CONSTRUCTOR
UNIVERSITY

Study Program Handbook

Computer Science

Bachelor of Science

Subject-specific Examination Regulations for Computer Science (Fachspezifische Prüfungsordnung)

The subject-specific examination regulations for Computer Science are defined by this program handbook and are valid only in combination with the General Examination Regulations for Undergraduate degree programs (General Examination Regulations = Rahmenprüfungsordnung). This handbook also contains the program-specific Study and Examination Plan (Chapter 6).

Upon graduation, students in this program will receive a Bachelor of Science (BSc) degree with a scope of 180 ECTS (for specifics see Chapter 4 of this handbook).

Version	Valid as of	Decision	Details
Fall 2025- V1.1		Nov 14, 2025	Editorial update: Corrected error in Machine Learning; removed Probability and Random Processes as a prerequisite.
Fall 2025 – V1	Sep 01, 2025		
		Apr 26, 2023	Substantial change approved by the Academic Senate
		Jun 26, 2019	Originally approved by Academic Senate

Contents

1 Program Overview

1.1 Concept

1.1.1 The Constructor University Educational Concept

Constructor University aims to educate students for both an academic and a professional career by emphasizing three core objectives: academic excellence, personal development, and employability to succeed in the working world. Constructor University offers an excellent research driven education experience across disciplines to prepare students for graduate education as well as career success by combining disciplinary depth and interdisciplinary breadth with supplemental skills education and extra-curricular elements. Through a multi-disciplinary, holistic approach and exposure to cutting-edge technologies and challenges, Constructor University develops and enables the academic excellence, intellectual competences, societal engagement, professional and scientific skills of tomorrows leaders for a sustainable and peaceful future.

In this context, it is Constructor University's aim to educate talented young people from all over the world, regardless of nationality, religion, and material circumstances, to become citizens of the world who are able to take responsible roles for the democratic, peaceful, and sustainable development of the societies in which they live. This is achieved through high-quality teaching as well as manageable study loads and supportive study conditions. Study programs and related study abroad programs convey academic knowledge as well as the ability to interact positively with other individuals and groups in culturally diverse environments. The ability to succeed in the working world is a core objective for all study programs at Constructor University, both in terms of actual disciplinary subject matter and also to the social skills and intercultural competence. Study-program-specific modules and additional specializations provide the necessary depth, interdisciplinary offerings and the minor option provide breadth while the university-wide general foundation and methods modules, optional German language and Humanities modules, and an extended internship period strengthen the employability of students. The concept of living and learning together on an international campus with many cultural and social activities supplements students' education. In addition, Constructor University offers professional advising and counseling.

Constructor University's educational concept is highly regarded both nationally and internationally. While the university has consistently achieved top marks over the last decade in Germany's most comprehensive and detailed university ranking by the Center for Higher Education (CHE), it has also been listed by one of the most widely observed university rankings, the Times Higher Education (THE) ranking. More details on the current ranking positions can be found at https://constructor.university/more/about-us.

1.1.2 Program Concept

Computer Science lies at the core of all modern industries and plays a major role in most areas of science as well. Computer technology changes constantly, but the fundamental principles underlying these technologies have now developed into a mature science. The Computer Science Bachelor of Science program at Constructor University focuses on the understanding of these principles and their application in practice.

Students will obtain core computer science competencies and skills (e.g., programming and software engineering) and they will learn about fundamental abstractions and abstract notions of computing (e.g., formal languages, logic, and computability theory). They will learn about the principles behind and the proper usage of core technologies (e.g., databases, operating systems, and computer networks). Finally, students will develop an understanding of the limitations of technology and side effects of computing systems (e.g., security, dependability, legal, and ethical aspects). Because computer science is rooted in mathematics, students will take mathematical methods modules covering calculus, linear algebra, probability theory, and numerical methods or statistics.

The job market for computer scientists has been very favorable in the last few years, and there is no indication that this will change in the near future. Because of the rapid changes in the field, it is important to focus the education on the fundamental principles, as well as, subfields of promising future relevance. Cross-disciplinary breadth and flexibility, as well as social and work organization skills are increasingly important. The minor option allows the combination of the education in computer science with a different discipline, thereby facilitating a cross-disciplinary specialization. The academic qualifications and personal profiles for academic and industrial careers differ. Constructor University's Computer Science program responds to the needs of both areas by offering a Computer Science major designed for students who plan to work in the information technology industry or join graduate programs related to the discipline. Students choosing the minor option can acquire basic skills in a specific application domain, which makes them very well suited to work in a specific industrial sector. The minor option can also be used to obtain specific knowledge in a closely related discipline to develop a strong portfolio of knowledge at the intersection of computer science with related disciplines.

1.2 Specific Advantages of Computer Science at Constructor University

The Computer Science program at Constructor University aims to be rigorous with respect to the foundations, while at the same time being very contemporary with an international orientation.

