensions of reliable software systems. |
| LO.03 | understand operational contexts to plan and model software systems and are able to perform practical tasks in this area. |
| LO.04 | are able to complete projects using industrial standards while assessing and integrating contributions from stakeholders. |
| LO.05 | are able to acquire new specialized knowledge and understand the specifics of various application domains. |
| LO.06 | are able to dispel contemporary trends in the field and reduce them to their computational foundations. |
| LO.07 | understand and are able to adhere to the legal and organizational framework conditions of software systems. |
| LO.08 | are able to apply one's own professional knowledge and skills, as well as understand their interrelationships and limitations. |
| LO.09 | are able to perform research under guidance and systematically answer research questions using established methods. |

# 2 Modules

## 2.1 Computer Science Foundations

### 2.1.1 Module Profile

| | |
|---|---|
| Abbreviation: | CSF |
| Workload: | 27.0 ECTS |
| Semesters: | 1, 2, 3 |
| Addressed learning outcomes: | LO.01, LO.03, LO.06, LO.08 |

### 2.1.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to ... |
|---|---|
| CSF-LO.01 | understand and apply the theoretical, logical and mathematical foundations of computer science. |
| CSF-LO.02 | understand the operating principles of essential hardware and software components. |
| CSF-LO.03 | understand decidability, computability and complexity theory as well as their practical relevance. |
| CSF-LO.04 | understand the syntax and semantics of programming languages, including how languages are parsed and processed. |
| CSF-LO.05 | understand and apply formal language theory. |

### 2.1.3 Courses

- Discrete Mathematics and Algebra
- Computer Architecture
- Introduction to Computer Science
- Calculus

- Theoretical Computer Science
- Logic and Reasoning

## 2.2 Programming Paradigms

### 2.2.1 Module Profile

| | |
|---|---|
| Abbreviation: | PRO |
| Workload: | 25.0 ECTS |
| Semesters: | 1, 2, 3, 4 |
| Addressed learning outcomes: | LO.01, LO.02, LO.03, LO.06, LO.08 |

### 2.2.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to … |
|---|---|
| PRO-LO.01 | apply different programming paradigms and explain the contexts in which each paradigm is most appropriate. |
| PRO-LO.02 | solve computational problems, develop algorithms and implement solutions using appropriate data structures. |
| PRO-LO.03 | explain, apply and evaluate imperative and declarative problem-solving strategies. |
| PRO-LO.04 | explain memory and apply its management strategies, including manual and automatic memory (de-)allocation. |
| PRO-LO.05 | evaluate implementation strategies with respect to performance and memory consumption. |
| PRO-LO.06 | apply different strategies of parallelization and explain their advantages and limitations. |
| PRO-LO.07 | write reliable and maintainable software using modern programming techniques and concise coding styles. |

### 2.2.3 Courses

- Imperative Programming
- Object-Oriented Programming
- Algorithms and Data Structures
- Functional Programming
- Parallel Computing

## 2.3 System Architecture and Infrastructure

### 2.3.1 Module Profile

| | |
|---|---|
| Abbreviation: | SYS |
| Workload: | 20.0 ECTS |
| Semesters: | 2, 5 |
| Addressed learning outcomes: | LO.01, LO.02, LO.03, LO.06, LO.08 |

### 2.3.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to ... |
|---|---|
| SYS-LO.01 | understand the principles and components of operating systems and apply their fundamental concepts. |
| SYS-LO.02 | understand the process of how high-level source code is turned into a low-level executable format. |
| SYS-LO.03 | understand the principles and architectures of computer networks in the small and in the large, including formats and protocols. |
| SYS-LO.04 | configure and manage network devices and services. |
| SYS-LO.05 | understand the architecture and apply the principles of distributed systems. |
| SYS-LO.06 | manage consistency, fault tolerance and scalability in distributed environments. |
| SYS-LO.07 | understand and apply the fundamental concepts of information security. |
| SYS-LO.08 | understand system reliability and identify and mitigate security threats and vulnerabilities. |

### 2.3.3 Courses

- Operating Systems
- Computer Networks
- Security and Privacy
- Distributed Systems

## 2.4 Software Engineering

### 2.4.1 Module Profile

| | |
|---|---|
| Abbreviation: | SWE |
| Workload: | 18.0 ECTS |
| Semesters: | 2, 3, 4 |
| Addressed learning outcomes: | LO.01, LO.02, LO.03, LO.04, LO.06, LO.08 |

### 2.4.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to ... |
|---|---|
| SWE-LO.01 | understand and apply various phases of the software development lifecycle. |
| SWE-LO.02 | understand economic and other impacts of inadequate software development practices. |
| SWE-LO.03 | understand the importance of project management, stakeholder involvement and teamwork in a software project. |
| SWE-LO.04 | develop software architecture and design artifacts for guiding the implementation of software systems. |
| SWE-LO.05 | apply software design principles to create modular, reusable and maintainable software. |
| SWE-LO.06 | apply software verification and quality assurance principles to ensure the quality of software systems. |
| SWE-LO.07 | understand type systems and type checking, and how they contribute to the reliability and safety of software. |
| SWE-LO.08 | apply configuration management and version control to ensure consistency and reproducibility. |

### 2.4.3 Courses

- Software Engineering and Project Management
- Software Architecture and Design
- Software Verification and Validation
- Software Review and Refactoring

## 2.5 Data and Intelligence

### 2.5.1 Module Profile

| | |
|---|---|
| Abbreviation: | INT |
| Workload: | 15.0 ECTS |
| Semesters: | 1, 3, 5 |
| Addressed learning outcomes: | LO.01, LO.02, LO.03, LO.06, LO.08 |

### 2.5.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to ... |
|---|---|
| INT-LO.01 | understand the architecture and properties of database management systems. |
| INT-LO.02 | understand and apply the principles of database design and modeling. |
| INT-LO.03 | apply database creation, manipulation and querying to practical problems. |
| INT-LO.04 | give an overview of various approaches that fall under the umbrella term ”artificial intelligence”. |
| INT-LO.05 | understand the statistical and technical background of artificial intelligence approaches. |
| INT-LO.06 | apply artificial intelligence approaches to practical problems and understand the importance of data. |
| INT-LO.07 | analyze and visualize data. |

### 2.5.3 Courses

- Database Systems
- Statistics and Visualization
- Artificial Intelligence

## 2.6 Interaction and User Engagement

### 2.6.1 Module Profile

| | |
|---|---|
| Abbreviation: | USE |
| Workload: | 15.0 ECTS |
| Semesters: | 4, 5 |
| Addressed learning outcomes: | LO.01, LO.02, LO.03, LO.05, LO.08 |

### 2.6.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to ... |
|---|---|
| USE-LO.01 | understand and apply fundamental concepts of user-centered design and usability principles. |
| USE-LO.02 | develop applications for specialized platforms like mobile devices and the web. |
| USE-LO.03 | develop applications that are responsive, resilient and elastic. |

### 2.6.3 Courses

- User Experience
- Web Development
- Mobile Development

## 2.7 Ethical Issues and Legal Frameworks

### 2.7.1 Module Profile

| | |
|---|---|
| Abbreviation: | LAW |
| Workload: | 12.0 ECTS |
| Semesters: | 2, 3, 6 |
| Addressed learning outcomes: | LO.07, LO.08 |

### 2.7.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to ... |
|---|---|
| LAW-LO.01 | evaluate key legal and ethical frameworks governing intellectual property, data protection, artificial intelligence and employment. |
| LAW-LO.02 | analyze intellectual property management strategies that balance innovation, protection and compliance with international treaties and national laws. |
| LAW-LO.03 | evaluate and implement solutions that ensure compliance with data protection laws, addressing ethical considerations in data usage, storage and AI-driven analytics. |
| LAW-LO.04 | assess regulatory approaches to artificial intelligence, including ethical considerations and liability issues. |
| LAW-LO.05 | analyze competition law frameworks, focusing on the regulation of monopolistic practices, antitrust/competition enforcement and their impact on innovation and market dynamics in the tech industry. |
| LAW-LO.06 | analyze and apply international employment regulations, including work permits, self-employment frameworks and cross-border compliance strategies, to facilitate global workforce mobility. |
| LAW-LO.07 | analyze the complexities of legal regulations and ethical concerns in global contexts, focusing on their consequences on software innovation, competition and societal well-being. |

### 2.7.3 Courses

- Intellectual Property and Data Protection Law
- International Employment Law

- Legal Frameworks of Artificial Intelligence and Market Regulation
- Human Factors in Computer Science

## 2.8 Social Issues and Professional Practice

### 2.8.1 Module Profile

| | |
|---|---|
| Abbreviation: | SIP |
| Workload: | 27.0 ECTS |
| Semesters: | 1, 3, 4, 5, 6 |
| Addressed learning outcomes: | LO.03, LO.04, LO.05, LO.08 |

### 2.8.2 Learning Outcomes

| Taxonomy | Learning Outcome<br>After completing the module, students are able to ... |
|---|---|
| SIP-LO.01 | understand and apply techniques for conflict resolution and knowledge acquisition. |
| SIP-LO.02 | apply the acquired competencies and skills to practical tasks. |
| SIP-LO.03 | collaborate in distributed international teams. |

### 2.8.3 Courses

- Learning and Study Techniques
- Team Communication and Conflict Management
- Computer Science Project I
- Computer Science Project II
- Computer Science Project III
- Professional Internship

## 2.9 Scientific Research

### 2.9.1 Module Profile

| | |
|---|---|
| Abbreviation: | SCI |
| Workload: | 21.0 ECTS |
| Semesters: | 3, 4, 6 |
| Addressed learning outcomes: | LO.01, LO.05, LO.08, LO.09 |

### 2.9.2 Learning Outcomes

| **Taxonomy** | **Learning Outcome**<br>After completing the module, students are able to … |
|---|---|
| SCI-LO.01 | formulate research questions and answer them using established research methods under academic integrity. |
| SCI-LO.02 | abide by the current state of scientific knowledge regarding data acquisition, analysis and interpretation. |
| SCI-LO.03 | author a research paper using an established document structure and a scientific writing style. |
| SCI-LO.04 | communicate research topics and research results to an academic audience. |

### 2.9.3 Courses

- Scientific Writing
- Scientific Methods
- Bachelor Thesis
- State of Computer Science

# 3 Courses

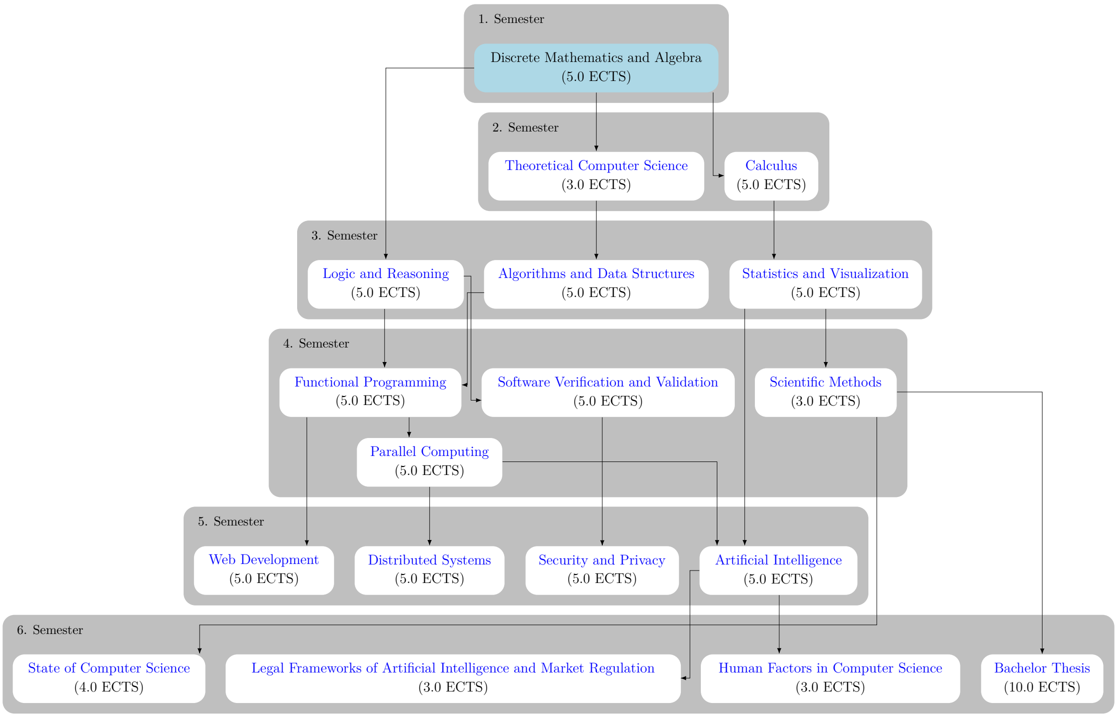
## 3.1 1. Semester

### 3.1.1 Discrete Mathematics and Algebra

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Theoretical Computer Science (3.0 ECTS)
Calculus (5.0 ECTS)
3. Semester
Logic and Reasoning (5.0 ECTS)
Algorithms and Data Structures (5.0 ECTS)
Statistics and Visualization (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Software Verification and Validation (5.0 ECTS)
Scientific Methods (3.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Web Development (5.0 ECTS)
Distributed Systems (5.0 ECTS)
Security and Privacy (5.0 ECTS)
Artificial Intelligence (5.0 ECTS)
6. Semester
State of Computer Science (4.0 ECTS)
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)
Bachelor Thesis (10.0 ECTS)

### Course Contents

- Orientation within the curriculum
- Propositional logic
- Number theory
- Boolean algebra
- Set theory
- Relations and functions
- Combinatorics
- Graphs
- Algebras
- Vectors
- Matrices
- Linear equations
- Recurrence relations

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the syntax and semantics of propositional logic, including truth tables, logical equivalences and normal forms. | CSF-LO.01 |
| set up, evaluate and simplify logical expressions. | CSF-LO.01 |
| explain concepts from number theory, like divisibility, modular arithmetic, greatest common divisors and prime factorization, and apply them to computer science problems, like cryptography. | CSF-LO.01 |
| explain the principles of Boolean algebra, including its axioms and theorems, and apply them to computer science problems, like logical circuit design. | CSF-LO.01 |

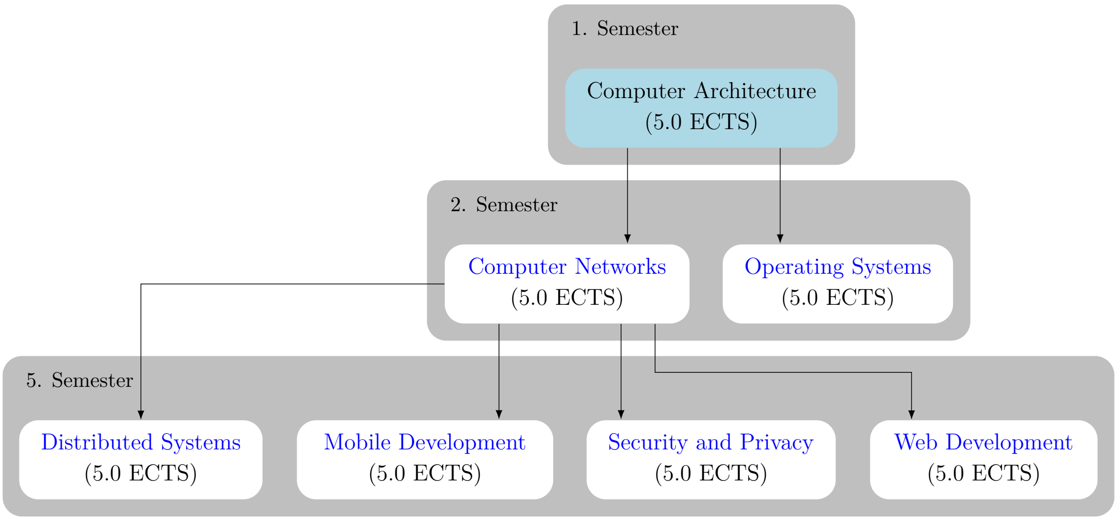
| | |
|---|---|
| explain and apply the foundational concepts of sets, subsets, operations on sets and Venn diagrams. | CSF-LO.01 |
| explain the properties of relations, equivalence relations, partial orders and functions. | CSF-LO.01 |
| model real-world relationships using relations and functions, and solve problems involving mappings, closures and composition. | CSF-LO.01 |
| explain the principles of counting, including permutations and combinations, and apply them to practical problems like arrangements and selections. | CSF-LO.01 |
| explain graph theory, including graph representations, types of graphs, paths, cycles and connectivity. | CSF-LO.01 |
| apply graph algorithms like shortest path, minimum spanning tree and network flow to real-world scenarios. | CSF-LO.01 |
| explain and apply algebraic structures like groups, rings and fields. | CSF-LO.01 |
| explain the properties and operations of vectors, and apply them to computer science problems, like geometric problems and data transformations. | CSF-LO.01 |
| apply matrices to computer science problems, like solving systems of linear equations and performing transformations in computer graphics. | CSF-LO.01 |
| solve systems of linear equations using various strategies, like Gaussian elimination, LU decomposition and matrix inversion. | CSF-LO.01 |
| apply recurrence relations to computer science problems, like divide-and-conquer algorithms and the visualization of fractals. | CSF-LO.01 |
| demonstrate meticulousness by verifying the correctness and completeness of calculations. | LO.08 |

### 3.1.2 Computer Architecture

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- History of computer architectures
- von Neumann architecture
- Processor memory hierarchy
- Latencies in computer systems
- Spatial and temporal locality
- Caches and cache coherency
- Gates, flip-flops, registers
- Sequential logic

- Boolean algebra
- Sum-of-products form and minimization
- SIMD and MIMD architectures
- I/O devices
- Storages and drives
- Buses
- Multiprocessors and multicore organization
- Assembly language
- PC hardware

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain how historical developments have influenced current computer architectures. | LO.06 |
| explain the components of the von Neumann architecture and its corresponding fetch-decode-execute cycle. | CSF-LO.02 |
| describe different levels of memory and their impact on system performance. | CSF-LO.02 |
| describe and calculate various types of latencies in a computer system. | CSF-LO.02 |
| describe caches and cache coherence problems in multiprocessor systems. | CSF-LO.02 |
| explain how logic gates, flip-flops and registers form the building blocks of memory and processor control units. | CSF-LO.02 |
| explain how sequential logic is used in processors for instruction sequencing and pipeline control. | CSF-LO.02 |
| simplify Boolean expressions and functions using laws, theorems and minimization. | CSF-LO.01,<br>CSF-LO.02 |

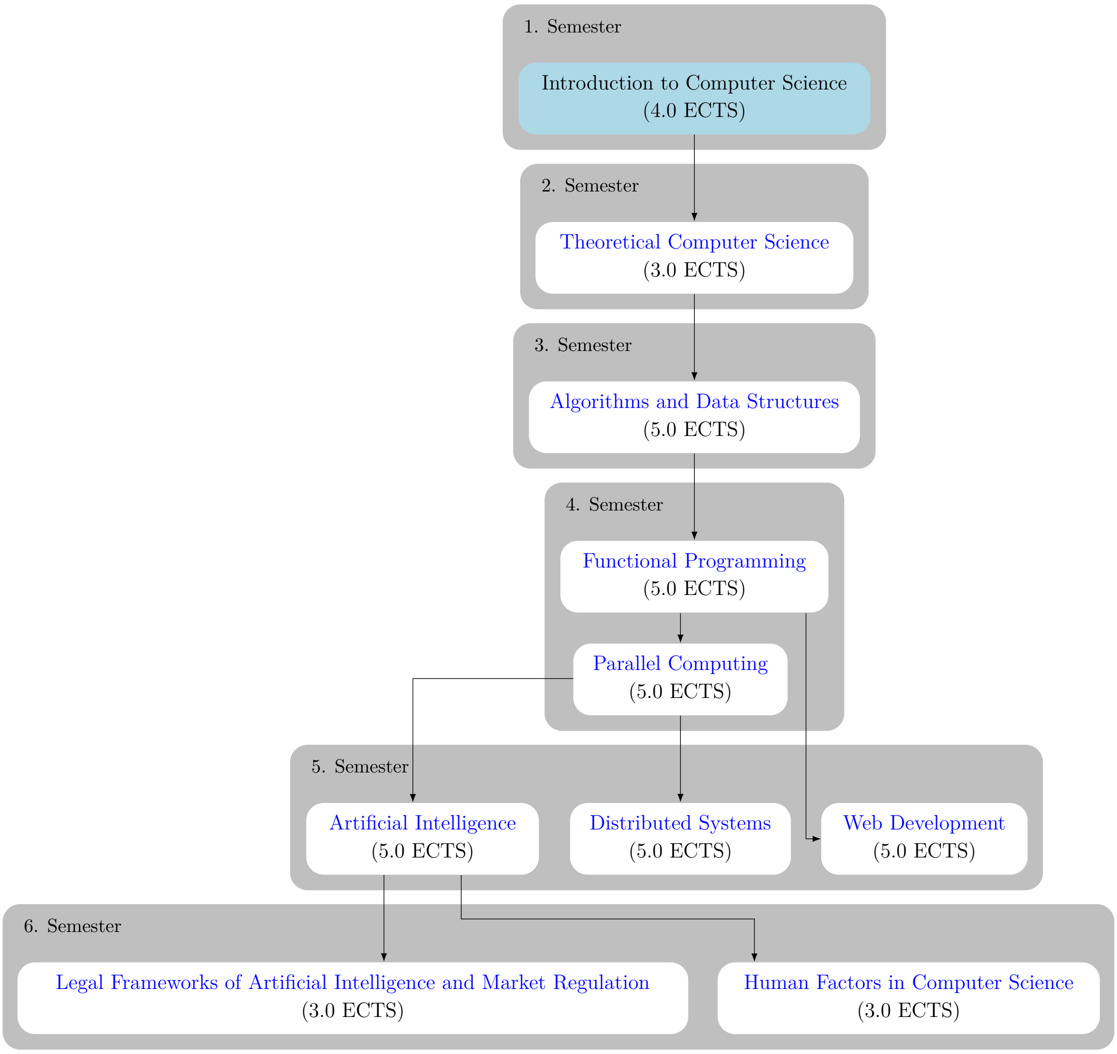
| | |
|---|---|
| explain the benefits and challenges of parallel computer architectures. | CSF-LO.02 |
| explain various I/O devices and techniques, such as polling, interrupts, memory-mapping and Direct Memory Access. | CSF-LO.02 |
| explain different types of storage devices and their characteristics, such as access time, reliability and cost. | CSF-LO.02 |
| explain different types of buses (e.g., data bus, address bus, control bus) and their impact on system performance. | CSF-LO.02 |
| write simple programs in an assembly language. | CSF-LO.02 |
| describe the components of a personal computer, explain their relevance and assemble them correctly. | CSF-LO.02 |
| demonstrate meticulousness by verifying the correctness and completeness of programs and computer assemblies. | LO.08 |

### 3.1.3 Introduction to Computer Science

#### Course Profile

| | |
|---|---|
| Workload: | 4.0 ECTS |
| Teaching units: | 56 |
| Course type: | Lecture with exercises |
| Assessment: | Final |

#### Context and Dependencies


1. Semester
Introduction to Computer Science (4.0 ECTS)
2. Semester
Theoretical Computer Science (3.0 ECTS)
3. Semester
Algorithms and Data Structures (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
Distributed Systems (5.0 ECTS)
Web Development (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)

## Course Contents

- Orientation within the curriculum
- History of computer science
- Future of computer science
- Subfields of computer science
- Signals, messages, information
- Data transmission and communication
- Digitalization and discretization
- Character encoding
- Bits, bytes, words and endianness
- Error detection and correction
- Numeral systems
- State machines
- Modeling and abstraction principle

## Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain historical milestones, current trends and emerging technologies in computer science. | LO.06 |
| explain common subfields of computer science and their mapping onto the curriculum. | LO.08 |
| explain how information of various sources is created, represented, transmitted and received. | CSF-LO.01 |
| explain the conversion of analog signals into digital form and the representation of continuous variables as discrete values. | CSF-LO.01 |

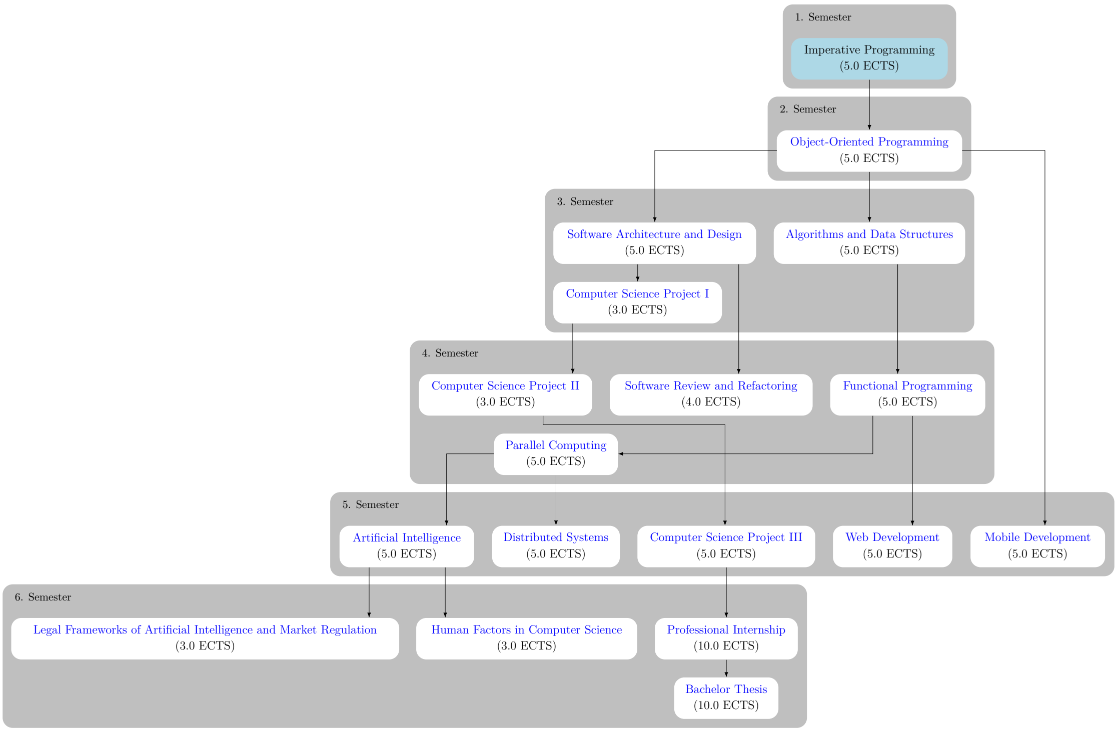
| | |
|---|---|
| explain how data is stored and manipulated in computer memory and how different architectures impact the way data is read and written. | CSF-LO.01 |
| describe and apply basic error detection and correction techniques for reliability, such as parity checks and checksums. | CSF-LO.01 |
| convert numbers between various numeral systems and explain the significance of various representations for computer science. | CSF-LO.01 |
| create finite state machines and explain how they are used to model computational processes. | CSF-LO.01 |
| explain how modeling and abstraction are used to simplify complex systems and problems in computer science. | CSF-LO.01 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions when applying selected concepts. | LO.08 |

### 3.1.4 Imperative Programming

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
3. Semester
Software Architecture and Design (5.0 ECTS)
Algorithms and Data Structures (5.0 ECTS)
Computer Science Project I (3.0 ECTS)
4. Semester
Computer Science Project II (3.0 ECTS)
Software Review and Refactoring (4.0 ECTS)
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
Distributed Systems (5.0 ECTS)
Computer Science Project III (5.0 ECTS)
Web Development (5.0 ECTS)
Mobile Development (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)
Professional Internship (10.0 ECTS)
Bachelor Thesis (10.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Principles of imperative programming
- Syntax and semantics of the selected programming language
- Variables, statements and expressions
- Control structures

- Input and output
- Data types and arrays
- Functions and operators
- Debugging and exception handling
- Translation of source code
- Memory management, layout and addressing
- Version control

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to … | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the syntax and semantics of the used imperative programming language. | PRO-LO.01 |
| explain the imperative programming paradigm, including statements, expressions, variables and their evaluation. | PRO-LO.01 |
| explain and apply the necessary steps for turning written code into an executable program. | PRO-LO.01,<br>PRO-LO.03 |
| explain and systematically apply imperative programming constructs like structs, enumerations and functions. | PRO-LO.01,<br>PRO-LO.03 |
| explain and systematically apply sequence, selection and iteration control structures. | PRO-LO.01,<br>PRO-LO.03 |
| explain compile and runtime errors and correct them in a systematic way. | PRO-LO.01,<br>PRO-LO.03 |
| explain the purpose of memory and the layout of objects in memory. | PRO-LO.04 |
| explain and systematically apply manual resource management strategies using imperative programming techniques. | PRO-LO.01,<br>PRO-LO.03,<br>PRO-LO.04 |
| explain the role of resource management for developing reliable software systems. | PRO-LO.04,<br>PRO-LO.07 |

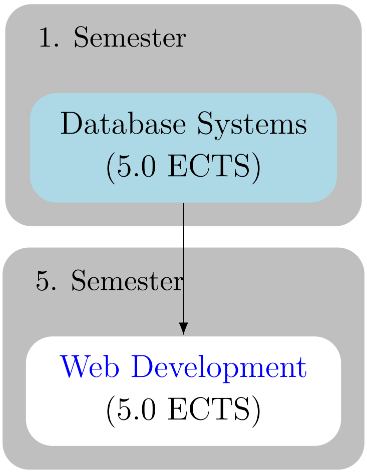
| | |
|---|---|
| explain and apply exception handling mechanisms using imperative programming techniques. | PRO-LO.01,<br>PRO-LO.03,<br>PRO-LO.07 |
| develop imperative programs using basic I/O operations for streams and files. | PRO-LO.01,<br>PRO-LO.03 |
| transform informal task specifications into formal imperative implementations using concise coding styles. | PRO-LO.01,<br>PRO-LO.03,<br>PRO-LO.07 |
| utilize a version control system to manage code artifacts. | PRO-LO.07 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions. | LO.08 |
| demonstrate persistence and internalize that problem-solving is an iterative process of overcoming different forms of failures. | LO.08 |
| demonstrate self-directedness and internalize that informal task specifications are never complete. | LO.08 |

### 3.1.5 Database Systems

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Database systems and architectures
- Data warehousing and data lifecycle
- Metadata
- Relational model and relational algebra
- SQL (data definition, data manipulation, query)
- Database design
- Entity Relationship (ER) modeling
- Normalization
- Transaction management

- ACID and isolation
- File organization and indices
- Functions, procedures, exceptions
- Programming language integration

### Learning Outcomes

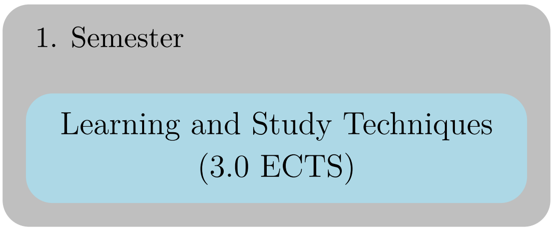
| **Learning Outcome**<br>After completing the course, students are able to … | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the role and the properties of databases in the context of reliable software systems. | INT-LO.01 |
| explain the purposes, advantages and disadvantages of different database management systems. | INT-LO.01 |
| explain the principles of relational database theory and apply them to practical tasks. | INT-LO.02 |
| explain the inner workings of important database manipulation operations. | INT-LO.03 |
| model, improve and deploy database designs based on verbal requirement descriptions of real-world scenarios. | INT-LO.02 |
| query and manipulate data stored in relational databases within real-world scenarios. | INT-LO.03 |
| query and manipulate data stored in relational databases using a conventional programming language. | INT-LO.03 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |

### 3.1.6 Learning and Study Techniques

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 30 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



1. Semester

Learning and Study Techniques (3.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Goal definition and prioritization
- Learning theory, planning and techniques
- Stress management
- Time management and work techniques
- Argumentation and presentation
- Software tools
- Search engines and chat bots

#### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |

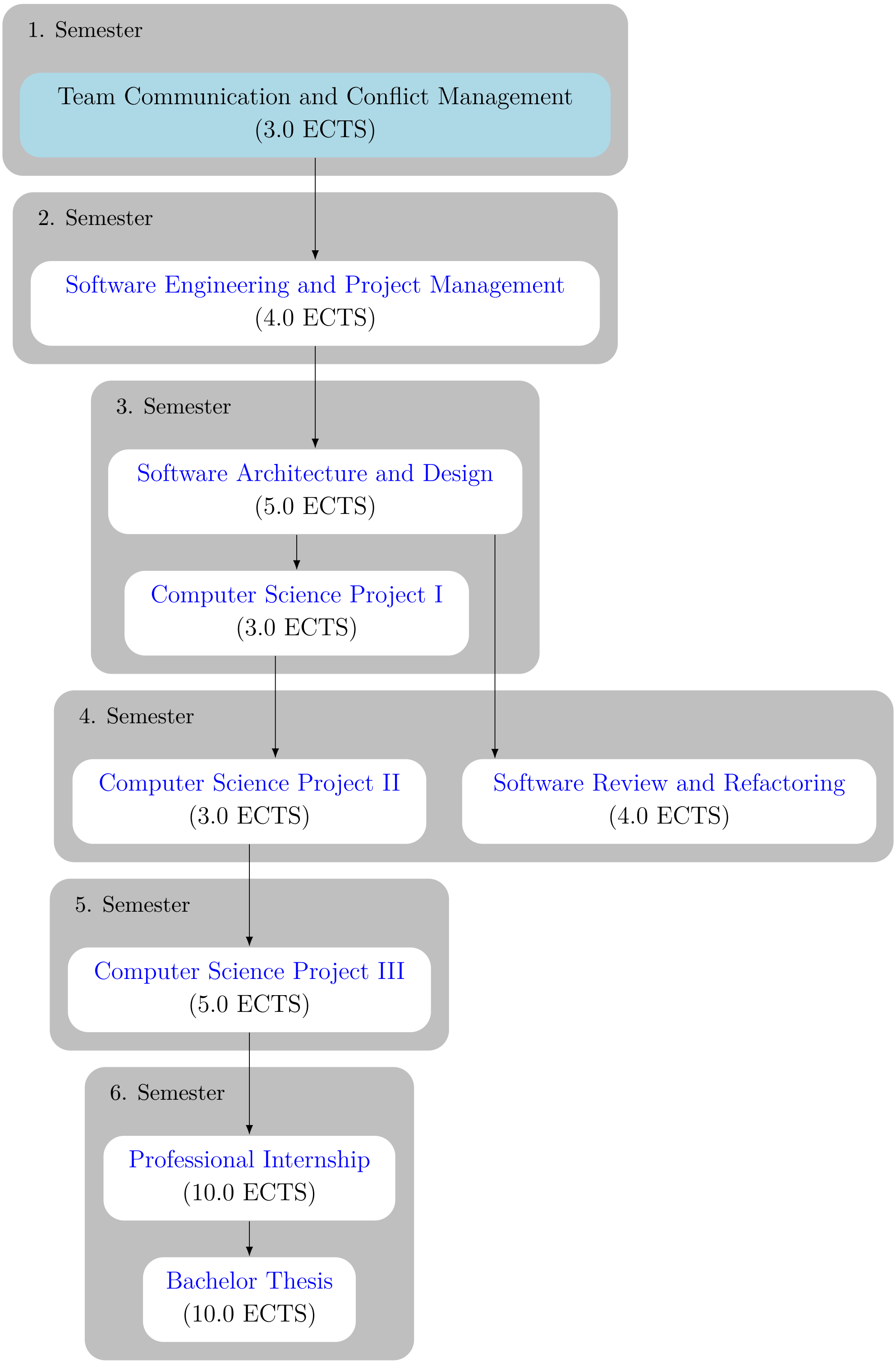
| | |
|---|---|
| organize one's study routine using self-management techniques and utilize one's resources efficiently. | LO.08,<br>SIP-LO.01 |
| systematically apply software tools to manage and improve one's study routine. | LO.08,<br>SIP-LO.01 |
| formulate terms for online research and search engines in a goal-directed way. | LO.08,<br>SIP-LO.01 |
| apply fundamentals of communication and explain various models of communication. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |
| collaborate in teams and give constructive feedback. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |
| explain group dynamic models and reflect on group processes and one's own role within a team. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |
| present and argument in presentations and discussions on a level that is adequate for the respective audience. | LO.08,<br>SIP-LO.01 |
| demonstrate willingness for collaboration to improve the work of both peers and one's own. | LO.08,<br>SIP-LO.03 |

### 3.1.7 Team Communication and Conflict Management

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 30 |
| Course type: | Integrated course |
| Assessment: | Continuous |

### Context and Dependencies


1. Semester
Team Communication and Conflict Management
(3.0 ECTS)
2. Semester
Software Engineering and Project Management
(4.0 ECTS)
3. Semester
Software Architecture and Design
(5.0 ECTS)
Computer Science Project I
(3.0 ECTS)
4. Semester
Computer Science Project II
(3.0 ECTS)
Software Review and Refactoring
(4.0 ECTS)
5. Semester
Computer Science Project III
(5.0 ECTS)
6. Semester
Professional Internship
(10.0 ECTS)
Bachelor Thesis
(10.0 ECTS)

## Course Contents

- Orientation within the curriculum
- Teambuilding and teamwork
- Communication and feedback techniques
- Models of communication
- Intercultural communication
- Impact of conflict management on professional practice
- Fundamentals of conflict management
- Conflict resolution strategies
- De-escalating communication
- Conflict competence development

## Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the fundamentals of conflicts, strategies for conflict resolution and their impact on professional practice. | LO.08,<br>SIP-LO.01 |
| recognize the various facets of conflict behavior and react accordingly. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |
| reflect over one's own conflict behavior, improve on it and act accordingly in the context of professional practice. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |
| manage conflicts using various models of conversation and communication. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |
| demonstrate willingness for collaboration to improve conflict management skills of both peers and one's own. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |

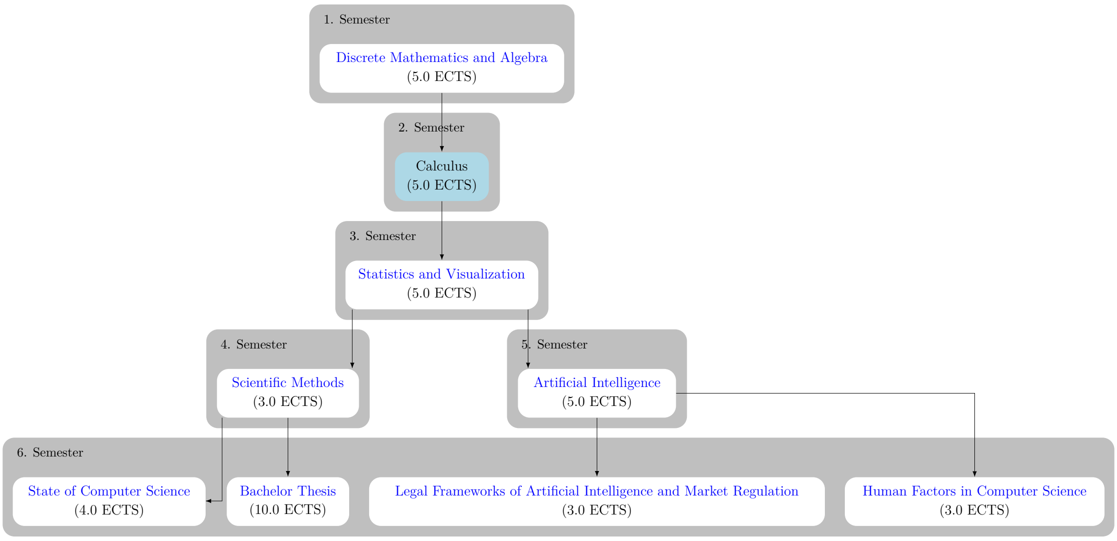
## 3.2 2. Semester

### 3.2.1 Calculus

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Calculus (5.0 ECTS)
3. Semester
Statistics and Visualization (5.0 ECTS)
4. Semester
Scientific Methods (3.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
6. Semester
State of Computer Science (4.0 ECTS)
Bachelor Thesis (10.0 ECTS)
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Sequences and series
- Limits
- Metric spaces
- Functions
- Continuity
- Differential calculus

- Integral calculus
- Gradients
- Gradient descent

### Learning Outcomes

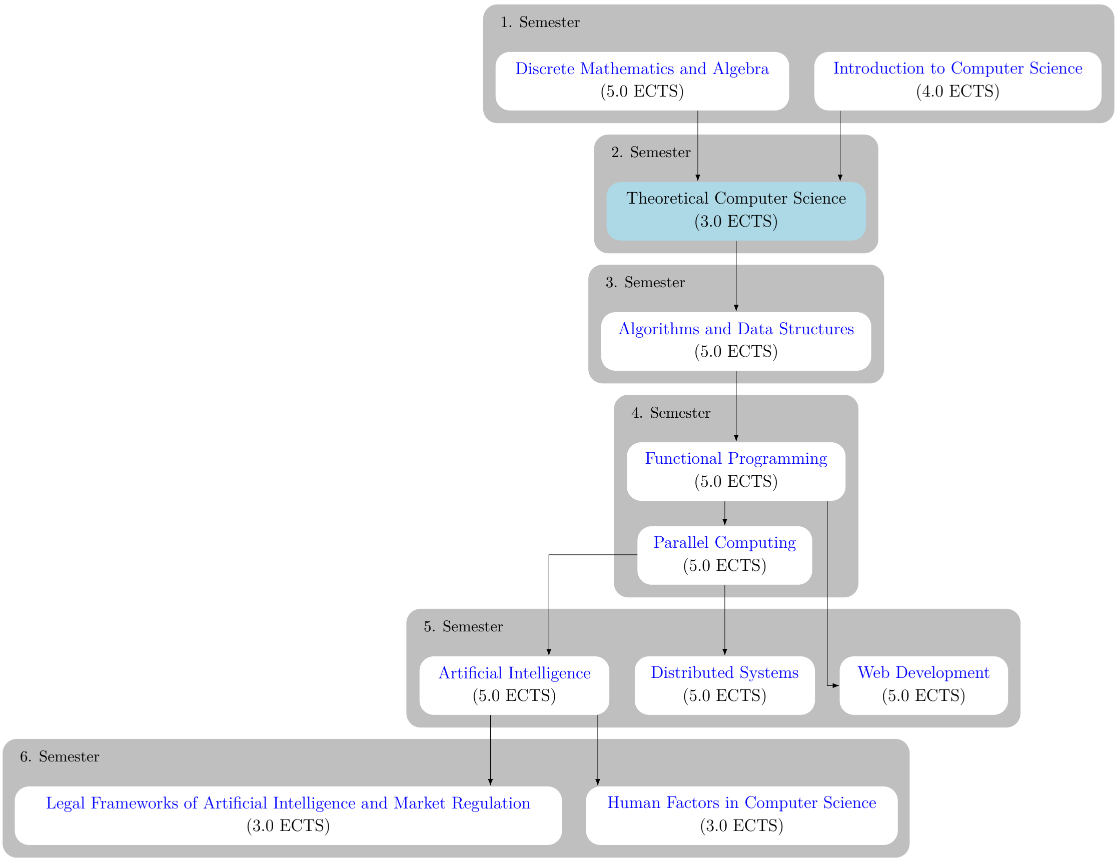
| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the concepts of convergence and divergence in sequences and series, and apply them to practical problems relevant for computer science. | CSF-LO.01 |
| explain limits and apply them to computer science problems, like evaluating the asymptotic behavior of algorithms. | CSF-LO.01 |
| explain metric spaces and apply them to computer science problems, like data clustering and classification. | CSF-LO.01 |
| explain the (geometric) interpretation of derivatives and higher-order derivatives, and systematically apply rules of differentiation. | CSF-LO.01 |
| explain the (geometric) interpretation of integrals and apply them to computer science problems, like computing areas under curves to evaluate probabilities in statistical methods. | CSF-LO.01 |
| explain the role of gradients in multi-variable calculus and apply them to computer science problems, like optimization in artificial intelligence approaches. | LO.06,<br>CSF-LO.01 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions when applying selected concepts. | LO.08 |

### 3.2.2 Theoretical Computer Science

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 42 |
| Course type: | Lecture with exercises |
| Assessment: | Final |

#### Context and Dependencies


1. Semester
Discrete Mathematics and Algebra (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
2. Semester
Theoretical Computer Science (3.0 ECTS)
3. Semester
Algorithms and Data Structures (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
Distributed Systems (5.0 ECTS)
Web Development (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Fundamentals of algorithms
- Alphabets, words, languages

- Regular grammars and expressions
- Formal languages, context-free grammars
- Chomsky hierarchy
- Automata theory, Turing machine
- Recursive functions, lambda calculus
- Computability and decidability
- Complexity theory
- Optimization and approximation

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| define the linguistic terms involved in defining and applying formal languages. | CSF-LO.04, CSF-LO.05 |
| explain recursion, lambda calculus and Turing machine concepts as well as their importance with respect to computability and decidability of problems. | CSF-LO.05 |
| explain the interrelationship between complexity classes. | CSF-LO.03 |
| explain and apply selected strategies for the combinatorial optimization of complex problems. | CSF-LO.03 |
| design and apply formal languages. | CSF-LO.04, CSF-LO.05 |
| design and apply automata. | CSF-LO.05 |
| relate formal languages with their corresponding automata. | CSF-LO.05 |
| determine the automata class for a given formal language, and vice versa. | CSF-LO.05 |
| discuss the computational complexity of a given algorithm. | CSF-LO.03 |
| transform and simplify expressions of the lambda calculus. | CSF-LO.03 |
| determine the type of expressions using the lambda calculus. | CSF-LO.03 |

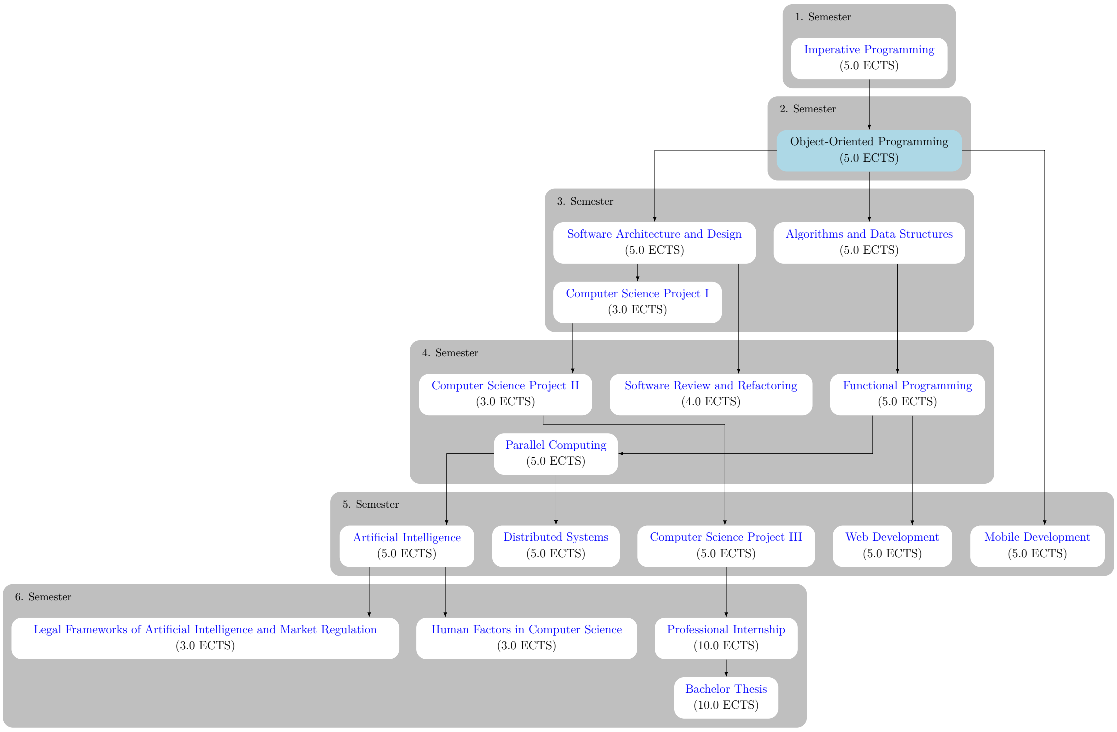
| | |
|---|---|
| apply the gained knowledge from the course to practical examples. | CSF-LO.01 |
| describe and apply techniques of handling complex computational problems. | CSF-LO.03 |
| analyze statements regarding algorithms, languages, automata, computability, decidability and complexity with respect to their correctness. | CSF-LO.01 |
| summarize and analyze scientific papers related to the course content. | LO.06,<br>CSF-LO.01 |
| demonstrate meticulousness by using precise language when making statements in the context of theoretical computer science. | LO.08 |