- The educational approach of the faculty is to relate the theoretical contents of the discipline to their contemporary application in industry and research. The instructors aim to include recent developments of the topics covered to demonstrate how basic methods or techniques are applied today and how the material covered relates to research challenges.
- Early involvement in research projects is an essential aspect of student education. Students can obtain a vivid research experience at a very early stage, which often develops into interdisciplinary collaborations later.
- This distinctive educational approach, together with the positive teaching environment, has been acknowledged in several rankings: In the computer science ranking published by the Centre for Higher Education (CHE) in 2015, the support by instructors and the relationship to research were ranked 1st of 68 study programs. In the European U-Multirank ranking published in 2018, the overall learning experience in computer science was ranked 10th and research-oriented teaching in computer science was ranked 2nd of 304 European universities offering Computer Science programs.
- The involvement of students and alumni in the program development process using a direct and open dialogue ensures that the program is constantly fine-tuned to the specific needs of students, such as covering certain topics at a certain time with respect to the preparation of internship or job applications.
- Student teams participate regularly in international programming competitions. Constructor University hosted the Northwestern European Regional Contest (NWERC) of the ACM International Collegiate Programming Contest on campus in 2010 and 2011. Student teams

have participated in NWERC competitions since then on an annual basis. In 2014, students organized the first JacobsHack! hackathon on campus, which was sponsored, among others, by Google, Microsoft, and SAP. The 2018 edition of JacobsHack!, sponsored, among others, by Facebook, Skyscanner, GitHub and Bloomberg, attracted participants from all over Europe. More recently, students participated in a Causal Machine Learning Hackathon sponsored by BMW and Constructor University in 2024.

1.3 Program-Specific Educational Aims

1.3.1 Qualification Aims

The main subject-specific qualification aim is to enable students to take up qualified employment in modern industries involving information technology or to enter graduate programs related to computer science. Graduates of the Computer Science program have obtained the following competencies:

- Computer science competence
  Graduates are familiar with the theoretical foundations of computer science, and they are able to design and develop computer systems addressing a given application scenario. They are able to analyze and structure complex problems and they are able to address them using methods of computer science. Graduates are able to construct and maintain complex computer systems using a structured, analytic, and creative approach.

- Communication competence
  Graduates are able to communicate subject-specific topics convincingly in both spoken and written form to fellow computer scientists or to customers.

- Teamwork and project management competence
  Graduates are able to work effectively in a team, and they are able to organize workflows in complex development efforts. They are familiar with tools that support the development, testing, and maintenance of large software systems and they are able to take design decisions in a constructive way.

- Learning competence
  Graduates have acquired a solid foundation enabling them to assess their own knowledge and skills, learn effectively, and remain up to date with the latest developments in the rapidly evolving field of computer science.

- Personal and professional competence
  Graduates are able to develop a professional profile, justify professional decisions based on theoretical and methodical knowledge, and critically reflect on their behavior with respect to their consequences for society.

The design of the Computer Science program follows national guidelines published by the Gesellschaft für Informatik (GI) (GI: Empfehlungen für Bachelor- und Masterprogramme im Studienfach Informatik an Hochschulen, July 2016) and international guidelines published jointly by the Association for Computing Machinery (ACM) and the Institute of Electrical and Electronics Engineers (IEEE) (ACM/IEEE: Computer Science Curricula 2013, December 2013).

1.3.2 Intended Learning Outcomes

By the end of the program, students will be able to

1. work professionally in the highly dynamic computer science field and enter graduate programs related to computer science;
2. apply fundamental concepts of computer science while solving problems;
3. think in an analytical way at multiple levels of abstraction;
4. develop, analyze and implement algorithms using modern software engineering methods;
5. understand the characteristics of a range of computing platforms and their advantages and limitations;
6. choose from multiple programming paradigms, languages and algorithms to solve a given problem adequately;
7. describe the fundamental theory of computation and computability;
8. apply the necessary mathematical methods;
9. recognize the context in which computer systems operate, including interactions with people and the physical world;
10. describe the state of published knowledge in their field or a specialization within it;
11. analyze and model real-life scenarios in organizations and industries using contemporary techniques of computer science, also taking methods and insights of other disciplines into account;
12. appropriately communicate solutions of problems in computer science in both spoken and written form to specialists and non-specialists;
13. draw scientifically founded conclusions that consider social, professional, scientific, and ethical aspects;
14. work effectively in a diverse team and take responsibility in a team;
15. take responsibility for their own learning, personal and professional development and role in society, reflecting on their practice and evaluating critical feedback;
16. adhere to and defend ethical, scientific, and professional standards.

1.4 Career Options and Support

Computer science is one of the key disciplines of the 21st century, which affects almost all modern industries. Consequently, the possible career paths are very broad for graduates with a computer science degree and the job market is highly favorable. The job market includes jobs such as software engineer, system integrator, information systems manager, data analyst, database administrator, application developer, cyber security analyst, IT consultant, and system analyst.

Graduates of the Computer Science program at Constructor University have obtained positions in companies of the information technology sector such as Amazon, Cleversoft, Facebook, Google, Microsoft, SAP, Skype, 360 Treasury Systems, Twitter, Research Gate, and VMware, as well as within companies that use information technology extensively such as the BMW Group, Deutsche Bank, KPMG, and Uber. Some graduates have founded their own companies such as Deep Web Solutions GmbH, Take Off Labs, and techOS GmbH.

Past graduates have also chosen to continue their education by enrolling into graduate programs at other German universities such as the RWTH Aachen, the Technical University Berlin, and the Technical University München; at other European universities such as the University of Amsterdam, the University of Cambridge, EPFL Lausanne, the University College London, the University of Oxford, and

ETH Zürich; or at international universities such as Carnegie Mellon University, Cornell University, and the University of Montreal.

The Career Service Center (CSC) helps students in their career development. It provides students with high-quality training and coaching in CV creation, cover letter formulation, interview preparation, effective presenting, business etiquette, and employer research as well as in many other aspects, thus helping students identify and follow up on rewarding careers after graduating from Constructor University. Furthermore, the Alumni Office helps students establish a long-lasting and worldwide network which provides support when exploring job options in academia, industry, and elsewhere.

1.5 Admission Requirements

Admission to Constructor University is selective and based on a candidate's school and/or university achievements, recommendations, self-presentation, and performance on standardized tests. Students admitted to Constructor University demonstrate exceptional academic achievements, intellectual creativity, and the desire and motivation to make a difference in the world.

The following documents need to be submitted with the application:

- Recommendation Letter (optional)
- Official or certified copies of high school/university transcripts
- Educational History Form
- Standardized test results (SAT/ACT) if applicable
- Motivation statement
- ZeeMee electronic resume (optional)
- Language proficiency test results (TOEFL Score: 90, IELTS: Level 6.5 or equivalent)

Formal admission requirements are subject to higher education law and are outlined in the Admission and Enrollment Policy of Constructor University.

For more detailed information about the admission visit: https://constructor.university/admission-aid/application-information-undergraduate

1.6 More Information and contacts

For more information, please contact the study program chair:

Name: Prof. Dr. Jürgen Schönwälder

Email: jschoenwaelder@constructor.university

or visit our program website: https://constructor.university/programs/undergraduate-education/computer-science

For more information on Student Services please visit:

Student services | Constructor University

2 The Curricular Structure

2.1 General

The curricular structure provides multiple elements for enhancing employability, interdisciplinarity, and internationality. The unique CONSTRUCTOR Track, offered across all undergraduate study programs, provides comprehensive tailor-made modules designed to achieve and foster career competency. Additionally, a mandatory internship of at least two months after the second year of study and the possibility to study abroad for one semester give students the opportunity to gain insight into the professional world, apply their intercultural competences and reflect on their roles and ambitions for employment and in a globalized society.

All undergraduate programs at Constructor University are based on a coherently modularized structure, which provides students with an extensive and flexible choice of study plans to meet the educational aims of their major as well as minor study interests and complete their studies within the regular period.

The framework policies and procedures regulating undergraduate study programs at Constructor University can be found on the website (https://constructor.university/student-life/student-services/university-policies).

2.2 The Constructor University 4C Model

Constructor University offers study programs that comply with the regulations of the European Higher Education Area. All study programs are structured according to the European Credit Transfer System (ECTS), which facilitates credit transfer between academic institutions. The three-year undergraduate programs involve six semesters of study with a total of 180 ECTS credit points (CP). The undergraduate curricular structure follows an innovative and student-centered modularization scheme, the 4C Model. It groups the disciplinary content of the study program in three overarching themes, CHOICE-CORE-CAREER according to the year of study, while the university-wide CONSTRUCTOR Track is dedicated to multidisciplinary content dedicated to methods as well as intellectual skills and is integrated across all three years of study. The default module size is 5 CP, with smaller 2.5 CP modules being possible as justified exceptions, e.g., if the learning goals are more suitable for 2.5 CP and the overall student workload is balanced.

2.2.1 Year 1 – CHOICE

The first study year is characterized by a university-specific offering of disciplinary education that builds on and expands upon the students' entrance qualifications. Students select introductory modules for a total of 45 CP from the CHOICE area of a variety of study programs, of which 15-45 CP will belong to their intended major. A unique feature of our curricular structure allows students to select their major freely upon entering Constructor University. The team of Academic Advising Services offers curriculum counseling to all Bachelor students independently of their major, while Academic Advisors, in their capacity as contact persons from the faculty, support students individually in deciding on their major study program.