### 3.2.3 Object-Oriented Programming

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
3. Semester
Software Architecture and Design (5.0 ECTS)
Algorithms and Data Structures (5.0 ECTS)
Computer Science Project I (3.0 ECTS)
4. Semester
Computer Science Project II (3.0 ECTS)
Software Review and Refactoring (4.0 ECTS)
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
Distributed Systems (5.0 ECTS)
Computer Science Project III (5.0 ECTS)
Web Development (5.0 ECTS)
Mobile Development (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)
Professional Internship (10.0 ECTS)
Bachelor Thesis (10.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Principles of object-oriented programming
- Syntax and semantics of the selected programming language
- Program structure using classes and their members
- Debugging and exception handling in contrast to imperative programming

- Encapsulation
- Inheritance
- Polymorphism
- Constructors and destructors
- Value types and reference types
- Memory management in contrast to imperative programming
- Method overloading
- Generic programming

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the differences between the object-oriented and other programming paradigms. | PRO-LO.01 |
| explain the role of object-oriented programming in the context of reliable software systems. | PRO-LO.01,<br>PRO-LO.07 |
| explain the syntax and semantics of the used object-oriented programming language. | PRO-LO.01 |
| explain and systematically apply object-oriented principles like abstraction, polymorphism, inheritance and encapsulation to real-world scenarios. | PRO-LO.01,<br>PRO-LO.03 |
| explain and systematically apply object-oriented programming constructs like classes and members to real-world scenarios. | PRO-LO.01,<br>PRO-LO.03 |
| explain the difference between value types and reference types and apply these concepts systematically. | PRO-LO.01,<br>PRO-LO.03 |
| explain and systematically apply manual resource management strategies using object-oriented techniques (e.g., destructors, the rule of three, finalizers). | PRO-LO.01,<br>PRO-LO.03,<br>PRO-LO.04 |
| explain the role of resource management for developing reliable software systems. | PRO-LO.04,<br>PRO-LO.07 |

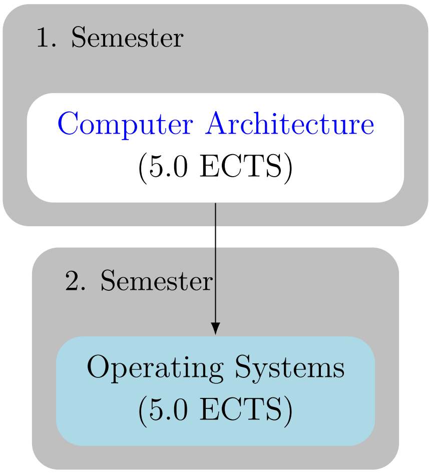
| | |
|---|---|
| explain and apply exception handling mechanisms using object-oriented programming techniques. | PRO-LO.01,<br>PRO-LO.03 |
| explain and systematically apply generic programming techniques to real-world scenarios. | PRO-LO.01,<br>PRO-LO.03 |
| transform informal task specifications into formal object-oriented designs and implementations using concise coding styles. | PRO-LO.01,<br>PRO-LO.03,<br>PRO-LO.07 |
| improve object-oriented designs and implementations in order to comply with renowned object-oriented design guidelines. | PRO-LO.01,<br>PRO-LO.03,<br>PRO-LO.07 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |
| demonstrate meticulousness by ensuring the correctness and completeness of solutions. | LO.08 |
| demonstrate persistence and internalize that problem-solving is an iterative process of overcoming different forms of failures. | LO.08 |
| demonstrate self-directedness and internalize that informal task specifications are never complete. | LO.08 |

### 3.2.4 Operating Systems

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Role of operating systems
- Impact of operating systems on reliability
- Operating system kernel designs
- Virtualization and isolation
- Boot sequence
- System calls and modes
- Interrupts and services routines
- Physical memory, virtual memory and addressing
- Paging and thrashing

- File systems, (memory-mapped) files, metadata and access
- Processes, threads and their isolation
- Interprocess communication
- Preemptive and non-preemptive process scheduling
- Context switching
- Linking
- Memory segmentation
- Instruction set architecture
- Virtual machines
- Scripting and systems programming

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to … | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the fundamental role of operating systems in managing hardware and software resources reliably. | SYS-LO.01 |
| compare different operating system kernel designs in terms of performance, security, reliability and maintainability. | SYS-LO.01 |
| install and configure an operating system in a virtual environment, including virtual file systems and virtual devices. | SYS-LO.01 |
| explain the boot sequence of an operating system and trace the boot sequence using analysis tools and boot logs. | SYS-LO.01 |
| explain user mode and kernel mode, and trace system calls using analysis tools and logs. | SYS-LO.01 |
| explain how interrupts work and their role in managing hardware communication. | SYS-LO.01 |
| explain the differences between physical and virtual memory and their importance in process isolation. | SYS-LO.01 |

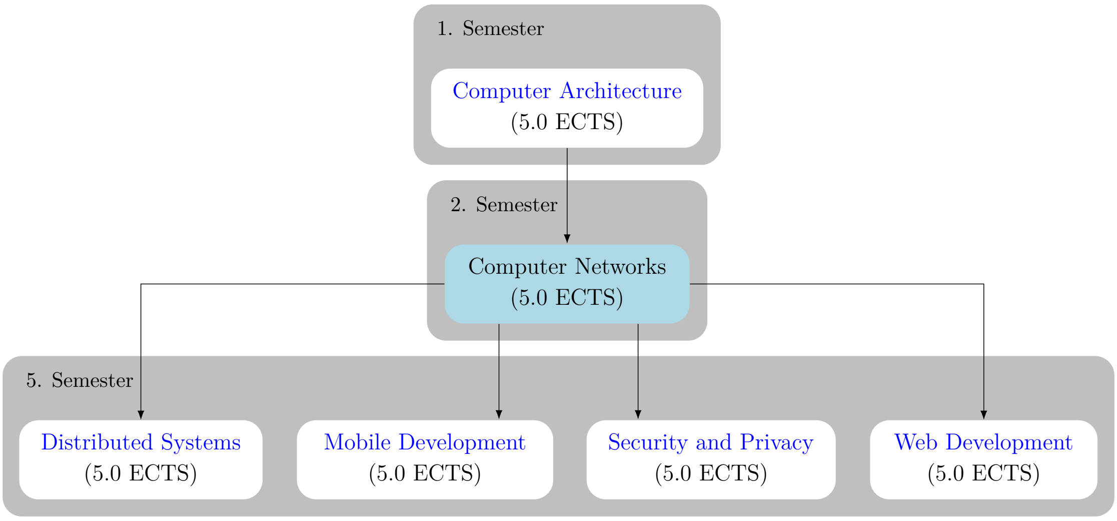
| | |
|---|---|
| identify and describe the causes of thrashing and strategies to prevent it. | SYS-LO.01 |
| describe how file systems and (memory-mapped) files are managed and apply file operations in real-world scenarios. | SYS-LO.01 |
| differentiate between processes and threads and their respective lifecycles. | SYS-LO.01 |
| explain process scheduling, context switching and interprocess communication strategies. | SYS-LO.01 |
| explain the role of stack and heap and their interrelationship with memory allocation and deallocation. | SYS-LO.02 |
| explain how threads and function calls are handled using stack and heap, as well as their impact on reliable software systems. | SYS-LO.02,<br>SYS-LO.08 |
| explain the difference between static and dynamic linking, as well as their advantages and disadvantages. | SYS-LO.02,<br>SYS-LO.08 |
| explain the role of an instruction set architecture in the program execution process. | SYS-LO.02 |
| explain how advanced translation concepts like virtual machines and cross compilation are integrated into to program execution process. | SYS-LO.02 |
| write programs to automate tasks, including file search, piping, system commands, environment variables, files and processes. | SYS-LO.01 |
| demonstrate meticulousness by carefully performing operations on operating system resources without damaging them. | LO.08 |

### 3.2.5 Computer Networks

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Organization of the internet
- Network components
- Naming and address schemes
- Local area network topologies
- Layers and the hourglass model
- Application layer protocols
- Switching and routing
- Internet protocol scalability

- Principles of wired communication
- Principles of wireless communication
- Principles of reliable communication
- Error, flow and congestion control
- Interactions with network APIs

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the structure of the internet and the communication between its network components. | SYS-LO.03 |
| set up, configure and analyze the traffic of a computer network using routers and switches for simulated real-world scenarios. | SYS-LO.03,<br>SYS-LO.04 |
| explain IP addressing and DNS resolution. | SYS-LO.03 |
| assign IP addresses within a subnet, and configure and troubleshoot basic DNS setups. | SYS-LO.04 |
| describe common LAN topologies and explain their advantages and disadvantages. | SYS-LO.03 |
| explain the OSI/hourglass model and trace network traffic at different layers using analysis tools. | SYS-LO.03,<br>SYS-LO.04 |
| explain common application layer protocols and their roles in enabling user services. | SYS-LO.03 |
| explain scalability challenges in IP-based networks and configure corresponding solution strategies, like IPv6 and NAT. | SYS-LO.03,<br>SYS-LO.04 |
| explain transmission of data over different media and its properties of communication, like bandwidth, latency and signal degradation. | SYS-LO.03 |
| explain the role of acknowledgments, timeouts, retransmissions and corresponding protocols in ensuring network reliability. | SYS-LO.03 |
| develop programs which make use of a network API to reliably solve problems in simulated real-world scenarios. | SYS-LO.04,<br>SYS-LO.06 |

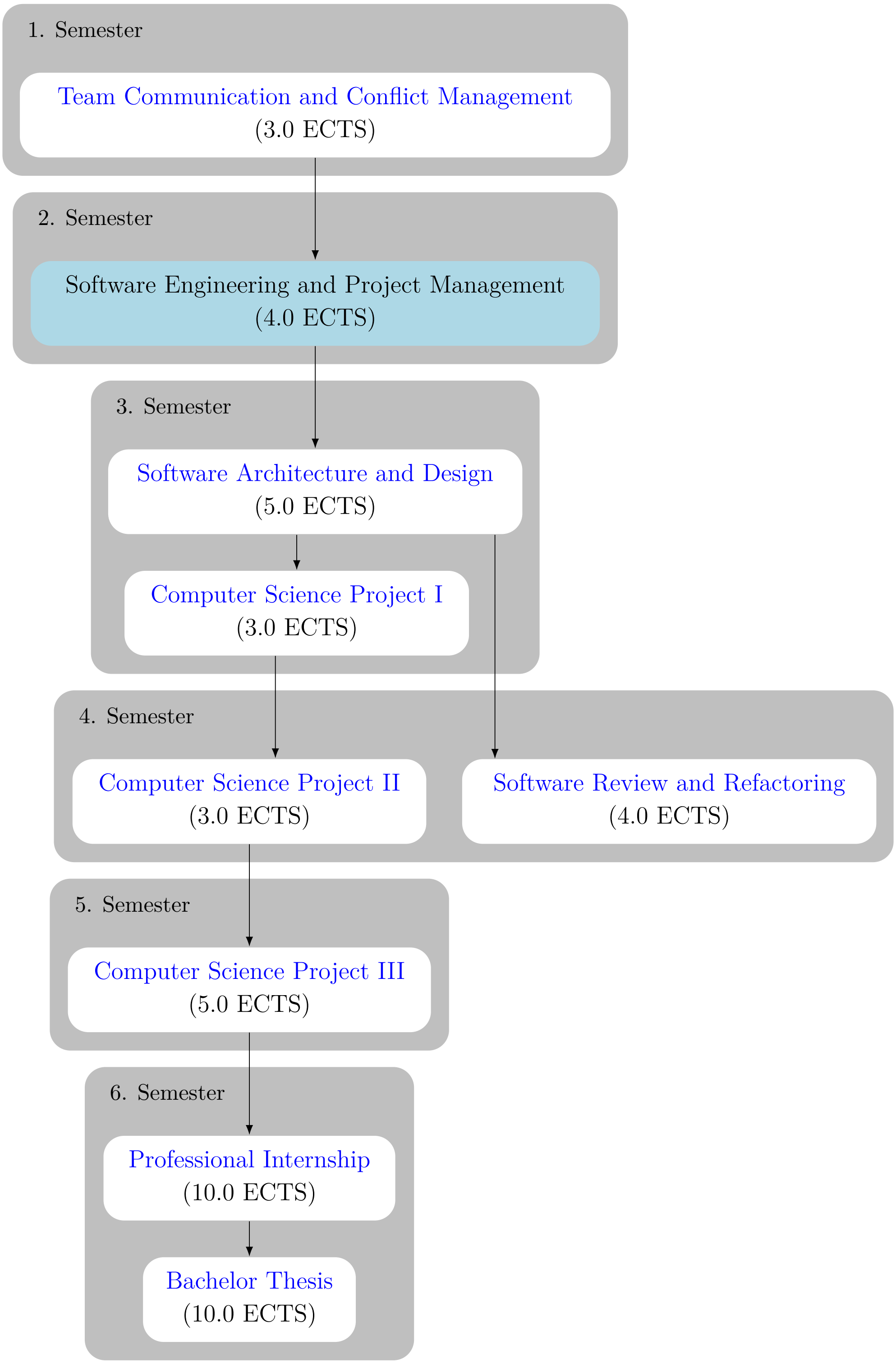
| | |
|---|---|
| demonstrate meticulousness by carefully configuring computer networks that perform reliably and securely. | LO.08 |

### 3.2.6 Software Engineering and Project Management

#### Course Profile

| | |
|---|---|
| Workload: | 4.0 ECTS |
| Teaching units: | 40 |
| Course type: | Integrated course |
| Assessment: | Continuous |

### Context and Dependencies


1. Semester
Team Communication and Conflict Management (3.0 ECTS)
2. Semester
Software Engineering and Project Management (4.0 ECTS)
3. Semester
Software Architecture and Design (5.0 ECTS)
Computer Science Project I (3.0 ECTS)
4. Semester
Computer Science Project II (3.0 ECTS)
Software Review and Refactoring (4.0 ECTS)
5. Semester
Computer Science Project III (5.0 ECTS)
6. Semester
Professional Internship (10.0 ECTS)
Bachelor Thesis (10.0 ECTS)

### Course Contents

- Orientation within the curriculum
- Definition and context of the term ”software engineering”
- Software development lifecycle
- Classic process models for software development
- Agile process models for software development
- Definition and context of the term ”project management”
- Roles and responsibilities in a software project team
- Opportunities and challenges of international teams
- Agile project management
- Context analysis
- Use case analysis
- Requirements elicitation and specification
- Requirements verification and validation

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| define the term ”software engineering” and distinguish it from general programming. | SWE-LO.01, SWE-LO.02 |
| describe the phases of the software development lifecycle and compare iterative and sequential approaches to software development. | SWE-LO.01 |
| describe the characteristics of classic process models for software development and analyze their advantages and limitations. | SWE-LO.01, SWE-LO.02 |
| describe the characteristics of agile process models for software development and analyze their advantages and limitations. | SWE-LO.01, SWE-LO.02 |

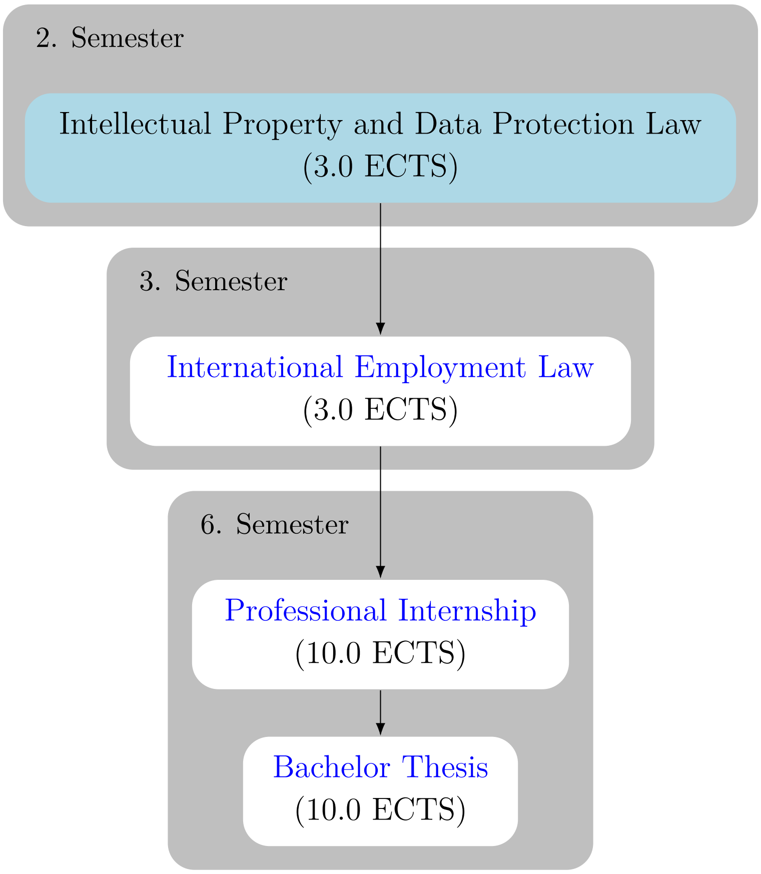
| | |
|---|---|
| compare agile process models to classical process models for software development. | SWE-LO.01 |
| apply agile software development practices to projects that resemble real-world scenarios. | SWE-LO.01 |
| define the term ”project management” and explain its relationship to software engineering. | SWE-LO.03 |
| identify the roles within (agile) software development teams and discuss the opportunities and challenges of distributed teams in terms of culture and collaboration. | SWE-LO.03 |
| analyze cultural, linguistic and organizational challenges for projects that resemble real-world scenarios. | SWE-LO.03 |
| propose strategies for effective communication and collaboration in international projects. | SWE-LO.03 |
| apply agile project management techniques, like sprint planning and retrospectives. | SWE-LO.01,<br>SWE-LO.03 |
| identify relevant stakeholders for projects that resemble real-world scenarios and document their requirements. | SWE-LO.01,<br>SWE-LO.03 |
| evaluate external factors that influence the scope and planning of projects that resemble real-world scenarios. | SWE-LO.01,<br>SWE-LO.03 |
| create use case diagrams and narratives to capture functional requirements. | SWE-LO.01,<br>SWE-LO.03 |
| gather requirements and write clear and testable software requirements specifications. | SWE-LO.01,<br>SWE-LO.03 |
| ensure that requirements are correct, complete and aligned with stakeholder needs. | SWE-LO.02,<br>SWE-LO.03 |
| describe the software development artifacts that are created and maintained in every software development phase. | SWE-LO.01 |
| create high-level software development plans for projects, identifying each phase and corresponding tasks. | SWE-LO.01,<br>SWE-LO.03 |
| demonstrate meticulousness by verifying the correctness and completeness of software engineering and project management artifacts. | LO.08 |

### 3.2.7 Intellectual Property and Data Protection Law

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 42 |
| Course type: | Lecture with exercises |
| Assessment: | Final |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Philosophical foundations of intellectual property
- Forms of intellectual property

- Laws, regulations and their applicability
- Intersection of international and national laws
- Treaties related to intellectual property and data protection
- Data ownership and custodianship
- Incident handling and breach disclosure
- Intellectual property rights of software
- Software licenses
- Limitations on copyright protections
- Enforcement of intellectual property rights
- Data aggregation concerns, AI-based aggregation
- Societal impacts on breakdowns in privacy
- Case studies and applications

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| evaluate philosophical justifications for intellectual property, including utilitarianism and economic incentives. | LAW-LO.01,<br>LAW-LO.02 |
| explain various forms of intellectual property, such as copyrights, patents and trademarks. | LAW-LO.01,<br>LAW-LO.02 |
| explain national and international laws governing intellectual property and data protection. | LAW-LO.01,<br>LAW-LO.02,<br>LAW-LO.03 |
| explain how companies apply intellectual property and data protection laws when operating across borders. | LAW-LO.01,<br>LAW-LO.02,<br>LAW-LO.03 |
| explain legal and ethical dimensions of data ownership, including the rights and responsibilities of data custodians. | LAW-LO.01,<br>LAW-LO.03 |

| | |
|---|---|
| explain legal requirements for handling data breaches and disclosing incidents. | LAW-LO.01,<br>LAW-LO.03 |
| explain intellectual property challenges specific to software, including patentability and the protection of algorithms. | LAW-LO.01,<br>LAW-LO.02 |
| evaluate the consequences of software licenses on software distribution, collaboration and intellectual property rights. | LAW-LO.02 |
| assess the legality of derivative works, reverse engineering and content reuse. | LAW-LO.02 |
| explain mechanisms of enforcing intellectual property rights, including litigation, cease-and-desist letters and digital enforcement tools. | LAW-LO.01 |
| explain the balance between innovation in artificial intelligence and the need for stringent data privacy protections. | LAW-LO.01,<br>LAW-LO.03,<br>LAW-LO.04 |
| assess societal consequences of weakening privacy protections, including surveillance, trust erosion and inequality. | LAW-LO.01,<br>LAW-LO.03,<br>LAW-LO.07 |
| evaluate how issues regarding intellectual property and data protection are treated in real-world scenarios. | LAW-LO.02,<br>LAW-LO.03 |

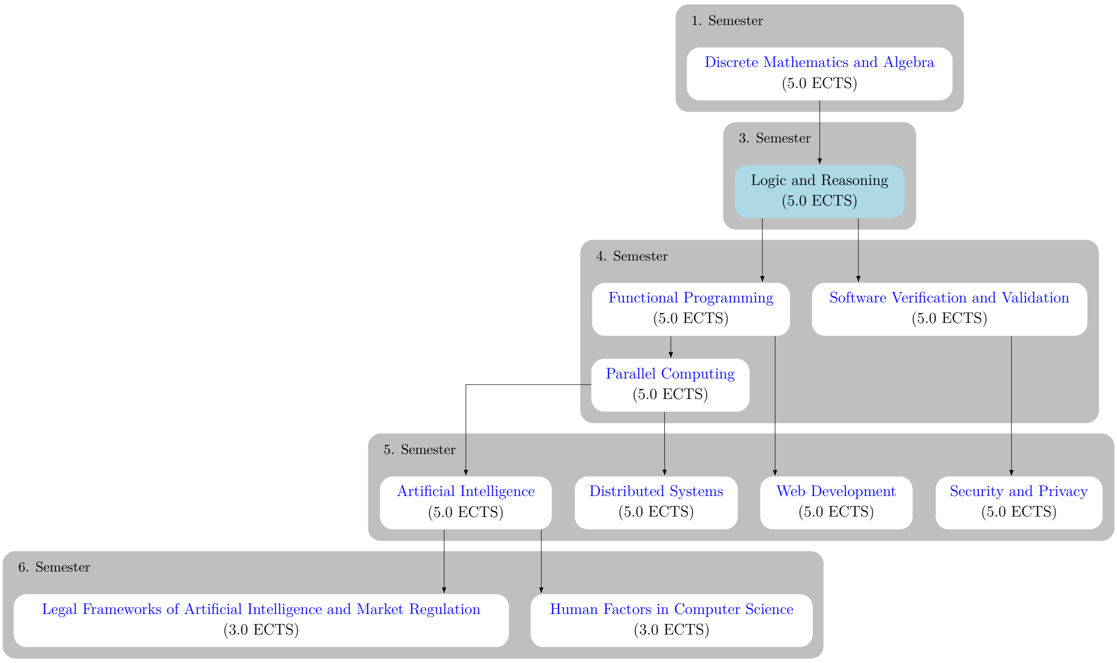
## 3.3 3. Semester

### 3.3.1 Logic and Reasoning

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Propositional logic
- Predicate logic
- Horn clauses
- Proof techniques

- Inference rules
- Resolution
- Higher-order logic
- Non-monotonic logic
- Logic programming

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| construct and interpret truth tables for logical formulas. | CSF-LO.01 |
| formulate logical arguments of real-world scenarios and prove their validity using propositional logic. | CSF-LO.01 |
| explain the syntax and semantics of predicate logic, including the use of quantifiers. | CSF-LO.01 |
| translate natural language statements of real-world scenarios into predicate logic expressions and evaluate them. | CSF-LO.01 |
| explain the interrelationship between Horn clauses and logic programming. | CSF-LO.01 |
| explain the resolution method and perform rule-based reasoning using Horn clauses. | CSF-LO.01 |
| explain various proof techniques, including direct proofs, proof by contradiction and mathematical induction. | CSF-LO.01 |
| analyze the correctness of logical arguments using proof techniques. | CSF-LO.01 |
| derive consequences from premises using a set of inference rules in real-world scenarios. | CSF-LO.01 |
| explain the applications of higher-order logic in computer science, especially in program verification and formal methods. | CSF-LO.01 |
| explain where non-monotonic reasoning is essential, such as in belief revision and default logic. | CSF-LO.01 |

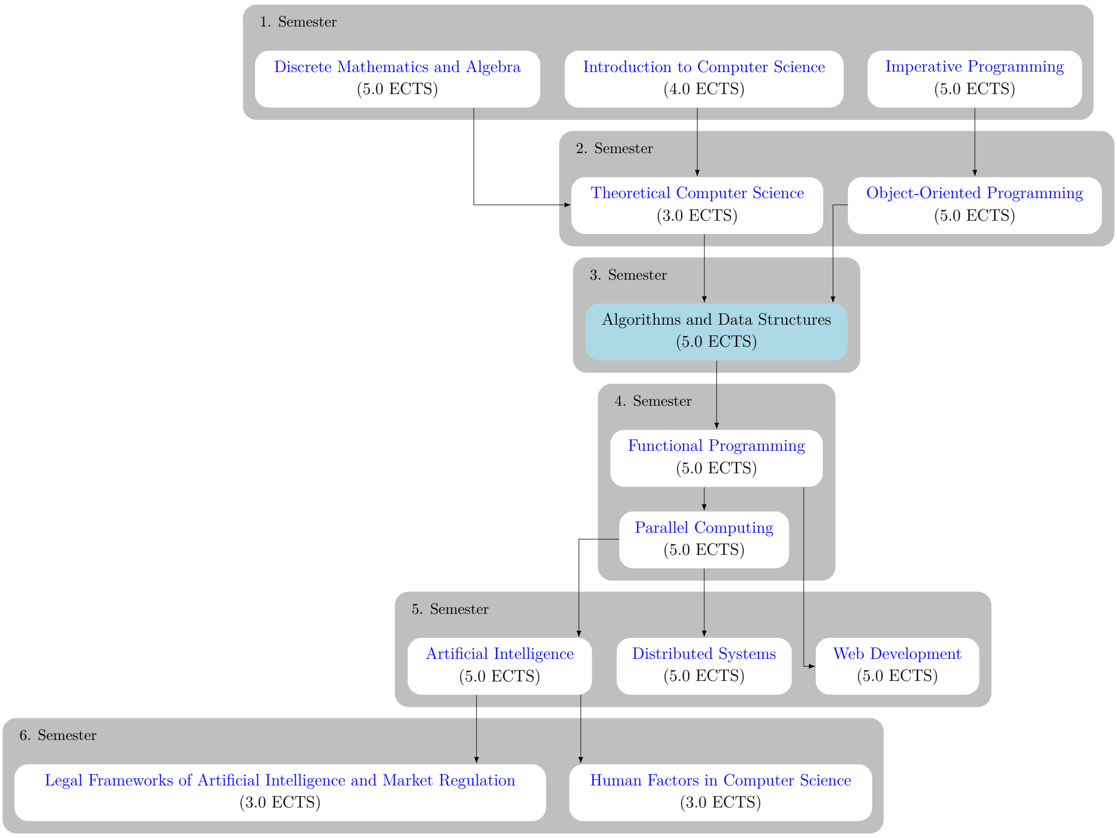
| | |
|---|---|
| apply the gained knowledge from the course to practical examples using a logic programming language and understand the syntax and semantics of the used programming language. | CSF-LO.01,<br>CSF-LO.04 |
| demonstrate meticulousness by verifying the correctness and completeness of formulas and inferences. | LO.08 |

### 3.3.2 Algorithms and Data Structures

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Algorithm design paradigms
- Graphs and graph algorithms
- Trees and tree algorithms

- Arrays
- Lists
- Heaps
- Stacks
- Queues
- Maps and sets
- Sorting algorithms
- Searching algorithms
- Hashing algorithms
- String processing
- Recursion and tail recursion
- Time and space complexity and analysis
- Selection of data structures and algorithms
- Storing and loading data structures
- Immutability and thread safety

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the different algorithmic paradigms. | PRO-LO.01 |
| explain and implement common data structures, both mutable and immutable. | PRO-LO.01,<br>PRO-LO.02,<br>PRO-LO.07 |
| explain and implement common searching, sorting and hashing algorithms. | PRO-LO.01,<br>PRO-LO.02 |
| explain and empirically evaluate the advantages and disadvantages of data structures and algorithms. | PRO-LO.05 |

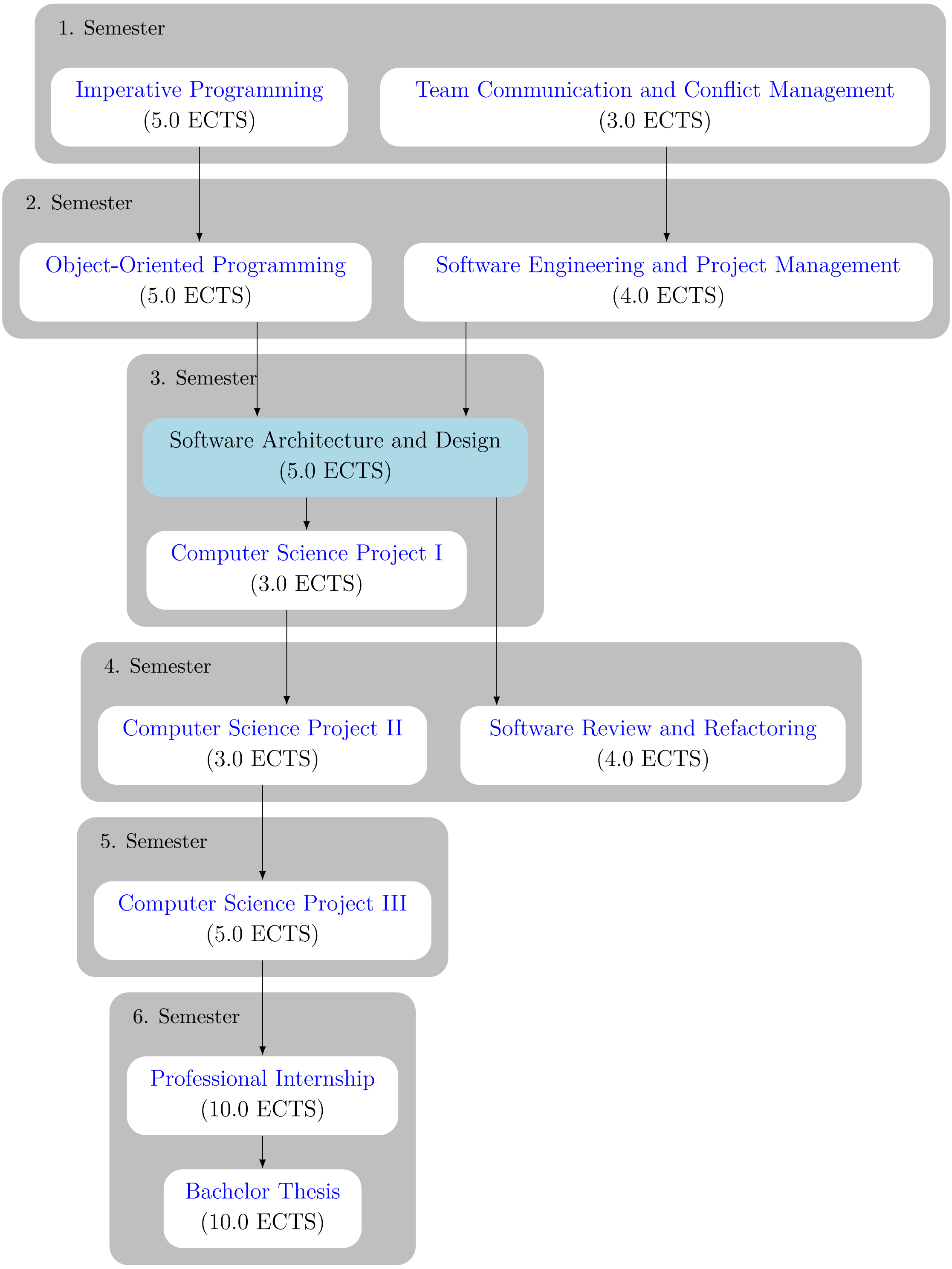
| | |
|---|---|
| estimate and measure the space and time complexity of algorithms. | PRO-LO.05 |
| systematically decide and justify which data structures and algorithms are appropriate for given real-world scenarios. | PRO-LO.03,<br>PRO-LO.05 |
| explain the advantages and disadvantages of iterative and recursive implementation strategies. | PRO-LO.01,<br>PRO-LO.05 |
| develop a recursive solution strategy for given real-world scenarios. | PRO-LO.01,<br>PRO-LO.03 |
| transform informal task specifications into formal data structures and algorithms using concise coding styles. | PRO-LO.01,<br>PRO-LO.03,<br>PRO-LO.07 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions. | LO.08 |
| demonstrate persistence and internalize that problem-solving is an iterative process of overcoming different forms of failures. | LO.08 |
| demonstrate self-directedness and internalize that informal task specifications are never complete. | LO.08 |

### 3.3.3 Software Architecture and Design

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

### Context and Dependencies


1. Semester
Imperative Programming
(5.0 ECTS)
Team Communication and Conflict Management
(3.0 ECTS)
2. Semester
Object-Oriented Programming
(5.0 ECTS)
Software Engineering and Project Management
(4.0 ECTS)
3. Semester
Software Architecture and Design
(5.0 ECTS)
Computer Science Project I
(3.0 ECTS)
4. Semester
Computer Science Project II
(3.0 ECTS)
Software Review and Refactoring
(4.0 ECTS)
5. Semester
Computer Science Project III
(5.0 ECTS)
6. Semester
Professional Internship
(10.0 ECTS)
Bachelor Thesis
(10.0 ECTS)

### Course Contents

- Orientation within the curriculum
- Definition and context of the term ”software architecture”
- Definition and context of the term ”software design”
- Quality attributes of software architectures
- Documentation and communication of software architectures
- Software architecture and design principles
- Software modeling and modeling notations
- Architectural styles and patterns
- Service-oriented architectures
- Event-driven architectures
- Design patterns
- Model-driven software development
- Domain-specific languages

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| define the term ”software architecture” and explain the areas of responsibility of a software architect. | SWE-LO.01,<br>SWE-LO.03 |
| define the term ”software design” and explain how it complements software architecture. | SWE-LO.01,<br>SWE-LO.03 |
| explain the impact of software architecture and design on the software development process. | SWE-LO.02 |
| explain various quality attributes of software architectures and analyze trade-offs between them in architectural decisions. | SWE-LO.02 |
| design, document and communicate software architectures. | SWE-LO.03,<br>SWE-LO.04 |

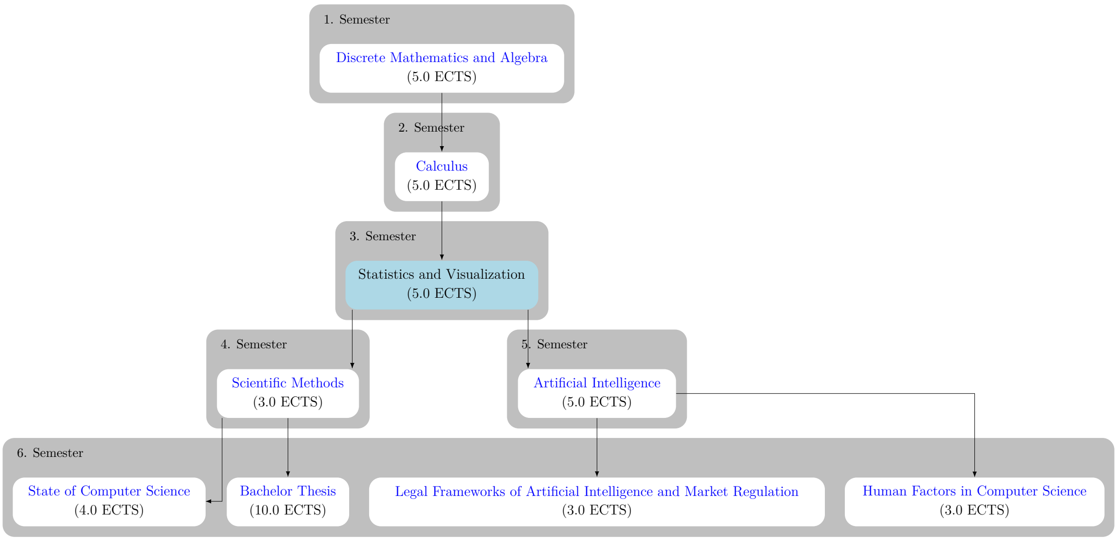
| | |
|---|---|
| critically evaluate software systems against software architecture and design principles, like abstraction, modularity, reusability, separation of concerns, maintainability and extensibility. | SWE-LO.02 |
| model software solutions that respect software architecture and design principles, like abstraction, modularity, reusability, separation of concerns, maintainability and extensibility. | SWE-LO.04,<br>SWE-LO.05 |
| use modeling notations like UML or SysML to represent software architecture and design views. | SWE-LO.04 |
| explain and interpret software architecture and design models, and translate them into executable code. | SWE-LO.04 |
| explain the properties of common architectural styles, architectural patterns and design patterns. | SWE-LO.04 |
| systematically select architectural styles, architectural patterns and design patterns according to given software requirements. | SWE-LO.03,<br>SWE-LO.04 |
| design systems with future enhancements in mind, using strategies like plugin architectures. | SWE-LO.05 |
| explain the fundamentals, challenges and technical details of service-oriented architectures. | SWE-LO.04 |
| design services based on service-oriented architecture principles. | SWE-LO.04,<br>SWE-LO.05 |
| design software systems based on event-driven architectural patterns. | SWE-LO.04,<br>SWE-LO.05 |
| explain the utility and the technical details of model-driven software development. | SWE-LO.04 |
| explain the impact of (meta-)modeling on the software development process. | SWE-LO.03,<br>SWE-LO.04 |
| develop textual and graphical domain-specific modeling languages. | SWE-LO.03,<br>SWE-LO.04 |
| demonstrate meticulousness by verifying the correctness and completeness of software architecture and design artifacts. | LO.08 |

### 3.3.4 Statistics and Visualization

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Probability spaces
- Random variables
- Probability distributions
- Descriptive statistics
- Inferential statistics
- Confidence intervals
- Correlation and regression
- Bayesian reasoning

- Effective and misleading visualizations
- Grammar of graphics
- Data sources and their access
- Types of data and their encoding
- Exploratory data analysis
- Data preprocessing
- Statistical visualizations, charts, graphs
- Analysis and visualization tools

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the impact of statistics on computer science. | INT-LO.05 |
| explain the process of data analysis, including data collection, cleaning, analysis and interpretation. | INT-LO.05 |
| calculate probabilities for events using the rules of probability. | INT-LO.07 |
| apply counting principles, such as permutations and combinations, to calculate the number of outcomes in real-world scenarios. | INT-LO.07 |
| compute and interpret the probability distribution, expectation and variance of random variables. | INT-LO.05,<br>INT-LO.07 |
| identify and apply common probability distributions. | INT-LO.07 |
| explain the central limit theorem and its implications for large sample distributions. | INT-LO.07 |
| formulate and test hypotheses using statistical methods. | INT-LO.07 |
| explain the concepts of Type I and Type II errors and their impact on hypothesis testing. | INT-LO.07 |
| compute confidence intervals, interpret them and apply them in decision-making processes in real-world scenarios. | INT-LO.07 |

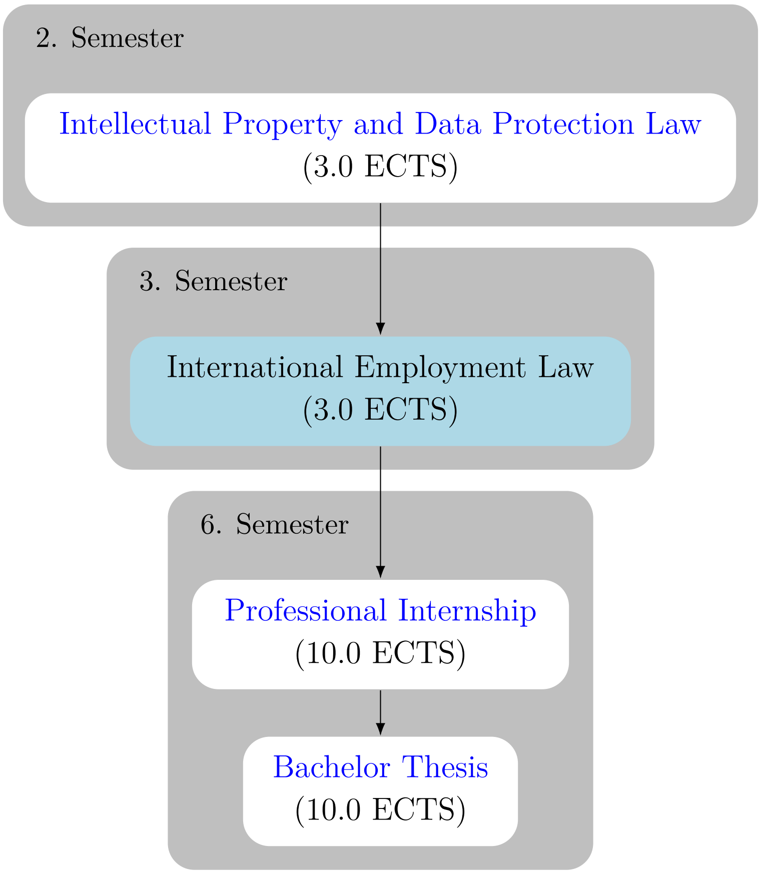
| | |
|---|---|
| explain the concepts of correlation and causation and how they differ. | INT-LO.07 |
| apply Bayes' theorem to update probabilities based on new evidence. | INT-LO.05,<br>INT-LO.07 |
| explain the importance of data visualization in making data understandable and actionable. | INT-LO.07 |
| visualize data through plots, interpret statistical results and compare datasets using summary statistics. | INT-LO.07 |
| design visualizations that follow best practices for clarity, readability and insightfulness. | INT-LO.07 |
| recognize common pitfalls and strategies that can make visualizations misleading. | INT-LO.07 |
| explain the theoretical framework for the systematic construction of visualizations. | INT-LO.07 |
| find, access and import data from multiple sources for analysis and visualization. | INT-LO.07 |
| uncover patterns, insights, trends, correlations and anomalies in raw datasets, and set corrective actions if needed. | INT-LO.05,<br>INT-LO.07 |
| choose and apply the correct analysis and visualization techniques for given data of real-world scenarios. | INT-LO.07 |
| demonstrate inventiveness by applying existing analysis and visualization strategies to new contexts. | LO.08 |
| demonstrate meticulousness by verifying the correctness and completeness of analyses and visualizations. | LO.08 |

### 3.3.5 International Employment Law

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 42 |
| Course type: | Lecture with exercises |
| Assessment: | Final |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Introduction to the global job market
- International employment regulations

- Work permits, visas and immigration laws
- Rights and duties of employees and employers
- Self-employment and freelancing across borders
- Social security, insurance and pensions
- Career planning and job search
- Selected legal fields in context to the other course content
- Case studies and applications

### Learning Outcomes

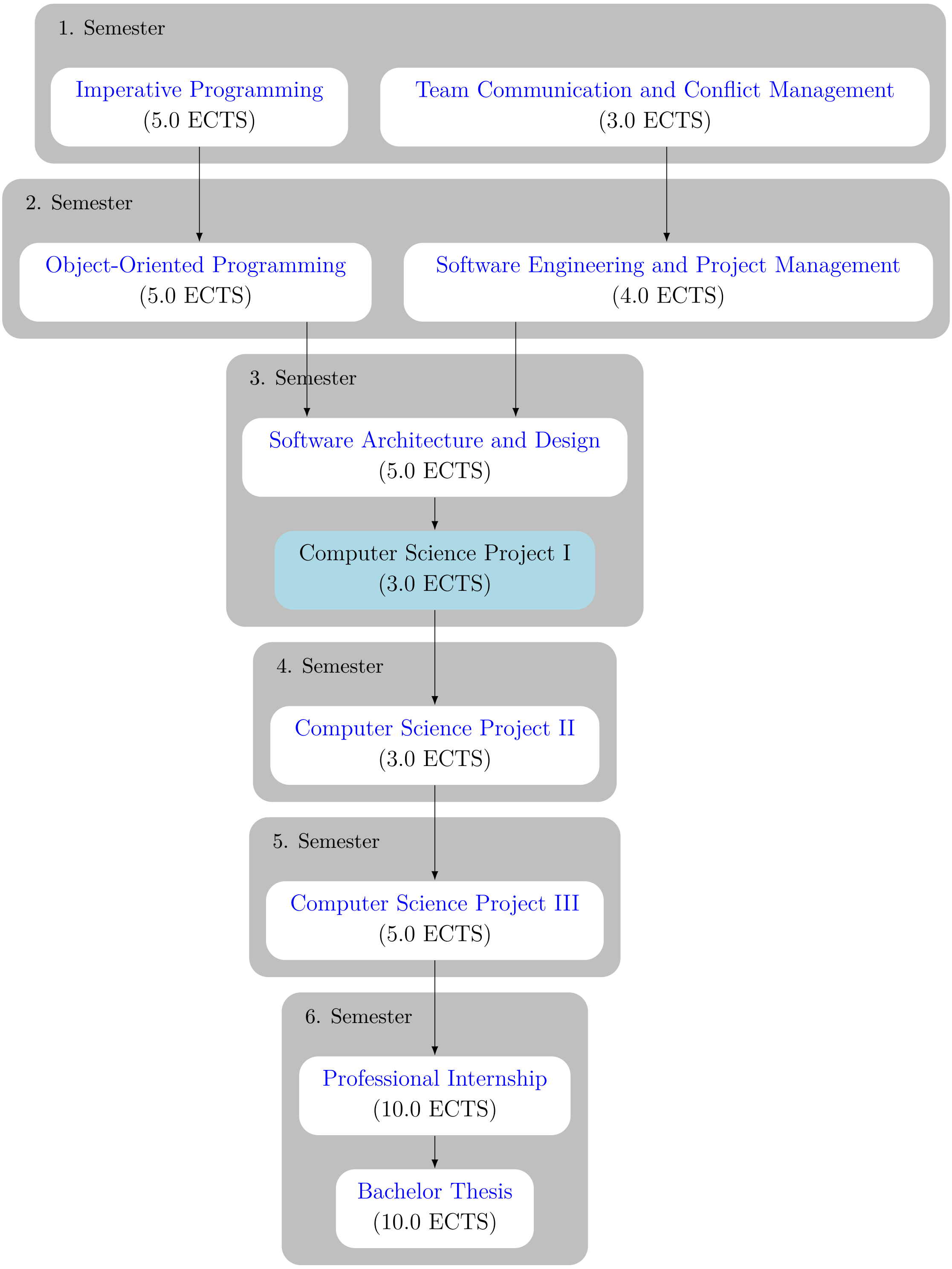
| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain international and national regulations concerning the job market. | LAW-LO.01,<br>LAW-LO.06 |
| identify the rights and responsibilities of employees and employers in selected global contexts. | LAW-LO.06 |
| explain regulations and simulate processes involved in self-employment and freelance work. | LAW-LO.06 |
| explain visa regulations, work permits and residency requirements. | LAW-LO.06 |
| develop strategies for compliance with labor laws and employer expectations across borders. | LAW-LO.06 |
| identify cross-connections to other areas of law, such as corporate law, discuss selected legal issues and apply them to real-world scenarios. | LAW-LO.06,<br>LAW-LO.07 |