To pursue Computer Science as a major, the following CHOICE modules (30 CP) need to be taken as mandatory (m) modules:

- CHOICE Module: Programming in C and C++ (m, 7.5 CP)
- CHOICE Module: Algorithms and Data Structures (m, 7.5 CP)
- CHOICE Module: Mathematical Foundations of Computer Science (m, 7.5 CP)
- CHOICE Module: Digital Systems and Computer Architecture (m, 7.5 CP)

The first two modules, Programming in C and C++ and Algorithms and Data Structures, introduce students to imperative and object-oriented programming and basic algorithms and data structures. The Mathematical Foundations of Computer Science module covers mathematical concepts like boolean algebra, propositional and predicate logic, abstract algebra, and graph theory. Students learn to work with formal notations and how to construct proofs. Starting with elementary digital gates, the Digital Systems and Computer Architecture module develops an understanding of how the hardware components of a computer system work. Students learn programming at the machine instruction level.

The remaining CHOICE modules (15 CP) can be selected in the first year of studies according to interest and/or with the aim to allow a change of major up until the beginning of the second year, when the major choice becomes fixed. Students not taking up a minor take the Development in JVM Languages module (7.5 CP) in the second semester.

Students can still change to another major at the beginning of their second year of studies if they have taken the corresponding mandatory CHOICE modules in their first year of studies. All students must participate in an entry advising session with their Academic Advisors to learn about their major change options and consult their Academic Advisor prior to changing their major.

Students that would like to retain a further option are strongly recommended to additionally register for the CHOICE modules of one of the following study programs in their first year:

- International Relations: Politics and History (IRPH)
  CHOICE Module: Introduction to International Relations Theory (7.5 CP)
  CHOICE Module: Introduction to Modern European History (7.5 CP)

- Integrated Social and Cognitive Psychology (ISCP)
  CHOICE Module: Essentials of Cognitive Psychology (7.5 CP)
  CHOICE Module: Essentials of Social Psychology (7.5 CP)

- Robotics and Intelligent Systems (RIS)
  CHOICE Module: Mathematical and Physical Foundations of Robotics I (m, 7.5 CP)
  CHOICE Module: Mathematical and Physical Foundations of Robotics II (m, 7.5 CP)

- Software, Data and Technology (SDT)
  CHOICE Module: Core Algorithms and Data Structures (m, 7.5 CP)
  CHOICE Module: Development in JVM Languages (m, 7.5 CP)

The module descriptions can be found in the respective Study Program Handbook.

2.2.2 Year 2 – CORE

In their second year, students take a total of 45 CP from a selection of in-depth, discipline-specific CORE modules. Building on the introductory CHOICE modules and applying the methods and skills acquired so far (see 2.3.1), these modules aim to expand the students' critical understanding of the key theories, principles, and methods in their major for the current state of knowledge and best practice.

To pursue Computer Science as a major, at least the following mandatory CORE modules (30 CP) need to be taken:

- CORE Module: Databases (m, 7.5 CP)
- CORE Module: Software Engineering (m, 7.5 CP)
- CORE Module: Operating Systems (m, 7.5 CP)
- CORE Module: Automata, Computability, and Complexity (m, 7.5 CP)

Students decide to complement their studies by taking the discipline-specific mandatory elective (me) CORE modules (15 CP):

- CORE Module: Functional Programming (me, 5 CP)
- CORE Module: Legal and Ethical Aspects of Computer Science (me, 2.5 CP)
- CORE Module: Machine Learning (me, 5 CP)
- CORE Module: Academic Skills in Computer Science (me, 2.5 CP)

or substitute these modules with CORE modules from other study programs with the aim of pursuing a minor in a second field.

Computer Science students can take CORE modules (or more advanced Specialization modules) from a second discipline, which allows them to incorporate a minor study track into their undergraduate education, within the 180 CP required for a bachelor's degree. The educational aims of a minor are to broaden the students' knowledge and skills, support the critical reflection of statements in complex contexts, foster an interdisciplinary approach to problem-solving, and to develop an individual academic and professional profile in line with students' strengths and interests. This extra qualification will be highlighted in the transcript.

The Academic Advising Coordinator, Academic Advisor, and the Study Program Chair of the minor study program support students in the realization of their minor selection; consultation with the Academic Advisor is mandatory when choosing a minor.

As a rule, obtaining a minor requires Computer Science students to

- select two CHOICE modules (15 CP) from the desired minor program in the first year and
- substitute the mandatory elective Computer Science CORE modules Functional Programming (me, 5 CP), Legal and Ethical Aspects of Computer Science (me, 2.5 CP), Machine Learning (me, 5 CP), and Academic Skills in CS (me, 2.5 CP) in the second year with the default minor CORE modules of the minor study program. Note that the substituted CORE modules can still be selected in the third year as specialization modules.

The requirements for each specific minor are described in the handbook of the study program offering the minor (Chapter 3.2) and are marked in the respective Study and Examination Plans. For an overview of accessible minors, please check the Major/Minor Combination Matrix which is published at the beginning of each academic year.

Note: Students pursuing Computer Science as a major cannot pursue Software, Data and Technology (SDT) or Data Science as a minor.