### 3.3.6 Computer Science Project I

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 30 |
| Course type: | Project |
| Assessment: | Continuous |

### Context and Dependencies



## Course Contents

- Orientation within the curriculum
- Independent execution of a project
- Presentation of results

## Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| apply the acquired technical and social skills to complete practical projects that are related to real-world scenarios. | LO.08, SIP-LO.01, SIP-LO.02, SIP-LO.03 |
| occupy a specific role within a team and fulfil the tasks of the role in a goal-driven manner in order to complete a project. | LO.08, SIP-LO.01, SIP-LO.02, SIP-LO.03 |
| execute, document and justify project-related architecture, design and implementation decisions. | SIP-LO.02 |
| present project results and discuss them with peers and professionals, like faculty members. | LO.08, SIP-LO.01, SIP-LO.02 |
| demonstrate adaptability by satisfying project goals which may be continually evolving. | LO.08, SIP-LO.02 |
| demonstrate willingness for collaboration to advance a project as a team. | LO.08, SIP-LO.01, SIP-LO.03 |
| demonstrate inventiveness by applying existing solution strategies to a new project. | LO.08, SIP-LO.02 |
| demonstrate persistence by overcoming different forms of failures during a project. | LO.08, SIP-LO.01 |
| demonstrate proactiveness by anticipating issues regarding the progression of a project. | LO.08 |

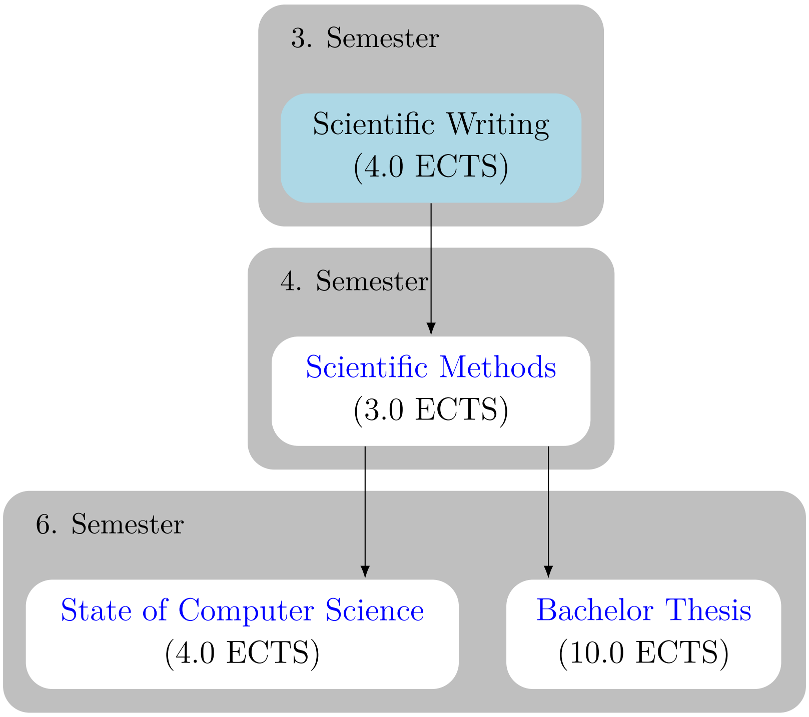
| | |
|---|---|
| demonstrate responsibility for all aspects of the solution developed during a project. | LO.08 |
| demonstrate self-directedness and internalize that goal definitions of a project are never complete. | LO.08 |

### 3.3.7 Scientific Writing

#### Course Profile

| | |
|---|---|
| Workload: | 4.0 ECTS |
| Teaching units: | 40 |
| Course type: | Seminar |
| Assessment: | Continuous |

#### Context and Dependencies


3. Semester
Scientific Writing (4.0 ECTS)
4. Semester
Scientific Methods (3.0 ECTS)
6. Semester
State of Computer Science (4.0 ECTS)
Bachelor Thesis (10.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Role and impact of scientific publications
- Process of performing scientific research
- Process of publishing scientific results
- Citing, plagiarism and academic integrity
- Search in scientific databases

- Structure of scientific documents
- Typesetting scientific documents
- Structure of a research proposal

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to … | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain how scientific publications are created, indexed and distributed. | SCI-LO.04 |
| explain the structure and expectations of a scientific document. | SCI-LO.01,<br>SCI-LO.03,<br>SCI-LO.04 |
| write a document that adheres to scientific standards using an established typesetting system. | SCI-LO.03,<br>SCI-LO.04 |
| integrate visual elements like figures, tables and equations into a scientific document, as well as their corresponding indexes. | SCI-LO.03,<br>SCI-LO.04 |
| integrate bibliography and citations into a scientific document. | SCI-LO.03,<br>SCI-LO.04 |
| write a research proposal for the upcoming bachelor thesis. | SCI-LO.01,<br>SCI-LO.03 |
| demonstrate meticulousness by using precise language when creating a scientific document. | LO.08 |

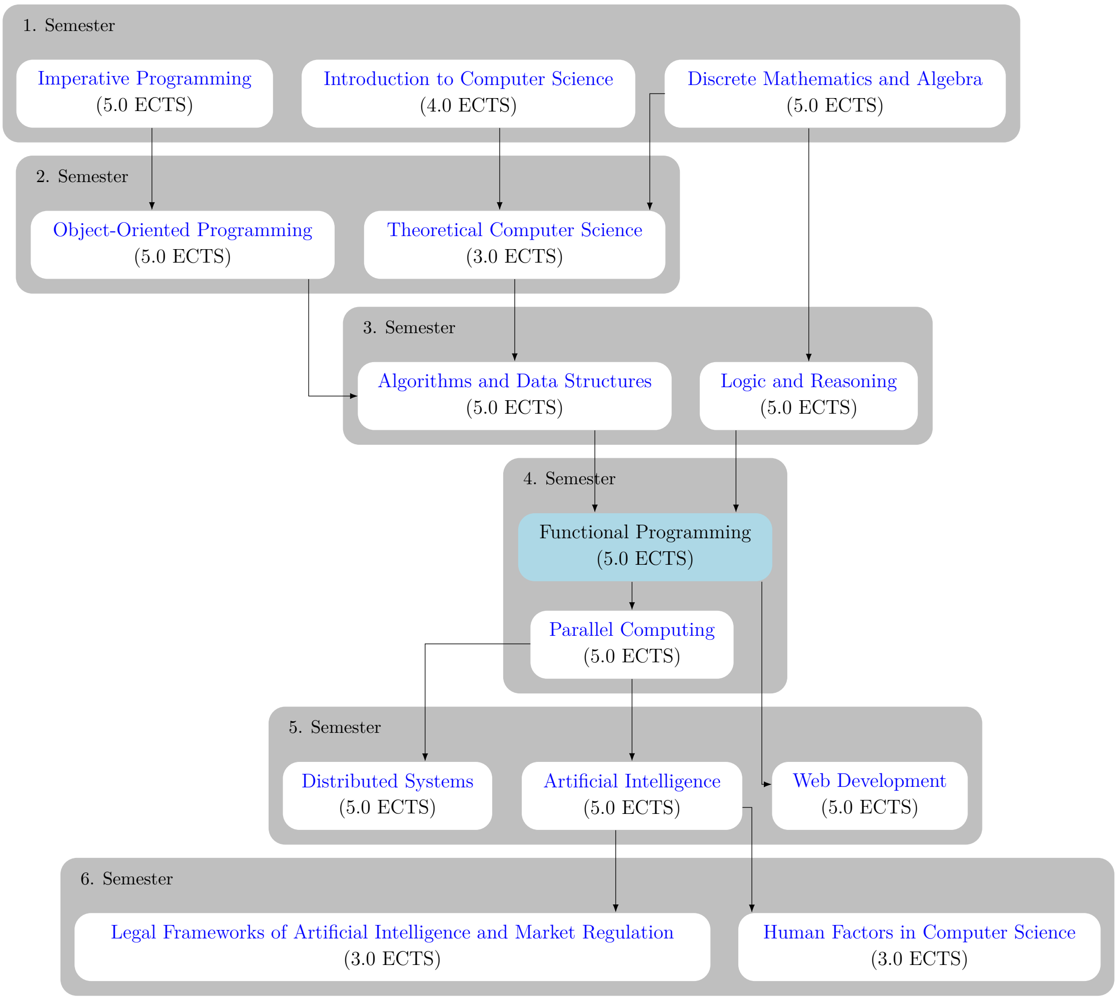
# 3.4 4. Semester

## 3.4.1 Functional Programming

### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Theoretical Computer Science (3.0 ECTS)
3. Semester
Algorithms and Data Structures (5.0 ECTS)
Logic and Reasoning (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Distributed Systems (5.0 ECTS)
Artificial Intelligence (5.0 ECTS)
Web Development (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)

### Course Contents

- Orientation within the curriculum
- Principles of functional programming and its contributions to reliability
- Relationship between the lambda calculus and functional programming
- Syntax and semantics of the selected programming language
- Immutability
- Lazy evaluation and Weak Head Normal Form
- Program structure using modules, types and functions
- Debugging and exception handling in contrast to imperative programming
- Data types and pattern matching
- Lists and list comprehensions
- Functions and recursion
- Lambda expressions, higher-order functions and compositions
- Common operations like map, filter and fold as well as their variations
- Currying and partial application
- Type classes
- Parametric polymorphism
- Side effect handling
- Functional design and the expression problem

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to … | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the differences between the functional and other programming paradigms. | PRO-LO.01 |

| | |
|---|---|
| explain the role of functional programming in the context of reliable software systems. | PRO-LO.01, PRO-LO.07 |
| explain the syntax and semantics of the used functional programming language. | PRO-LO.01 |
| explain and systematically apply functional programming principles like higher-order functions, lambda expressions and pattern matching to real-world scenarios. | PRO-LO.01, PRO-LO.03 |
| explain the advantages and disadvantages of lazy evaluation and apply the concept to real-world scenarios. | PRO-LO.01, PRO-LO.03 |
| explain and apply exception handling mechanisms using functional programming techniques. | PRO-LO.01, PRO-LO.03, PRO-LO.07 |
| explain and systematically apply functional abstractions like parametric polymorphism and type classes, including functors and monads, to real-world scenarios. | PRO-LO.01, PRO-LO.03, PRO-LO.07 |
| apply functional side effect handling and explain the differences with respect to other programming paradigms. | PRO-LO.01, PRO-LO.03, PRO-LO.07 |
| explain the concept of immutability and apply associated data structures and algorithms to real-world scenarios. | PRO-LO.01, PRO-LO.02, PRO-LO.03, PRO-LO.07 |
| design and use types and functions which adequately capture the properties of a real-world domain and prevent the misuse of their corresponding values. | PRO-LO.01, PRO-LO.03, PRO-LO.07 |
| explain and demonstrate the expression problem in the context of the functional programming paradigm. | PRO-LO.01, PRO-LO.03, PRO-LO.07 |
| transform informal task specifications into formal functional designs and implementations using concise coding styles. | PRO-LO.01, PRO-LO.03, PRO-LO.07 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions. | LO.08 |
| demonstrate persistence and internalize that problem-solving is an iterative process of overcoming different forms of failures. | LO.08 |

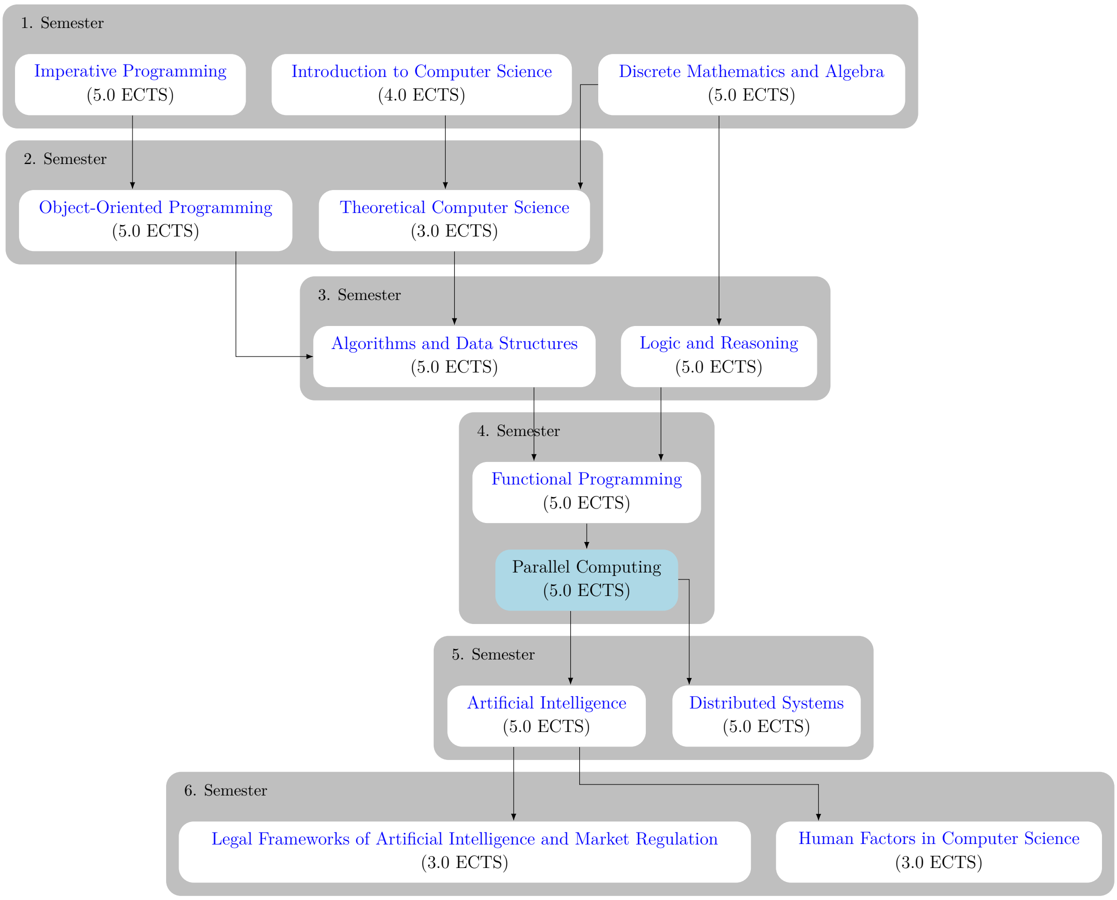
| | |
|---|---|
| demonstrate self-directedness and internalize that informal task specifications are never complete. | LO.08 |

### 3.4.2 Parallel Computing

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Theoretical Computer Science (3.0 ECTS)
3. Semester
Algorithms and Data Structures (5.0 ECTS)
Logic and Reasoning (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
Distributed Systems (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Parallelism and concurrency

- Shared memory and message-passing
- Amdahl's Law
- Scalability, granularity, communication overhead
- Data and task dependencies
- Races, locks, starvation and their prevention
- Parallel programming patterns
- Concurrent programming concepts
- Initiation and termination of parallel operations
- Determinism and non-determinism
- GPU programming
- Performance evaluation

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the challenges, opportunities and common application domains of parallelism and concurrency. | LO.06 |
| explain the impact of parallel computing on the reliability of software systems. | LO.06 |
| explain the difference between parallelism and concurrency. | PRO-LO.01,<br>PRO-LO.06 |
| explain the difference between shared memory and message-passing concurrency. | PRO-LO.01,<br>PRO-LO.06 |
| explain and apply Amdahl's Law. | PRO-LO.01,<br>PRO-LO.06 |
| explain parallel programming patterns and apply selected patterns to real-world scenarios in a reliable way. | PRO-LO.01,<br>PRO-LO.06,<br>PRO-LO.07 |

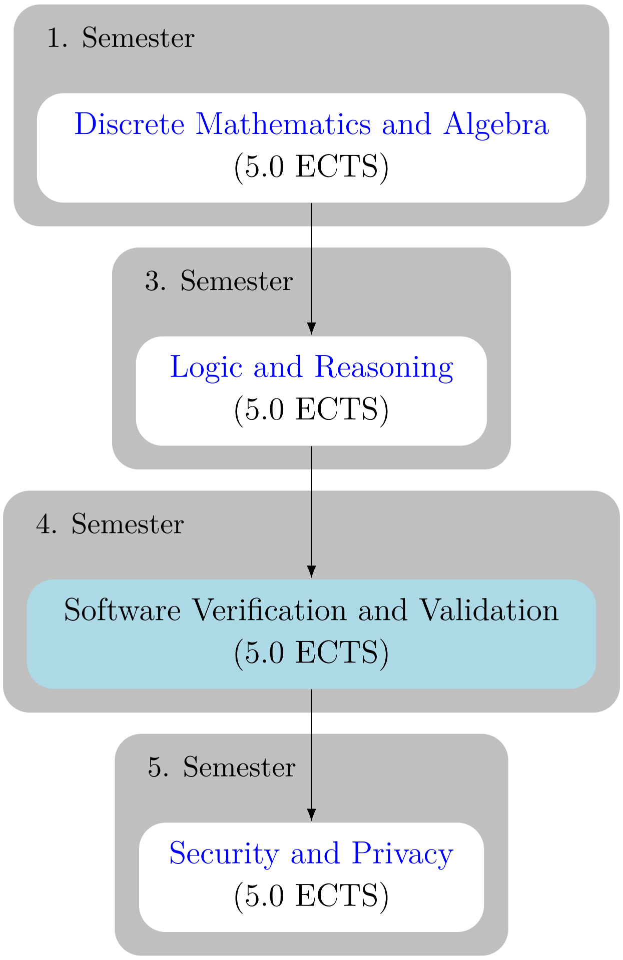
| | |
|---|---|
| explain concurrent programming concepts and apply selected concepts to real-world scenarios in a reliable way. | PRO-LO.01,<br>PRO-LO.06,<br>PRO-LO.07 |
| explain and apply the necessary CPU-GPU interactions for performing general-purpose operations on the GPU in a reliable way. | PRO-LO.01,<br>PRO-LO.06,<br>PRO-LO.07 |
| empirically evaluate the speedup and slowdowns of parallel and concurrent implementations and configurations. | PRO-LO.05 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions. | LO.08 |
| demonstrate persistence and internalize that problem-solving is an iterative process of overcoming different forms of failures. | LO.08 |

### 3.4.3 Software Verification and Validation

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum

- Verification and validation in the development lifecycle
- Static and dynamic software verification
- Errors prevented by static and dynamic verification
- Bugs, faults and failures
- Test-driven development
- Testing objectives and kinds
- Opaque-box and transparent-box testing
- Property-based testing
- Role of functional programming for software verification
- Principles of types and type safety
- Strong, weak, static and dynamic typing
- Type checking and type inference
- Principles of substructural type systems
- Principles of refinement types

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain how software verification and validation are integrated throughout the software development lifecycle. | SWE-LO.01 |
| explain the properties, advantages and disadvantages of static and dynamic software verification, as well as their differences. | SWE-LO.02, SWE-LO.06 |
| identify the types of errors that can be detected by static and dynamic software verification approaches. | SWE-LO.02, SWE-LO.06 |
| explain how test-driven development leads to more reliable software systems. | SWE-LO.01, SWE-LO.02 |

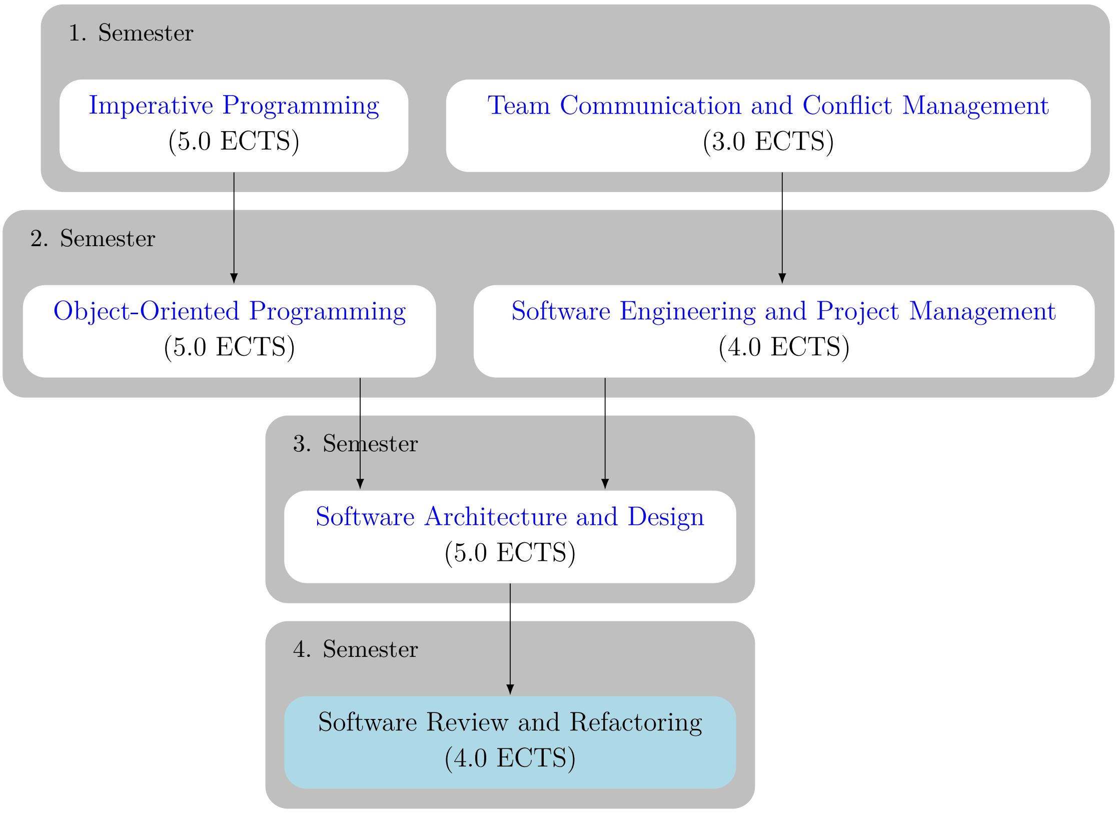
| | |
|---|---|
| develop and execute opaque-box, transparent-box and property-based tests for software that resemble real-world scenarios. | SWE-LO.06 |
| apply functional programming techniques which demonstrate a positive impact on software verification, like immutability and pure functions. | SWE-LO.06 |
| enforce correctness and prevent errors in software systems using advanced type systems. | SWE-LO.02,<br>SWE-LO.07 |
| demonstrate the impact of using imprecise types on software quality and explain the engineering trade-offs with respect to type precision. | SWE-LO.02,<br>SWE-LO.07 |
| give a high-level explanation of substructural type systems and their impact on managing resources and runtime. | SWE-LO.02,<br>SWE-LO.07 |
| utilize a programming language with a substructural type system to write programs that manage resources without leaks. | SWE-LO.02,<br>SWE-LO.07 |
| apply refinement types to real-world scenarios to enforce more precise conditions within types. | SWE-LO.02,<br>SWE-LO.07 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions. | LO.08 |

### 3.4.4 Software Review and Refactoring

#### Course Profile

| | |
|---|---|
| Workload: | 4.0 ECTS |
| Teaching units: | 40 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Team Communication and Conflict Management (3.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Software Engineering and Project Management (4.0 ECTS)
3. Semester
Software Architecture and Design (5.0 ECTS)
4. Semester
Software Review and Refactoring (4.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Clean code principles
- Refactoring and rewriting
- Anti-patterns and code smells

- Defensive programming
- Root cause analysis
- Software documentation
- Style guides, commenting, naming
- Pre-conditions, post-conditions, invariants
- Mutability and immutability
- Side-effects
- Over-design and over-engineering
- Configuration management and reproducibility
- Reliability metrics in software development
- Continuous integration and delivery

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| discuss and improve development artifacts of real-world scenarios. | SWE-LO.02,<br>SWE-LO.06 |
| systematically apply clean code principles to real-world scenarios. | SWE-LO.02,<br>SWE-LO.06 |
| systematically refactor existing development artifacts by applying proven patterns or rewriting them using different technologies. | SWE-LO.02,<br>SWE-LO.05 |
| simplify complex systems by designing appropriate layers of abstraction. | SWE-LO.02,<br>SWE-LO.05 |
| break down monolithic systems into modular components. | SWE-LO.02,<br>SWE-LO.05 |
| systematically analyze problems of existing software solutions and find the root cause of problems. | SWE-LO.02,<br>SWE-LO.06 |

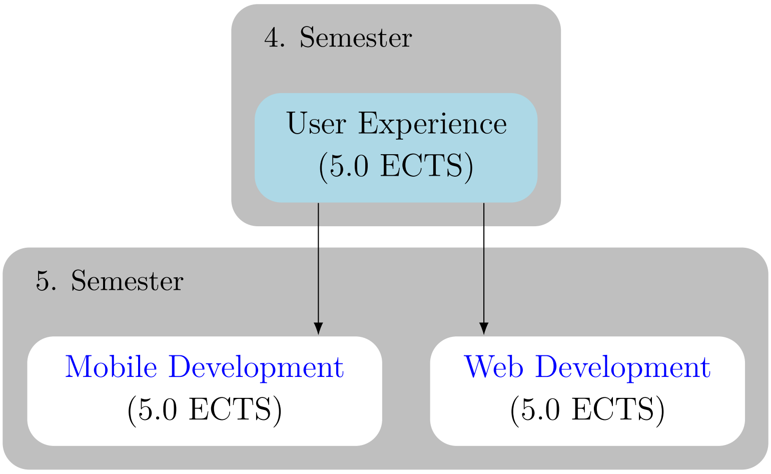
| | |
|---|---|
| detect anti-patterns and correct them accordingly. | SWE-LO.02,<br>SWE-LO.06 |
| document software development artifacts using established guidelines. | SWE-LO.02,<br>SWE-LO.05 |
| calculate reliability metrics for real-world scenarios, like defect density, failure rate and mean time to failure. | SWE-LO.02 |
| explain the properties and importance of reproducibility, version control and release management in maintaining software quality. | SWE-LO.02,<br>SWE-LO.08 |
| utilize a version control system to manage codebases, including branching, merging, conflict resolution, pull requests and reviews. | SWE-LO.08 |
| apply the principles of continuous integration and delivery to projects that resemble real-world scenarios. | SWE-LO.08 |
| demonstrate meticulousness by verifying the correctness and completeness of refactored development artifacts. | LO.08 |

### 3.4.5 User Experience

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



#### Course Contents

- Orientation within the curriculum
- Forms of graphic output
- Demographics and populations regarding disabilities
- Interaction techniques
- User interface design constraints and evaluation
- Human vision system
- Color theory, models and histograms
- Standard image formats
- User-centered design methodology
- Inclusive/universal design processes

- Accessibility standards
- Legal obligations regarding accessibility

### Learning Outcomes

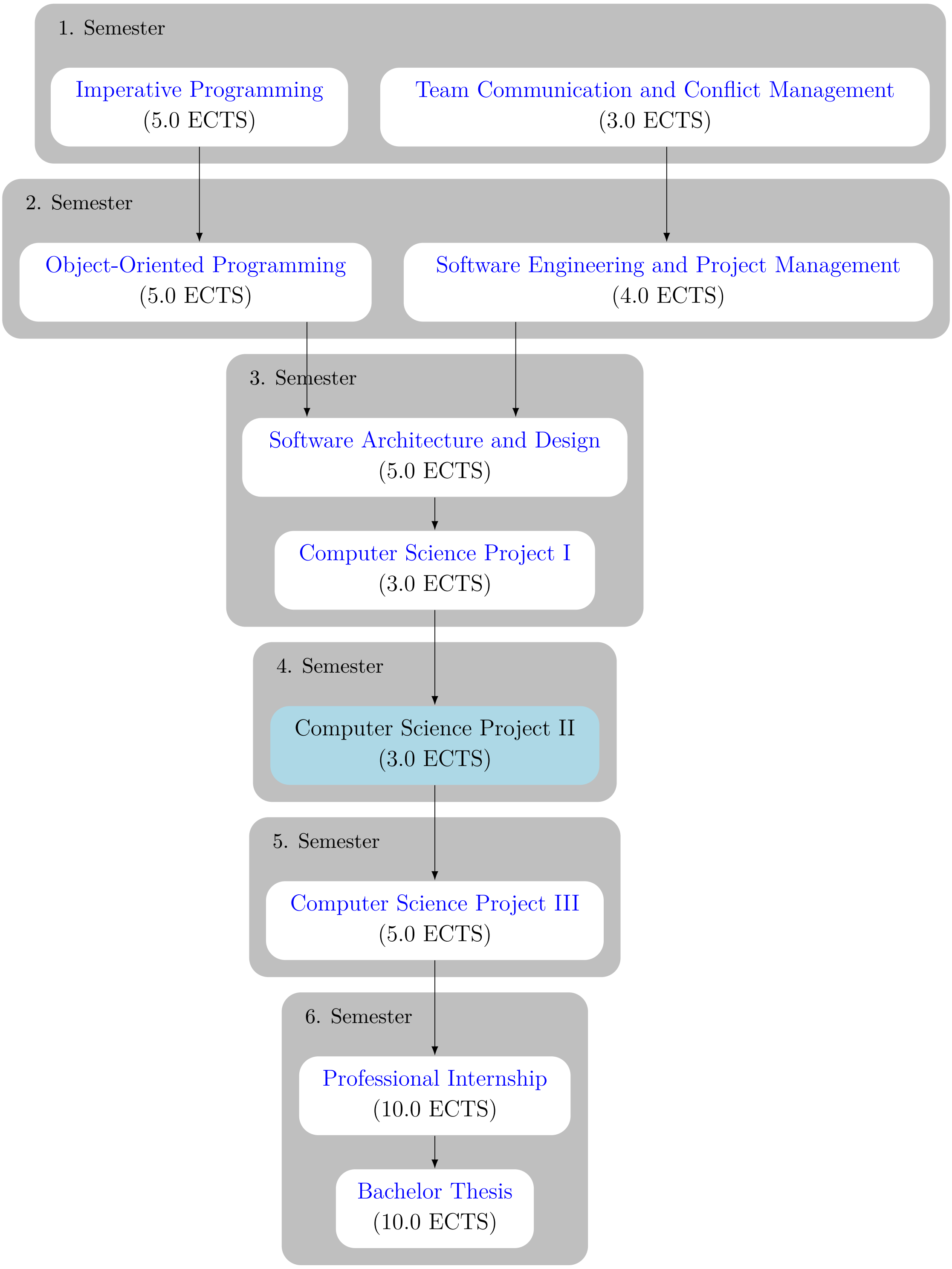
| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| create user personas that represent diverse populations, including those with physical, cognitive and sensory disabilities. | USE-LO.01 |
| analyze how interaction techniques affect usability and accessibility in real-life scenarios and suggest improvements. | USE-LO.01 |
| develop a prototype that includes various interaction techniques which ensures an intuitive and seamless user experience. | USE-LO.01,<br>USE-LO.03 |
| develop user interfaces that align with principles of human vision, ensuring readability, clarity and ease of use. | USE-LO.01,<br>USE-LO.03 |
| justify design decisions based on usability and accessibility factors. | USE-LO.01 |
| design user research activities such as interviews, surveys and usability testing. | USE-LO.01 |
| apply key accessibility standards to assess and improve the accessibility of user interfaces. | USE-LO.01 |
| demonstrate meticulousness by carefully integrating user needs into software solutions. | LO.08 |

### 3.4.6 Computer Science Project II

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 30 |
| Course type: | Project |
| Assessment: | Continuous |

### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Team Communication and Conflict Management (3.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Software Engineering and Project Management (4.0 ECTS)
3. Semester
Software Architecture and Design (5.0 ECTS)
Computer Science Project I (3.0 ECTS)
4. Semester
Computer Science Project II (3.0 ECTS)
5. Semester
Computer Science Project III (5.0 ECTS)
6. Semester
Professional Internship (10.0 ECTS)
Bachelor Thesis (10.0 ECTS)

### Course Contents

- Orientation within the curriculum
- Independent execution of a project
- Presentation of results

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| apply the acquired technical and social skills to complete practical projects that are related to real-world scenarios. | LO.08,<br>SIP-LO.01,<br>SIP-LO.02,<br>SIP-LO.03 |
| occupy a specific role within a team and fulfil the tasks of the role in a goal-driven manner in order to complete a project. | LO.08,<br>SIP-LO.01,<br>SIP-LO.02,<br>SIP-LO.03 |
| execute, document and justify project-related architecture, design and implementation decisions. | SIP-LO.02 |
| present project results and discuss them with peers and professionals, like faculty members. | LO.08,<br>SIP-LO.01,<br>SIP-LO.02 |
| demonstrate adaptability by satisfying project goals which may be continually evolving. | LO.08,<br>SIP-LO.02 |
| demonstrate willingness for collaboration to advance a project as a team. | LO.08,<br>SIP-LO.01,<br>SIP-LO.03 |
| demonstrate inventiveness by applying existing solution strategies to a new project. | LO.08,<br>SIP-LO.02 |
| demonstrate persistence by overcoming different forms of failures during a project. | LO.08,<br>SIP-LO.01 |
| demonstrate proactiveness by anticipating issues regarding the progression of a project. | LO.08 |

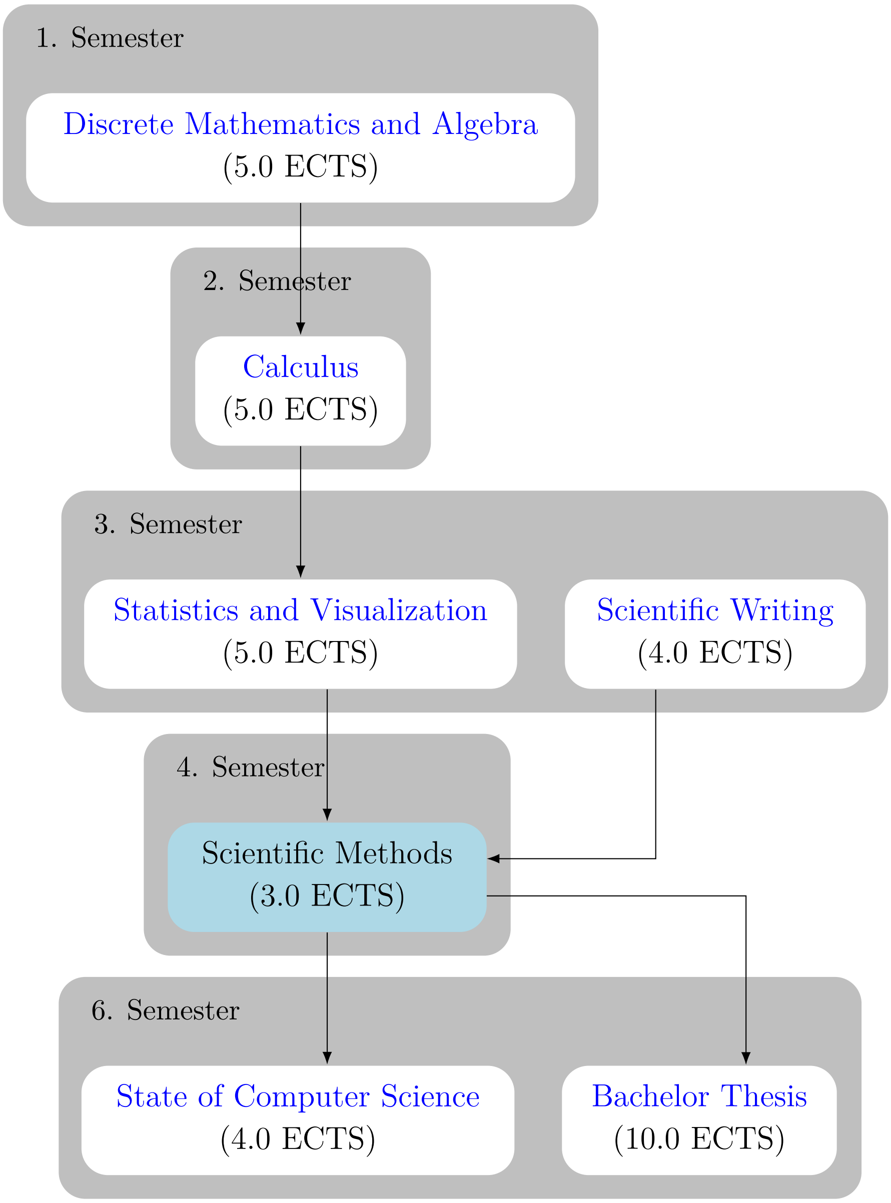
| | |
|---|---|
| demonstrate responsibility for all aspects of the solution developed during a project. | LO.08 |
| demonstrate self-directedness and internalize that goal definitions of a project are never complete. | LO.08 |

### 3.4.7 Scientific Methods

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 30 |
| Course type: | Seminar |
| Assessment: | Continuous |

### Context and Dependencies



### Course Contents

- Orientation within the curriculum
- Philosophical worldviews
- Selection of a research design
- Review of literature

- Research questions and hypotheses
- Quantitative methods
- Qualitative methods
- Mixed methods
- Design science research
- Data collection and recording
- Data analysis and interpretation
- Writing strategies

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the role of postpositivism, constructivism, transformative and pragmatism in research design. | SCI-LO.01,<br>SCI-LO.02 |
| organize, synthesize and evaluate existing literature. | SCI-LO.02,<br>SCI-LO.03 |
| identify research gaps and justify the need for new research. | SCI-LO.01,<br>SCI-LO.03 |
| formulate research questions and research proposals that align with a chosen research methodology. | SCI-LO.01,<br>SCI-LO.03 |
| differentiate between qualitative, quantitative, mixed methods and design science research approaches. | SCI-LO.01,<br>SCI-LO.02,<br>SCI-LO.03 |
| explain when and how to apply each research method based on a research question. | SCI-LO.01,<br>SCI-LO.02,<br>SCI-LO.03 |
| structure research papers and theses in a coherent, academic manner. | SCI-LO.03,<br>SCI-LO.04 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |

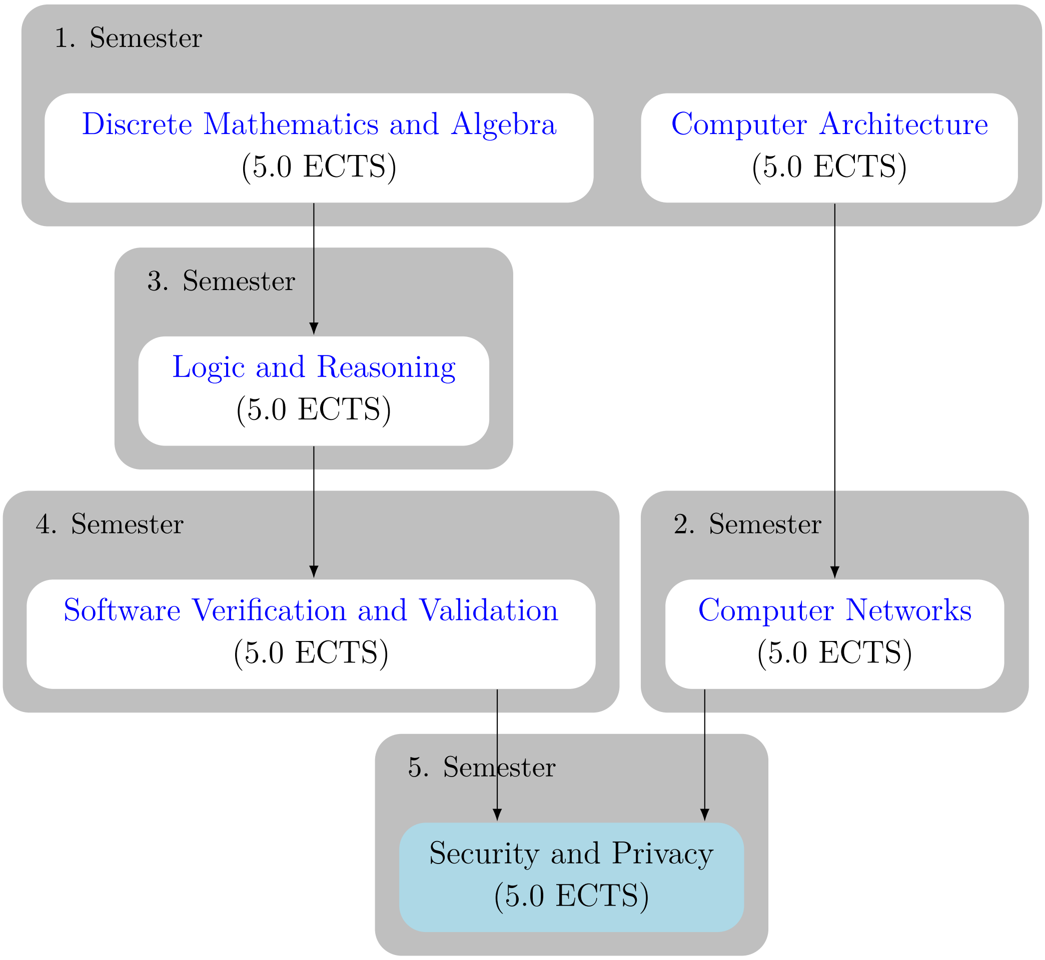
## 3.5 5. Semester

### 3.5.1 Security and Privacy

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies



### Course Contents

- Orientation within the curriculum
- Principles and practices of security and privacy
- Organizational objectives, policies and general risk assessment
- Security and privacy considerations in the development lifecycle
- Security and privacy considerations on every abstraction layer
- Engineering trade-offs regarding security
- Security attack domains and attack surfaces
- Security attack modes, techniques and tactics
- Common vulnerabilities and weaknesses, OWASP
- Protection and countermeasures
- Algorithmic, applied and mathematical views of cryptography
- Kerckhoffs's principle
- Classical, symmetric key and public key cryptography
- Data integrity and authentication
- Access control models
- Impact of artificial intelligence on security and privacy
- Homomorphic encryption

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| describe core principles of security and privacy and apply them by evaluating the security posture of a given system. | SYS-LO.07,<br>SYS-LO.08 |
| explain the role of security policies and perform a risk assessment based on an established framework. | SYS-LO.07,<br>SYS-LO.08 |

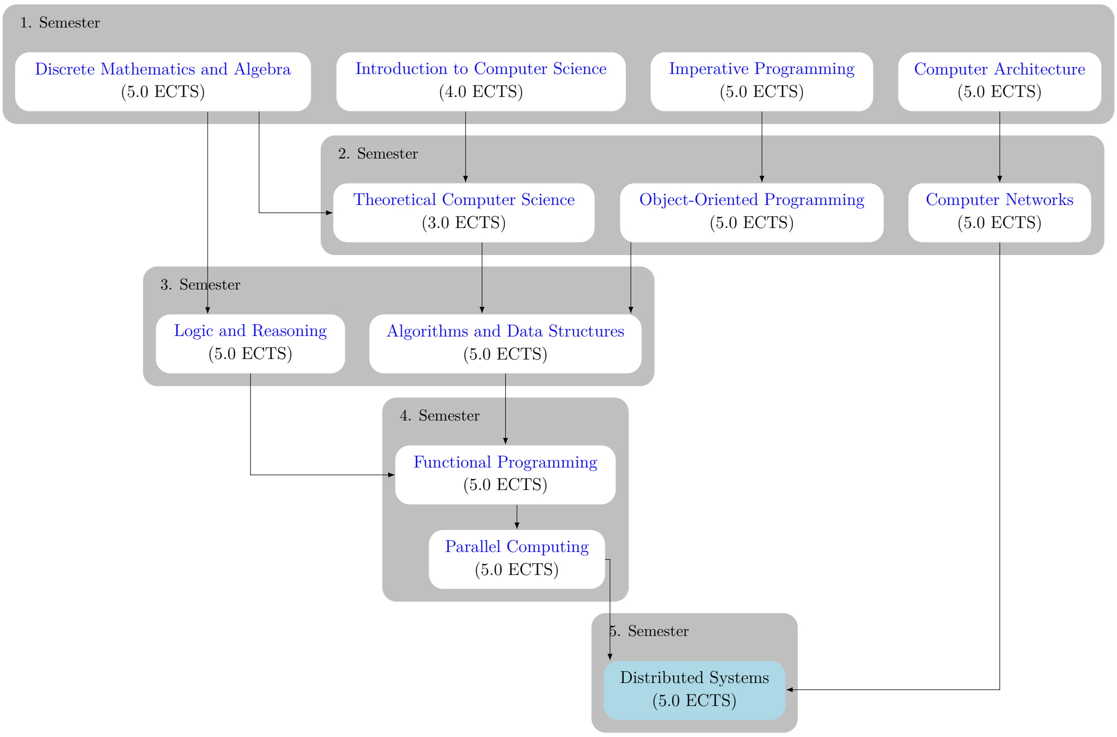
| | |
|---|---|
| identify security and privacy risks at different stages of the software development life cycle and give concrete examples of secure practices. | SYS-LO.07,<br>SYS-LO.08 |
| identify security and privacy risks on different abstraction layers, from hardware to applications. | SYS-LO.07,<br>SYS-LO.08 |
| recommend security solutions that balance protection with system requirements. | SYS-LO.07,<br>SYS-LO.08 |
| assess and reduce the attack surface of a given system using security best practices. | SYS-LO.07,<br>SYS-LO.08 |
| explain common attack modes and techniques and suggest corresponding countermeasures. | SYS-LO.07,<br>SYS-LO.08 |
| explain key cryptographic approaches and algorithms, and give a high-level description of their mathematical foundations. | SYS-LO.07 |
| analyze a given cryptographic system to determine if it aligns with Kerckhoffs's principle. | SYS-LO.07 |
| explain concepts of data integrity and authentication using hashing and digital signatures. | SYS-LO.07 |
| describe access control models and select/apply appropriate models to simulated real-world scenarios. | SYS-LO.07,<br>SYS-LO.08 |
| explain security- and privacy-related topics of artificial intelligence and homomorphic encryption, as well as their impact on computer science. | LO.06,<br>SYS-LO.07 |
| demonstrate meticulousness by verifying the correctness, security and privacy of solutions. | LO.08 |

## 3.5.2 Distributed Systems

### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

### Context and Dependencies


1. Semester
Discrete Mathematics and Algebra (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
Imperative Programming (5.0 ECTS)
Computer Architecture (5.0 ECTS)
2. Semester
Theoretical Computer Science (3.0 ECTS)
Object-Oriented Programming (5.0 ECTS)
Computer Networks (5.0 ECTS)
3. Semester
Logic and Reasoning (5.0 ECTS)
Algorithms and Data Structures (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Distributed Systems (5.0 ECTS)