2.2.3 Year 3 – CAREER

During their third year, students prepare and make decisions about their career path after graduation. To explore available choices and to gain professional experience, students undertake a mandatory summer internship. The third year of studies allows Computer Science students to take Specialization modules within their discipline, but also focuses on the responsibility of students beyond their discipline (see CONSTRUCTOR Track).

The fifth semester also opens a mobility window for a diverse range of study abroad options. Finally, the sixth semester is dedicated to fostering the students' research experience by involving them in an extended Bachelor thesis project.

2.2.3.1 Internship / Start-up and Career Skills Module

As a core element of Constructor University's employability approach students are required to engage in a mandatory two-month internship of 15 CP that will usually be completed during the summer between the second and third years of study. This gives students the opportunity to gain first-hand practical experience in a professional environment, apply their knowledge and understanding in a professional context, reflect on the relevance of their major to employment and society, reflect on their own role in employment and society, and find a professional orientation. The internship can also establish valuable contacts for the students' Bachelor's thesis project, for the selection of a Master program graduate school or further employment after graduation. This module is complemented by career advising and several career skills workshops throughout all six semesters that prepare students for the transition from student life to professional life. As an alternative to the full-time internship, students interested in setting up their own company can apply for a start-up option to focus on developing of their business plans.

For further information, please contact the Career Service Center (CSC) (https://constructor.university/student-life/career-services).

2.2.3.2 Specialization Modules

In the third year of their studies, students take 15 CP from major-specific or major-related, advanced Specialization modules to consolidate their knowledge and to be exposed to state-of-the-art research in the areas of their interest. This curricular component is offered as a portfolio of modules, from which students can make free selections during their 5th and 6th semester. The default specialization module size is 5 CP, with smaller 2.5 CP modules being possible as justified exceptions.

To pursue CS as a major, 15 CP from the following mandatory elective Specialization Modules need to be taken:

- CS Specialization: Computer Graphics (me, 5 CP)
- CS Specialization: Image Processing (me, 5 CP)
- CS Specialization: Distributed Algorithms (me, 5 CP)
- CS Specialization: Web Application Development (me, 5 CP)
- CS Specialization: Computer Networks (me, 5 CP)
- CS Specialization: Secure and Dependable Systems (me, 5 CP)
- CS Specialization: Ethical Hacking and Offensive Security (me, 5 CP)
- CS Specialization: Security Monitoring and Incident Response (me, 5 CP)
- CS Specialization: Linux Kernel Internals (me, 5 CP)

- CS Specialization: Advanced Operating Systems (me, 5 CP)
- RIS CORE: Computer Vision (me, 5 CP)
- RIS Specialization: Human Computer Interaction (me, 5 CP)
- RIS CORE: Artificial Intelligence (me, 5 CP)
- RIS CORE: Robotics (me, 5 CP)
- ECE Specialization: Digital Design (me, 5 CP)
- ECE CORE: Information Theory (me, 5 CP)
- SDT CORE: Functional Programming (me, 5 CP)

Students pursuing a minor in a second field of studies can additionally select Specialization Modules from:

- CS CORE: Legal and Ethical Aspects of Computer Science (me, 2.5 CP)
- CS CORE: Academic Skills in Computer Science (me, 2.5 CP)
- CS CORE: Machine Learning (me, 5 CP)

To obtain a Specialization in Cybersecurity students must successfully pass these specific modules:

- Secure and Dependable Systems (m, 5 CP)
- Ethical Hacking and Offensive Security (m, 5 CP)
- Security Monitoring and Incident Response (m, 5 CP)

In addition, students must write a Bachelor thesis on a topic related to Cybersecurity.

2.2.3.3 Specializations

Students can specialize on a specific specialization topic in their 3rd year by (i) obtaining 15 CP in specialization modules that belong to a defined specialization topic and (ii) writing their Bachelor thesis on a topic related to the specialization topic. The topic of a completed specialization will be listed on the diploma supplement and the transcript.

The following specialization topics are currently defined:

- Specialization Cybersecurity

  The increasing relevance of cybersecurity in both industry and research requires specialized training for future computer scientists. The specialization in Cybersecurity requires that students obtain 15 CP specialization credits in the following specialization modules and write their thesis on a topic related to cybersecurity:

  - Secure and Dependable Systems (m, 5 CP)
  - Ethical Hacking and Offensive Security (m, 5 CP)
  - Security Monitoring and Incident Response (m, 5 CP)

2.2.3.4 Study Abroad

Students have the opportunity to study abroad for a semester to extend their knowledge and abilities, broaden their horizons and reflect on their values and behavior in a different context as well as on their role in a global society. For a semester abroad (usually the 5th semester), modules related to the major with a workload equivalent to 22.5 CP must be completed. Modules recognized as study abroad CP need to be pre-approved according to Constructor University study abroad procedures. Several exchange programs allow students to directly enroll at prestigious partner institutions worldwide. Constructor University's participation in Erasmus+, the European Union's exchange program, provides an exchange semester at a number of European universities that include Erasmus study abroad funding.

For further information, please contact the International Office (https://constructor.university/student-life/study-abroad/international-office).

Computer Science students pursuing a study abroad in their 5th semester are required to select their modules at the study abroad partners such that they can be used to substitute between 10-15 CP of major-specific Specialization modules and between 5-15 CP of modules equivalent to the non-disciplinary New Skills modules (see CONSTRUCTOR Track). In their 6th semester, according to the study plan, returning study-abroad students complete the Bachelor Thesis/Seminar module (see next section), they take any missing Specialization modules to reach the required 15 CP in this area, and they take any missing New Skills modules to reach 15 CP in this area.

2.2.3.5 Bachelor Thesis/Seminar Module

This module is a mandatory graduation requirement for all undergraduate students. It consists of two module components in the major study program guided by a Constructor University faculty member: the Bachelor Thesis (12 CP) and a Seminar (3 CP). The title of the thesis will appear on the students' transcripts.

Within this module, students apply the knowledge skills, and methods they have acquired in their major discipline to become acquainted with actual research topics, ranging from the identification of suitable (short-term) research projects, preparatory literature searches, the realization of discipline-specific research, and the documentation, discussion, and interpretation of the results.

With their Bachelor Thesis students demonstrate mastery of the contents and methods of the computer science research field. Furthermore, students show the ability to analyze and solve a well-defined problem with scientific approaches, a critical reflection of the status quo in scientific literature, and the original development of their own ideas. With the permission of a Constructor University Faculty Supervisor, the Bachelor Thesis can also have an interdisciplinary nature. In the seminar, students present and discuss their theses in a course environment and reflect on their theoretical or experimental approach and conduct. They learn to present their chosen research topics concisely and comprehensively in front of an audience and to explain their methods, solutions, and results to both specialists and non-specialists.

2.3 The CONSTRUCTOR Track

The CONSTRUCTOR Track is another important feature of Constructor University's educational model. The Constructor Track runs orthogonal to the disciplinary CHOICE, CORE, and CAREER modules across all study years and is an integral part of all undergraduate study programs. It provides an intellectual tool kit for lifelong learning and encourages the use of diverse methodologies to approach cross-disciplinary problems. The CONSTRUCTOR track contains Methods, New Skills and German Language and Humanities modules.

2.3.1 Methods Modules

Methods such as mathematics, statistics, programming, data handling, presentation skills, academic writing, and scientific and experimental skills are offered to all students as part of the Methods area in their curriculum. The modules that are specifically assigned to each study program equip students with transferable academic skills. They convey and practice specific methods that are indispensable for each students' chosen study program. Students are required to take 20 CP in the Methods area. The size of all Methods modules is 5 CP.

To pursue Computer Science as major, the following Methods module (5 CP) is mandatory

- Methods Module: Elements of Linear Algebra (me, 5 CP)
- Methods Module: Elements of Calculus (me, 5 CP)
- Methods Module: Probability and Random Processes (m, 5 CP)

Students who have a strong mathematical background can also choose Matrix Algebra and Advanced Calculus I and II (me, 5 CP each) instead of Elements of Linear Algebra and Elements of Calculus.

For the remaining 5 CP CS students can choose between the Methods modules

- Methods Module: Numerical Methods (me, 5 CP)
- Methods Module: Statistics and Data Analytics (me, 5 CP)

2.3.2 New Skills Modules

This part of the curriculum constitutes an intellectual and conceptual tool kit that cultivates the capacity for a particular set of intellectual dispositions including curiosity, imagination, critical thought, and transferability. It nurtures a range of individual and societal capacities, such as self-reflection, argumentation and communication. Finally, it introduces students to the normative aspects of inquiry and research, including the norms governing sourcing, sharing, withholding materials and research results as well as others governing the responsibilities of expertise as well as the professional point of view

All students are required to take the following modules in their second year:

- New Skills Module: Logic (m, 2.5 CP)
- New Skills Module: Causation and Correlation (m, 2.5 CP)

These modules will be offered with two different perspectives, one of which the students can choose. The module perspectives are independent modules which examine the topic from different points of view. Please see the module description for more details.

In the third year, students take three 5 CP modules that build upon previous modules in the track and are partially constituted by modules that are more closely linked to each student's disciplinary field of study. The following module is mandatory for all students:

- New Skills Module: Argumentation, Data Visualization and Communication (m, 5 CP)

This module will also be offered with two different perspectives of which the students can choose.

In their fifth semester, students may choose between:

- New Skills Module: Linear Model/Matrices (me, 5 CP) and
- New Skills Module: Complex Problem Solving (me, 5 CP).

The sixth semester also contains the choice between two modules, namely:

- New Skills Module: Agency, Leadership and Accountability (me, 5 CP) and
- New Skills Module: Community Impact Project (me, 5 CP).

Students who study abroad during the fifth semester and are not substituting the mandatory Argumentation, Data Visualization and Communication module, are required to take this module during their sixth semester. Students who remain on campus are free to take the Argumentation, Data Visualization and Communication module in person in either the fifth or sixth semester as they prefer.