### Course Contents

- Orientation within the curriculum
- Distributed architectural styles and patterns
- Centralization, decentralization, code migration
- Distributed communication media
- Cooperatively maintained data structures

- CAP theorem and its consequences
- Middleware-based communication
- Sockets and remote procedure calls
- Transparency and pitfalls
- Web services
- Non-determinism, consistency and fault tolerance
- Queuing systems

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to … | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the properties, advantages and disadvantages of distributed architectural styles and patterns. | SYS-LO.05 |
| explain the properties, advantages and disadvantages of centralized and decentralized distribution approaches. | SYS-LO.05 |
| explain the properties, advantages and disadvantages of cooperatively maintained data structures, like distributed hash tables and distributed file systems. | SYS-LO.05 |
| explain trade-offs between consistency, availability and partition tolerance and apply this understanding to existing system designs. | SYS-LO.05,<br>SYS-LO.06 |
| develop distributed applications using sockets and RPC libraries that simulate real-world scenarios. | SYS-LO.05,<br>SYS-LO.06 |
| analyze and discuss the transparency and corresponding pitfalls of a self-developed RPC-based application. | SYS-LO.05,<br>SYS-LO.06,<br>SYS-LO.08 |
| develop and deploy web services for real-world scenarios that adhere to the desired design principles of service-oriented architectures. | SYS-LO.05,<br>SYS-LO.06,<br>SYS-LO.08 |

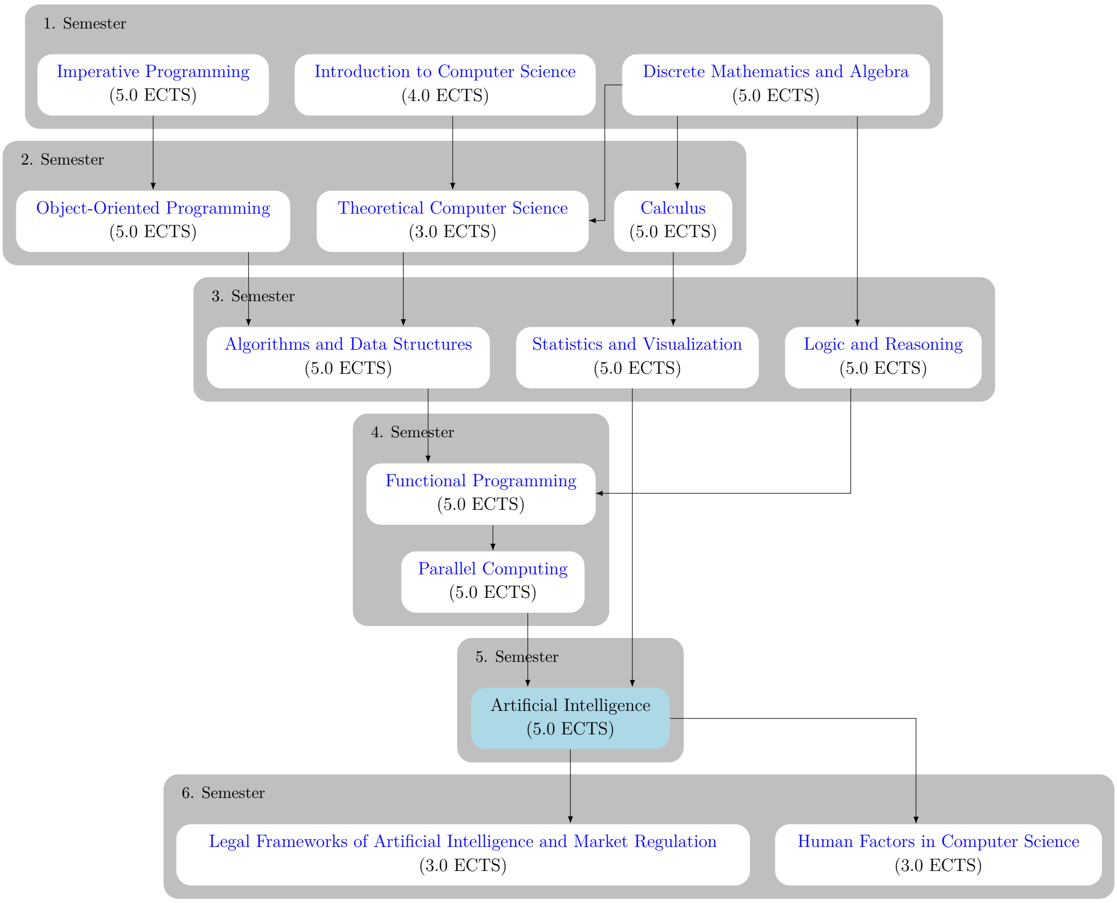
| | |
|---|---|
| explain algorithms for consistency and fault tolerance, like Paxos and Raft, and how eventual consistency is achieved in distributed systems. | SYS-LO.05 |
| develop distributed systems that simulate real-world scenarios using a message queuing library. | SYS-LO.05,<br>SYS-LO.06,<br>SYS-LO.08 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions. | LO.08 |

### 3.5.3 Artificial Intelligence

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Theoretical Computer Science (3.0 ECTS)
Calculus (5.0 ECTS)
3. Semester
Algorithms and Data Structures (5.0 ECTS)
Statistics and Visualization (5.0 ECTS)
Logic and Reasoning (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)
Human Factors in Computer Science (3.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Applications, problems, growth and impact

- Subfields of artificial intelligence
- Intelligent behavior
- Autonomous agents
- Problem characteristics
- Search approaches
- Space and time complexity of search
- Knowledge representation
- Planning approaches
- Predictive and descriptive models
- Data, features and examples
- Sources of error
- Bias-variance trade-off
- Regression and classification
- Model evaluation and tuning
- (Generative) neural networks
- Role of GPUs in machine learning

#### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| identify application domains of artificial intelligence and explain current challenges as well as its potential growth and societal impact. | LO.06 |
| differentiate and relate the subfields of artificial intelligence, such as machine learning, deep learning, natural language processing, robotics and computer vision. | LO.06,<br>INT-LO.04 |

| | |
|---|---|
| explain the concept of intelligent behavior and how it is modeled in artificial systems, and draw comparisons with human intelligence. | LO.06 |
| explain the concept of autonomous agents and their characteristics, including how they perceive environments, make decisions and act independently. | LO.06,<br>INT-LO.05 |
| classify and analyze the characteristics of different types of artificial intelligence problems. | INT-LO.04 |
| explain the concept of state space representation and search, and how problems can be modeled using this framework. | INT-LO.04,<br>INT-LO.05 |
| describe knowledge representation techniques and explain how knowledge can be structured for reasoning and problem-solving. | INT-LO.04,<br>INT-LO.05 |
| explain the fundamental principles of planning and how a system generates action sequences to achieve specified goals. | INT-LO.04,<br>INT-LO.05 |
| explain the basic concepts of generative models and their application domains. | INT-LO.04,<br>INT-LO.05 |
| explain mathematical concepts relevant to machine learning, including statistics, probability and linear algebra. | LO.06,<br>INT-LO.04,<br>INT-LO.05 |
| differentiate the paradigms of machine learning and explain which paradigm is appropriate for specific real-world scenarios. | INT-LO.04,<br>INT-LO.05 |
| explain the significance of data quality and feature selection in machine learning. | INT-LO.05,<br>INT-LO.06 |
| differentiate supervised machine learning algorithms and systematically apply selected ones to real-world scenarios. | INT-LO.05,<br>INT-LO.06 |
| explain the bias-variance trade-off and its impact on model performance. | INT-LO.05,<br>INT-LO.06 |
| evaluate the performance of a self-created machine learning model and improve it by systematically tuning its parameters. | INT-LO.05,<br>INT-LO.06 |
| explain the fundamentals of (generative) neural networks and their architectures. | INT-LO.05,<br>INT-LO.06 |
| train (generative) neural networks, evaluate their performance and systematically tune their parameters. | INT-LO.05,<br>INT-LO.06 |
| explain the significance of GPU acceleration in training machine learning models. | INT-LO.05 |

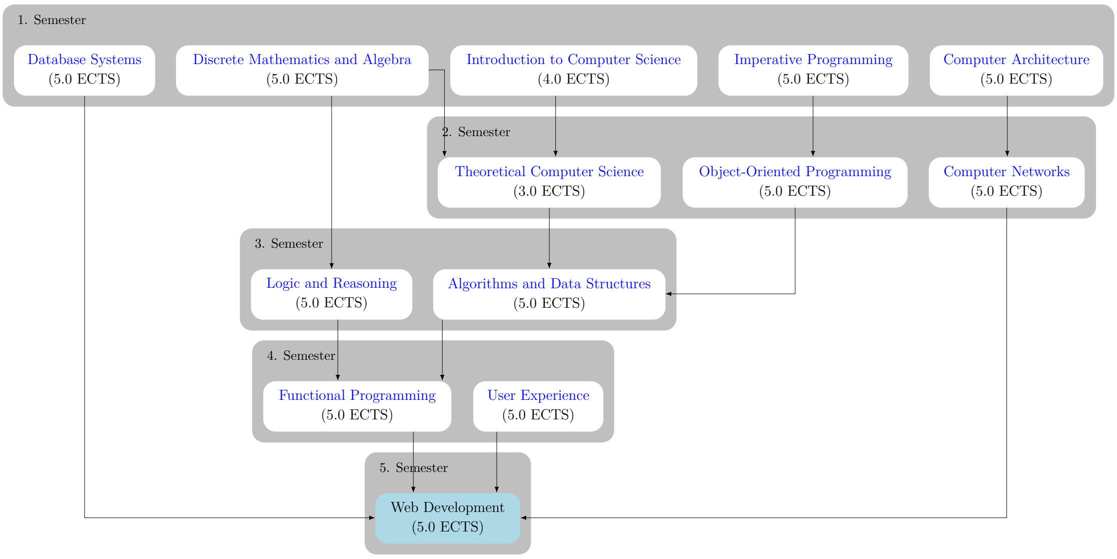
| | |
|---|---|
| demonstrate inventiveness by applying existing machine learning approaches to new contexts. | LO.08 |

### 3.5.4 Web Development

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Database Systems (5.0 ECTS)
Discrete Mathematics and Algebra (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
Imperative Programming (5.0 ECTS)
Computer Architecture (5.0 ECTS)
2. Semester
Theoretical Computer Science (3.0 ECTS)
Object-Oriented Programming (5.0 ECTS)
Computer Networks (5.0 ECTS)
3. Semester
Logic and Reasoning (5.0 ECTS)
Algorithms and Data Structures (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
User Experience (5.0 ECTS)
5. Semester
Web Development (5.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Architectural patterns for web applications
- Web-related development environment and technologies
- Responsive web design principles
- Frontend development
- Reactive programming
- Backend integration
- Web application security

## Learning Outcomes

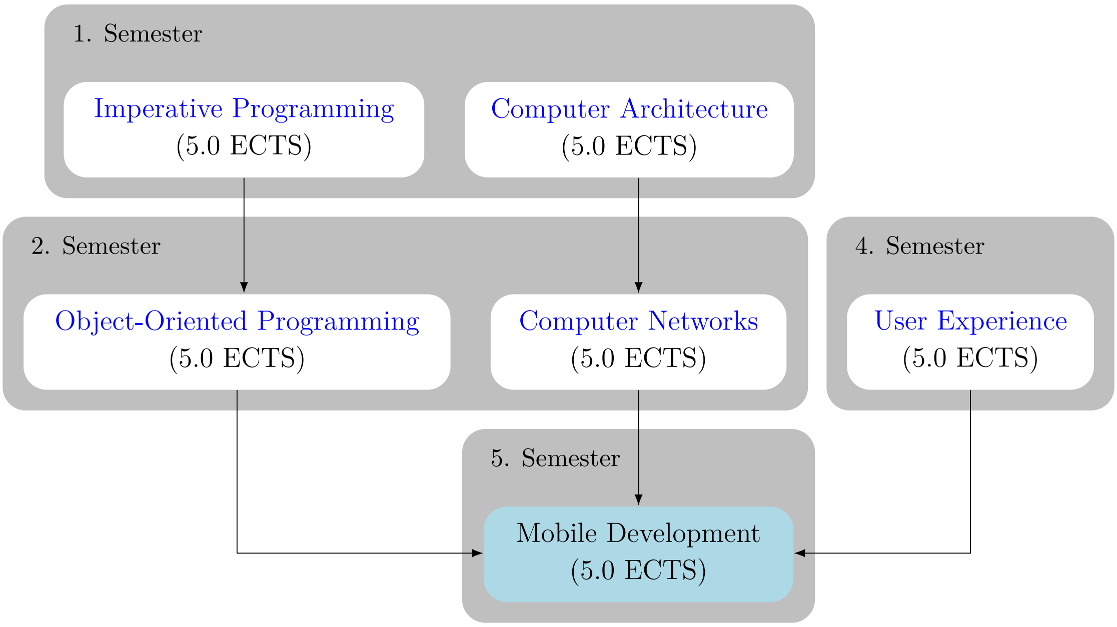
| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| explain the architecture of web applications and the roles of frontend, backend and databases. | USE-LO.02 |
| develop responsive web pages using markup, stylesheet and scripting languages in a reliable way for real-world scenarios. | USE-LO.02,<br>USE-LO.03 |
| design and implement server-side APIs to handle secure client-server communication and initiate backend activities. | USE-LO.02,<br>USE-LO.03 |
| develop web applications which adopt secure authentication, authorization and session handling techniques. | USE-LO.02,<br>USE-LO.03 |
| integrate best practices for security into web applications, like input validation, HTTPS and single sign-on. | USE-LO.02,<br>USE-LO.03 |
| demonstrate meticulousness by verifying the correctness, completeness, security and usability of solutions. | LO.08 |

### 3.5.5 Mobile Development

#### Course Profile

|  |  |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 50 |
| Course type: | Integrated course |
| Assessment: | Continuous |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Computer Architecture (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Computer Networks (5.0 ECTS)
4. Semester
User Experience (5.0 ECTS)
5. Semester
Mobile Development (5.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Mobile development lifecycle
- Native vs. hybrid vs. cross-platform applications
- Platform-specific development environment and technologies
- Architectural patterns for mobile applications
- Mobile user interface design principles
- Hardware components and sensor data

- Backend integration
- Mobile application security
- Deployment and publishing

### Learning Outcomes

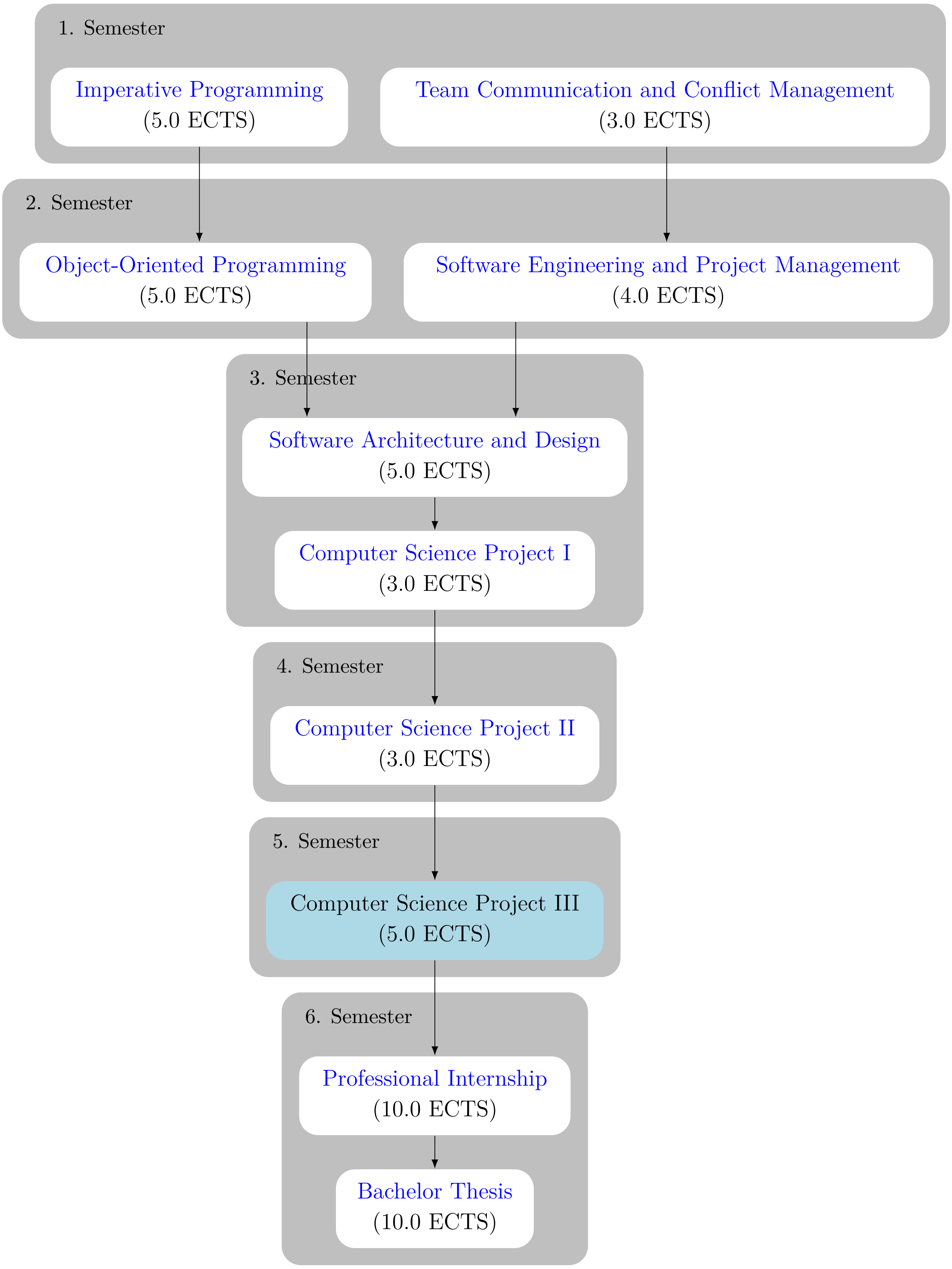
| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| develop mobile applications for selected platforms using related development tools and programming languages. | USE-LO.02,<br>USE-LO.03 |
| design user-friendly and accessible mobile user interfaces that enhance user experience. | USE-LO.01,<br>USE-LO.02,<br>USE-LO.03 |
| integrate hardware components and sensor data into mobile applications, like camera, GPS and voice. | USE-LO.02,<br>USE-LO.03 |
| integrate mobile applications with backend services while handling data communication securely and efficiently. | USE-LO.02,<br>USE-LO.03 |
| test, debug and deploy mobile applications while adhering to platform-specific guidelines and best practices. | USE-LO.02 |
| prepare a mobile application for release by simulating an app store submission process. | USE-LO.02 |
| demonstrate meticulousness by carefully considering the expectations of users when developing mobile applications. | LO.08 |

### 3.5.6 Computer Science Project III

#### Course Profile

| | |
|---|---|
| Workload: | 5.0 ECTS |
| Teaching units: | 30 |
| Course type: | Project |
| Assessment: | Continuous |

### Context and Dependencies



### Course Contents

- Orientation within the curriculum
- Independent execution of a project
- Presentation of results

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| apply the acquired technical and social skills to complete practical projects that are related to real-world scenarios. | LO.08, SIP-LO.01, SIP-LO.02, SIP-LO.03 |
| occupy a specific role within a team and fulfill the tasks of the role in a goal-driven manner in order to complete a project. | LO.08, SIP-LO.01, SIP-LO.02, SIP-LO.03 |
| acquire knowledge on their own and utilize it to complete practical projects that are related to real-world scenarios. | LO.08, SIP-LO.01, SIP-LO.02, SIP-LO.03 |
| execute, document and justify project-related architecture, design and implementation decisions. | SIP-LO.02 |
| present project results and discuss them with peers and professionals, like faculty members. | LO.08, SIP-LO.01, SIP-LO.02 |
| demonstrate adaptability by satisfying project goals which may be continually evolving. | LO.08, SIP-LO.02 |
| demonstrate willingness for collaboration to advance a project as a team. | LO.08, SIP-LO.01, SIP-LO.03 |
| demonstrate inventiveness by applying existing solution strategies to a new project. | LO.08, SIP-LO.02 |

| | |
|---|---|
| demonstrate persistence by overcoming different forms of failures during a project. | LO.08,<br>SIP-LO.01 |
| demonstrate proactiveness by anticipating issues regarding the progression of a project. | LO.08 |
| demonstrate responsibility for all aspects of the solution developed during a project. | LO.08 |
| demonstrate self-directedness and internalize that goal definitions of a project are never complete. | LO.08 |

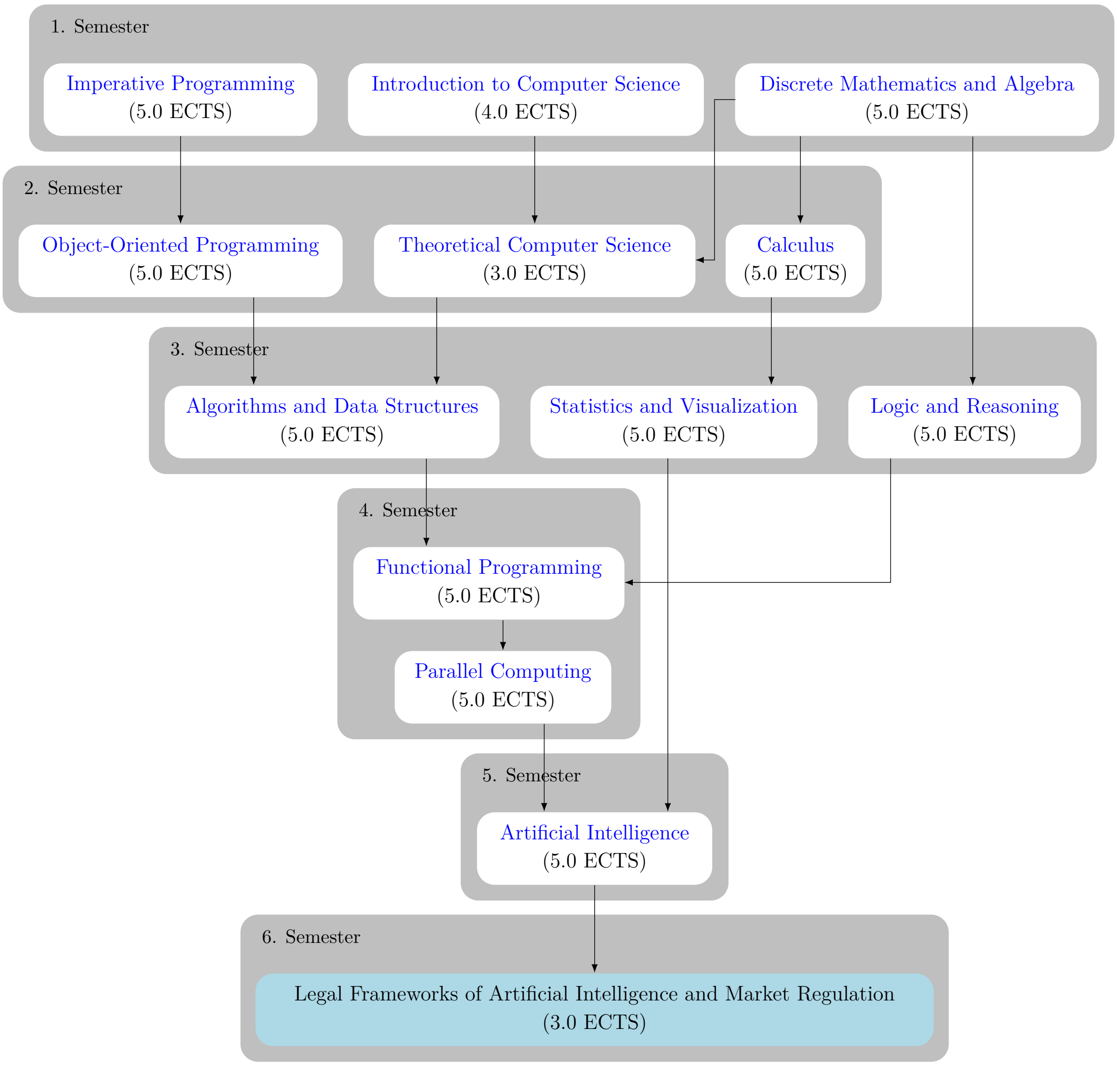
# 3.6 6. Semester

## 3.6.1 Legal Frameworks of Artificial Intelligence and Market Regulation

### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 42 |
| Course type: | Lecture with exercises |
| Assessment: | Final |

### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Theoretical Computer Science (3.0 ECTS)
Calculus (5.0 ECTS)
3. Semester
Algorithms and Data Structures (5.0 ECTS)
Statistics and Visualization (5.0 ECTS)
Logic and Reasoning (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
6. Semester
Legal Frameworks of Artificial Intelligence and Market Regulation (3.0 ECTS)

## Course Contents

- Orientation within the curriculum
- AI legal frameworks
- Ethics in AI development and use
- Legal Risk Assessment
- Innovation and Compliance
- Regulated and unregulated economies
- Monopolistic practices
- Pricing models and deployment strategies
- Ethical concerns surrounding economies of computing
- Case studies and applications

## Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| describe the fundamental principles and objectives of AI legal framework, including the balance between innovation, ethical considerations and legal compliance in global contexts. | LAW-LO.01,<br>LAW-LO.04,<br>LAW-LO.07 |
| evaluate legal risks associated with the development, deployment and use of AI systems, including liability issues and data protection challenges. | LAW-LO.03,<br>LAW-LO.04 |
| analyze how AI technologies influence regulatory frameworks, including competition law, consumer protection, with a focus on addressing cross-jurisdictional challenges. | LAW-LO.04,<br>LAW-LO.07 |
| effectively address the challenges at the intersection of technological innovation and legal compliance, particularly in competitive and rapidly evolving industries. | LAW-LO.07 |
| explain the differences between regulated and unregulated markets and how they impact technological innovation. | LAW-LO.05 |

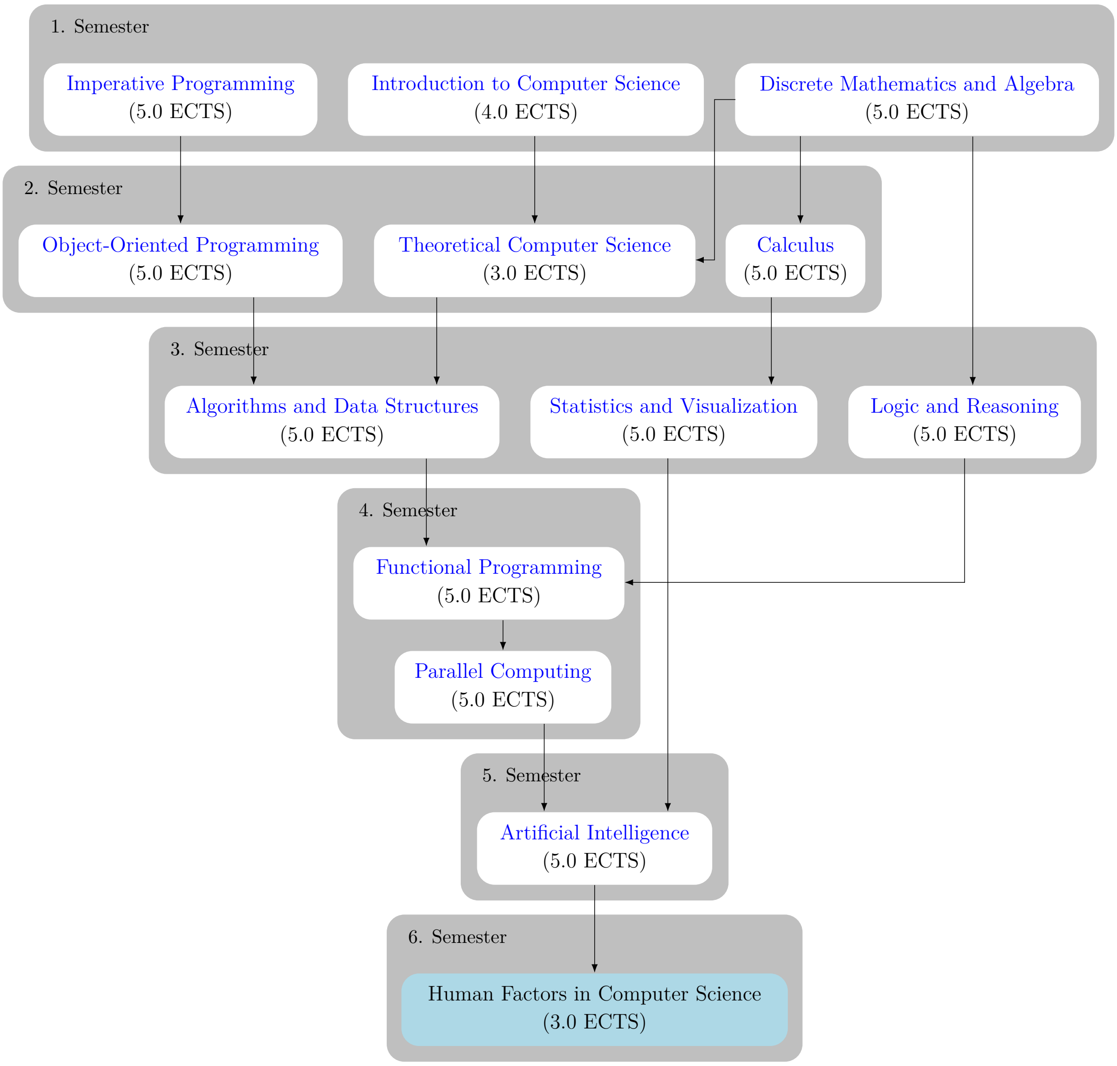
| | |
|---|---|
| discuss examples of regulatory interventions in the tech industry and their effects on competition and consumer welfare. | LAW-LO.05,<br>LAW-LO.07 |
| explain the impact of monopolistic practices on innovation, pricing and consumer choice. | LAW-LO.05,<br>LAW-LO.07 |
| analyze real-world scenarios of tech monopolies and their antitrust actions. | LAW-LO.05,<br>LAW-LO.07 |
| explain various pricing models for software systems and the strategic choices companies make when deploying new technologies and services. | LAW-LO.05 |

### 3.6.2 Human Factors in Computer Science

#### Course Profile

| | |
|---|---|
| Workload: | 3.0 ECTS |
| Teaching units: | 42 |
| Course type: | Lecture with exercises |
| Assessment: | Final |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Introduction to Computer Science (4.0 ECTS)
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Theoretical Computer Science (3.0 ECTS)
Calculus (5.0 ECTS)
3. Semester
Algorithms and Data Structures (5.0 ECTS)
Statistics and Visualization (5.0 ECTS)
Logic and Reasoning (5.0 ECTS)
4. Semester
Functional Programming (5.0 ECTS)
Parallel Computing (5.0 ECTS)
5. Semester
Artificial Intelligence (5.0 ECTS)
6. Semester
Human Factors in Computer Science (3.0 ECTS)

### Course Contents

- Orientation within the curriculum
- Societal and ethical impacts of computing
- Pervasive computing and surveillance
- (Algorithmic) fairness and trust
- Human agency and oversight in (autonomous) computing
- Green computing
- Environmental impacts of implementation decisions
- Accountability, responsibility and liability for computing products
- Ethical decision making
- Ethical dissent and whistleblowing
- Diversity in development teams
- Programming ethics
- Digital well-being

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| analyze societal impact of real-world applications and identify issues regarding ethics, fairness, bias, trust and explainability. | LAW-LO.07 |
| explain ubiquitous computing, its potential for surveillance and propose ethical frameworks for balancing innovation with personal privacy. | LAW-LO.03,<br>LAW-LO.07 |
| identify biases in algorithms and develop strategies for improving trust in AI systems. | LAW-LO.04,<br>LAW-LO.07 |
| explain real-world scenarios where lack of human oversight in computing can led to negative outcomes. | LAW-LO.07 |

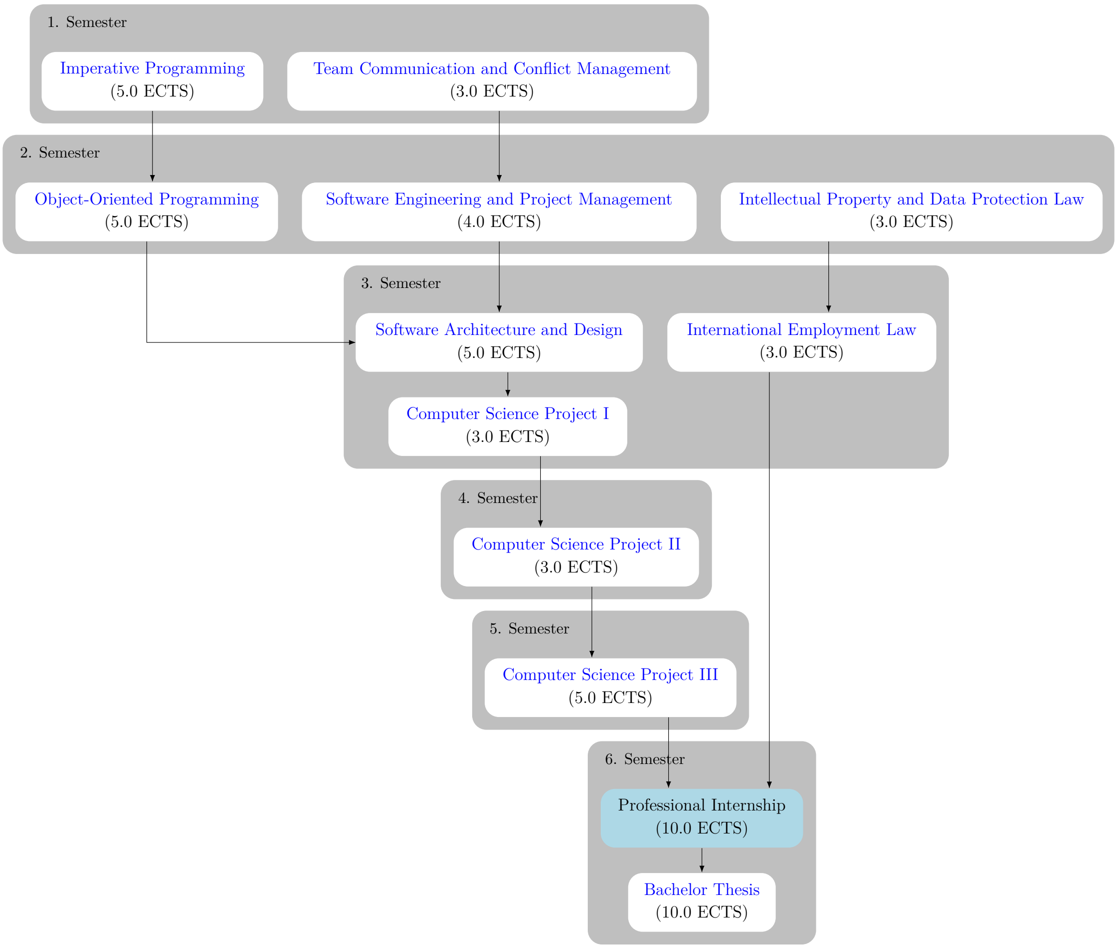
| | |
|---|---|
| explain environmental impacts of computing infrastructures and recommend eco-friendly alternatives in technology decision-making processes. | LAW-LO.07 |
| explain personal and corporate liability in technology failures and unethical utilization of computing technology. | LAW-LO.01,<br>LAW-LO.04,<br>LAW-LO.07 |
| apply ethical decision-making frameworks to real-world computing scenarios. | LAW-LO.01,<br>LAW-LO.07 |
| explain how to act ethically when faced with unethical practices within organizations. | LAW-LO.01,<br>LAW-LO.07 |
| explain how diversity improves creativity, problem-solving and product development. | LAW-LO.07 |
| explain the social responsibility of programmers in contributing to the greater good through their work. | LAW-LO.07 |
| propose strategies for balancing technology usage with a healthy lifestyle. | LAW-LO.07 |

### 3.6.3 Professional Internship

#### Course Profile

| | |
|---|---|
| Workload: | 10.0 ECTS |
| Teaching units: | 2 |
| Course type: | Internship |
| Assessment: | Final |

#### Context and Dependencies


1. Semester
Imperative Programming
(5.0 ECTS)
Team Communication and Conflict Management
(3.0 ECTS)
2. Semester
Object-Oriented Programming
(5.0 ECTS)
Software Engineering and Project Management
(4.0 ECTS)
Intellectual Property and Data Protection Law
(3.0 ECTS)
3. Semester
Software Architecture and Design
(5.0 ECTS)
International Employment Law
(3.0 ECTS)
Computer Science Project I
(3.0 ECTS)
4. Semester
Computer Science Project II
(3.0 ECTS)
5. Semester
Computer Science Project III
(5.0 ECTS)
6. Semester
Professional Internship
(10.0 ECTS)
Bachelor Thesis
(10.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Accompanying seminar

- Internship activity

### Learning Outcomes

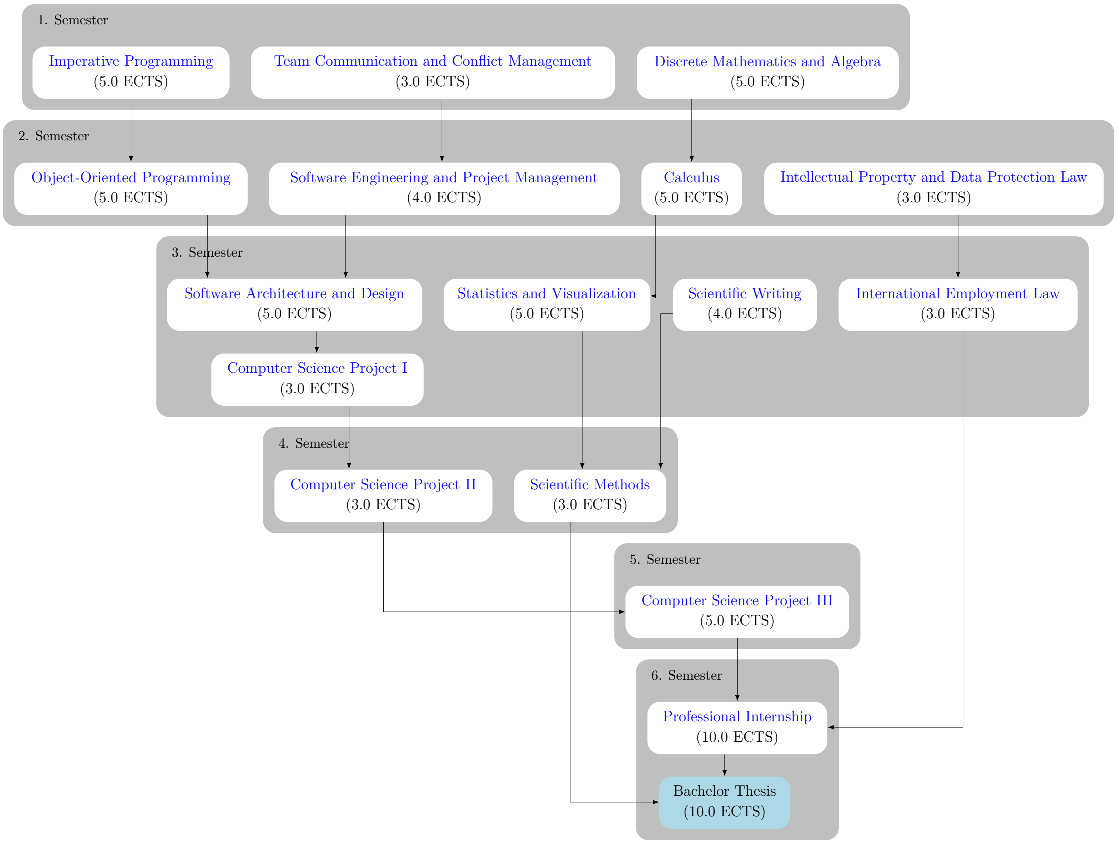
| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| perform activities related to computer science while being employed. | LO.08,<br>SIP-LO.02 |
| reflect on the interrelationships between internship activities and associated topics of the curriculum. | LO.08 |
| summarize and document the insights gained by carrying out an internship. | SIP-LO.02 |
| demonstrate adaptability by satisfying internship goals which may be continually evolving. | LO.08,<br>SIP-LO.02 |
| demonstrate inventiveness by applying learned solution strategies to real-world problems. | LO.08,<br>SIP-LO.02 |
| demonstrate meticulousness by verifying the correctness and completeness of solutions developed during an internship. | LO.08,<br>SIP-LO.02 |
| demonstrate persistence by overcoming different forms of failures during an internship. | LO.08,<br>SIP-LO.01 |
| demonstrate responsibility for all aspects of the solution developed during an internship. | LO.08 |
| demonstrate self-directedness by carrying out the internship based on a self-developed plan. | LO.08,<br>SIP-LO.02 |

### 3.6.4 Bachelor Thesis

#### Course Profile

| | |
|---|---|
| Workload: | 10.0 ECTS |
| Teaching units: | 20 |
| Course type: | Seminar |
| Assessment: | Final |

#### Context and Dependencies


1. Semester
Imperative Programming (5.0 ECTS)
Team Communication and Conflict Management (3.0 ECTS)
Discrete Mathematics and Algebra (5.0 ECTS)
2. Semester
Object-Oriented Programming (5.0 ECTS)
Software Engineering and Project Management (4.0 ECTS)
Calculus (5.0 ECTS)
Intellectual Property and Data Protection Law (3.0 ECTS)
3. Semester
Software Architecture and Design (5.0 ECTS)
Statistics and Visualization (5.0 ECTS)
Scientific Writing (4.0 ECTS)
International Employment Law (3.0 ECTS)
Computer Science Project I (3.0 ECTS)
4. Semester
Computer Science Project II (3.0 ECTS)
Scientific Methods (3.0 ECTS)
5. Semester
Computer Science Project III (5.0 ECTS)
6. Semester
Professional Internship (10.0 ECTS)
Bachelor Thesis (10.0 ECTS)

#### Course Contents

- Orientation within the curriculum
- Planning a research paper
- Conducting a planned study
- Writing a research paper

### Learning Outcomes

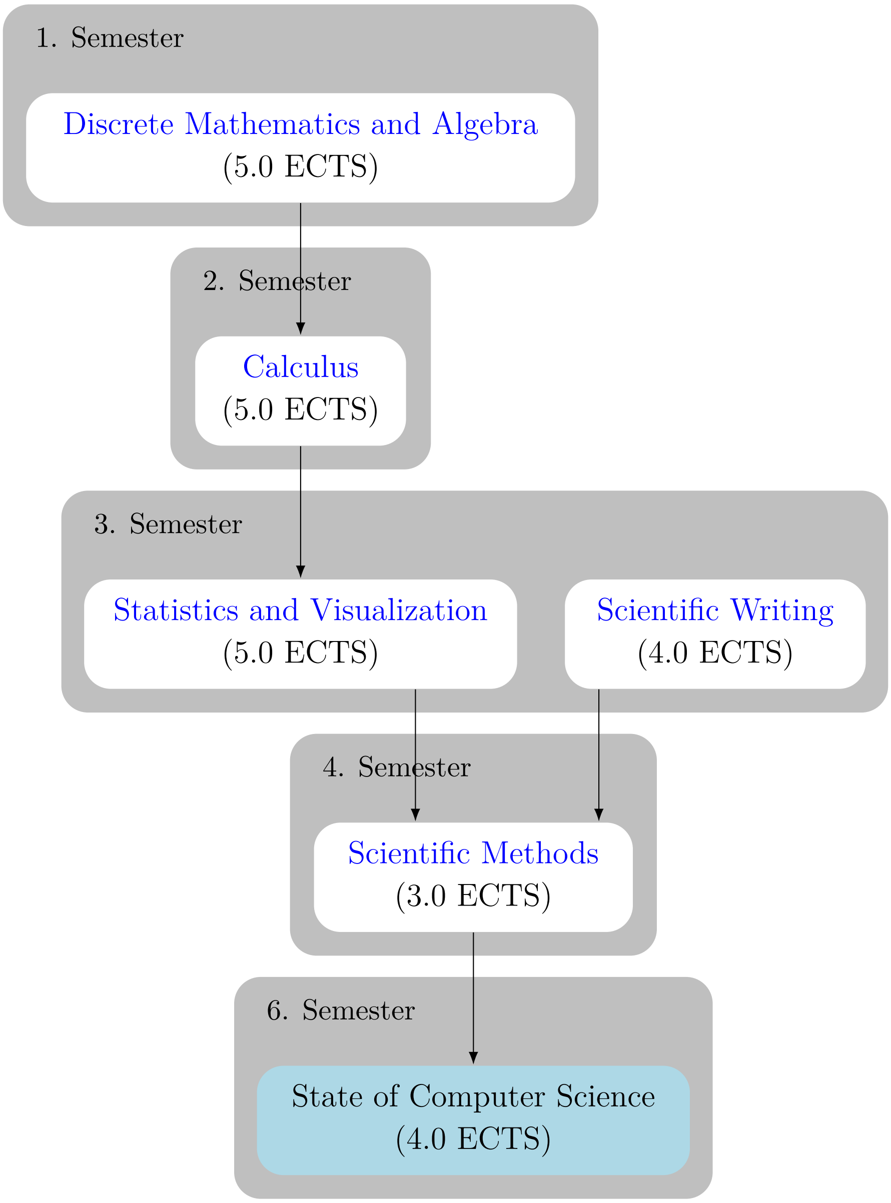
| **Learning Outcome**<br>After completing the course, students are able to ... | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| assess the current state of scientific knowledge in a particular field and derive a research gap. | SCI-LO.01, SCI-LO.02 |
| formulate research questions. | SCI-LO.01 |
| plan the process of authoring a research paper by writing a research proposal. | SCI-LO.01, SCI-LO.03 |
| answer research questions using established research methods under academic integrity. | SCI-LO.01, SCI-LO.02 |
| critically reflect on the procedure, results and impact of a self-conducted scientific study. | SCI-LO.02 |
| author a research paper describing the results of a self-conducted scientific study. | SCI-LO.03, SCI-LO.04 |
| demonstrate inventiveness by applying existing solution strategies to new contexts. | LO.08 |
| demonstrate adaptability by satisfying a research goal which may be continually evolving. | LO.08 |
| demonstrate proactiveness by anticipating issues regarding the progression of the bachelor thesis. | LO.08 |
| demonstrate self-directedness by adhering to the planned research process. | LO.08, SCI-LO.01 |
| demonstrate responsibility for all aspects of the solution developed during the bachelor thesis. | LO.08 |

### 3.6.5 State of Computer Science

#### Course Profile

| | |
|---|---|
| Workload: | 4.0 ECTS |
| Teaching units: | 20 |
| Course type: | Seminar |
| Assessment: | Continuous |

### Context and Dependencies



### Course Contents

- Orientation within the curriculum
- Current research in computer science
- Future of computer science
- Links to potential master level research

### Learning Outcomes

| **Learning Outcome**<br>After completing the course, students are able to … | **Addressed Learning Outcomes** |
|---|---|
| explain contentual dependencies between this course and previous and subsequent courses of the curriculum. | LO.08 |
| identify a research trend, synthesize it, contextualize it, present it and discuss it with peers and professionals, like faculty members. | LO.06,<br>SCI-LO.02,<br>SCI-LO.04 |
| demonstrate self-directedness by identifying and addressing individual research interests. | LO.08 |