2.3.3 German Language and Humanities Modules

German language abilities foster students' intercultural awareness and enhance their employability in their host country. They are also beneficial for securing mandatory internships (between the 2nd and 3rd year) in German companies and academic institutions. Constructor University supports its students in acquiring basic as well as advanced German skills in the first year of the Constructor Track. Non-native speakers of German are encouraged to take 2 German modules (2.5 CP each), but are not obliged to do so. Native speakers and other students not taking advantage of this offering take alternative modules in Humanities in each of the first two semesters:

- Humanities Module: Introduction to Philosophical Ethics (2.5 CP)
- Humanities Module: Introduction to the Philosophy of Science (2.5 CP)
- Humanities Module: Introduction to Visual Culture (2.5 CP)

3 Computer Science as a Minor

3.1 Qualification Aims

Students obtaining a minor in Computer Science learn the basic principles of software development and modern software development processes. They acquire an understanding of how modern information systems are designed and implemented. Upon completion of the minor, they will have obtained sufficient knowledge about computer science concepts such that they can effectively work together with professionals with a Computer Science degree. Students obtaining a minor in Computer Science can help to drive digitalization processes, as they can effectively translate requirements of the field of their major into terminology and technology used by Computer Science professionals. Students majoring in a technical discipline can obtain a minor to strengthen their understanding of how to use software and hardware components effectively, thereby achieving efficient solutions for problems in their domain.

3.1.1 Intended Learning Outcomes

With a minor in Computer Science, students will be able to

1. develop solutions to problems in computer science in close collaboration with computer science professionals;
2. communicate requirements appropriately to their audience and understand computer science aspects of a solution;
3. apply programming concepts and basic algorithms to solve software development problems of moderate complexity in an adequate way;
4. understand how design choices impact the efficiency of solutions.

3.2 Module Requirements

A minor in Computer Science requires 30 CP. The default option to obtain a minor in Computer Science is marked in the Study and Examination Plan in chapter 6. It includes the following mandatory CHOICE and CORE modules:

- CHOICE Module: Programming in C and C++ (m, 7.5 CP)
- CHOICE Module: Algorithms and Data Structures (m, 7.5 CP)
- CORE Module: Databases (m, 7.5 CP)
- CORE Module: Software Engineering (m, 7.5 CP)

Upon the consultation with the Academic Advisor and approval by the CS Study Program Coordinator, individual CORE modules from the default minor can be replaced by other advanced modules (CORE or Specialization) from the CS major.

3.3 Degree

After successful completion, the minor in Computer Science will be listed on the final transcript under PROGRAM OF STUDY and BA/BSc – [name of the major] as “(Minor: Computer Science).”

4 Computer Science Undergraduate Program Regulations

4.1 Scope of these Regulations

The regulations in this handbook are valid for all students who entered the Computer Science undergraduate program at Constructor University in Fall 2025. In case of conflict between the regulations in this handbook and the general policies for Bachelor Studies, the latter apply (see https://constructor.university/student-life/student-services/university-policies).

In exceptional cases, certain necessary deviations from the regulations of this study handbook might occur during the course of study (e.g., change of the semester sequence, assessment type, or the teaching mode of courses).

Updates to Study Program Handbooks are based on the policies approved by the Academic Senate on substantial and nonsubstantial changes to study programs. Students are integrated in the decision-making process through their respective committee representatives. All students affected by the changes will be properly informed.

In general, Constructor University therefore reserves the right to change or modify the regulations of the program handbook also after its publication at any time and in its sole discretion.

4.2 Degree

Upon successful completion of the study program, students are awarded a Bachelor of Science degree in Computer Science.

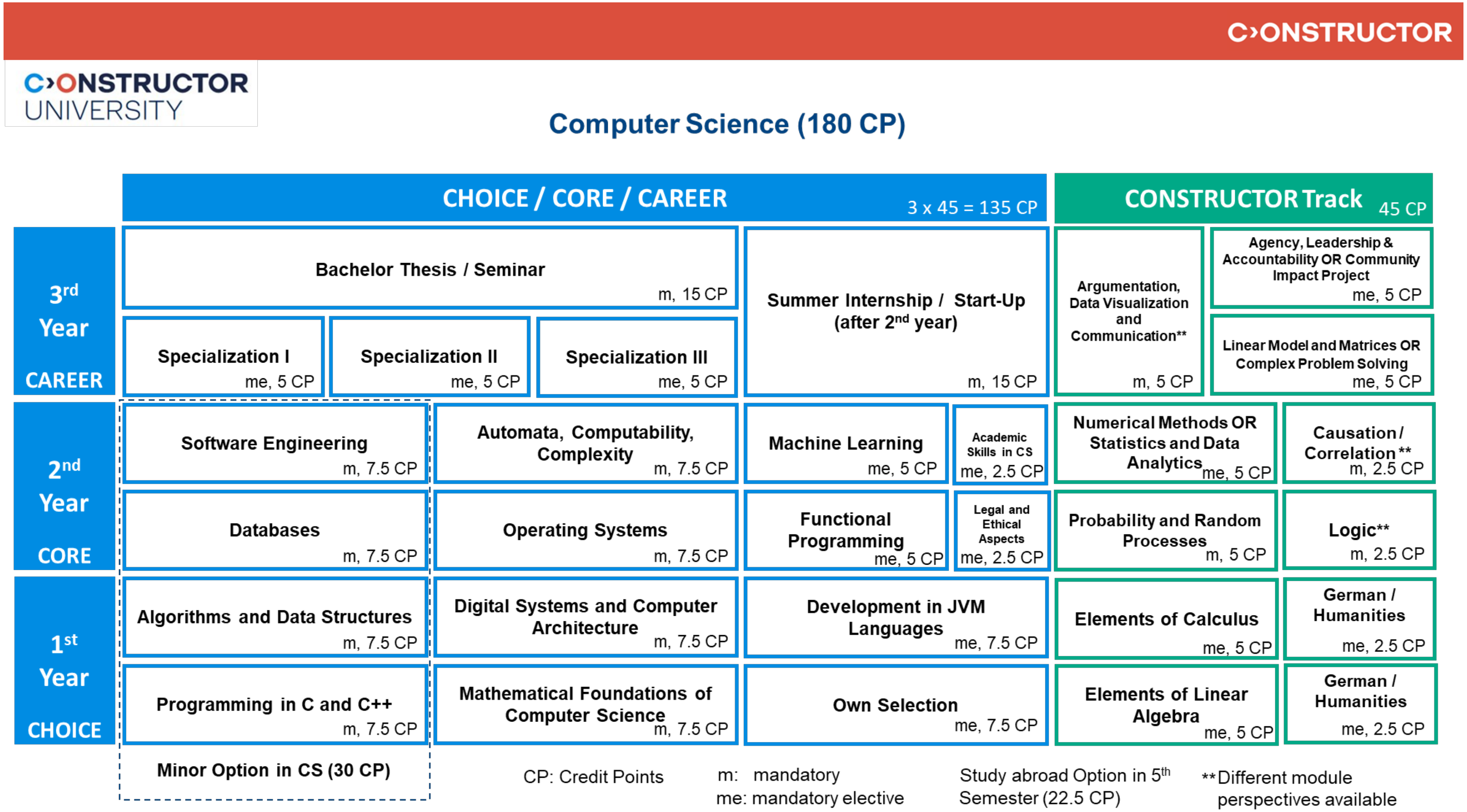
4.3 Graduation Requirements

To graduate, students need to obtain 180 CP. In addition, the following graduation requirements apply:

- Students need to complete all mandatory components of the program as indicated in the Study and Examination Plan in chapter 6 of this handbook.
- Students graduating in Computer Science without a minor have to obtain
    - 20 CP in Methods modules (mathematics),
    - 97.5 CP in Computer Science modules, and
    - 15 CP for the Bachelor thesis and the associated seminar.
- Students graduating in Computer Science with a minor in a second discipline have to obtain
    - 20 CP in Methods modules (mathematics),
    - 75 CP in Computer Science modules, and
    - 30 CP for the minor of choice
- Students have to obtain 15 CP for their Internship.
- Also, obtain a total of 25 CP in the general part of the Constructor Track

5 Schematic Study Plan for Computer Science

Figure 2 shows schematically the sequence and types of modules required for the study program. A more detailed description, including the assessment types, is given in the Study and Examination Plans in the following section.



6 Study and Examination Plan

Computer Science (CS) BSc

Matriculation Fall 2025

	Program-Specific Modules	Type	Assessment	Period	Status[1]	Sem.	CP
Year 1 - CHOICE *Take the mandatory CHOICE modules listed below, this is a requirement for the Computer Science program.*							**45**
	Unit: Programming, Algorithms, and Data Structures (default minor choice modules)						**15**
CH-230	**Module: Programming in C and C++**				**m**	**1**	**7.5**
CH-230-A	Programming in C and C++	Lecture	Written examination	Examination period			5
CH-230-B	Programming in C and C++ Tutorial	Tutorial	Program Code	During the semester			2.5
CH-231	**Module: Algorithms and Data Structures**				**m**	**2**	**7.5**
CH-231-A	Algorithms and Data Structures	Lecture	Written examination	Examination period			
	Unit: Computer Science, Robotics, and Intelligent Systems						**15**
CH-233	**Module: Mathematical Foundations of Computer Science**				**m**	**1**	**7.5**
CH-233-A	Mathematical Foundations of Computer Science	Lecture	Written examination	Examination period			5
CH-233-B	Mathematical Foundations of Computer Science Tutorial	Tutorial					2.5
CH-234	**Module: Digital Systems and Computer Architecture**				**m**	**2**	**7.5**
CH-234-A	Digital Systems and Computer Architecture	Lecture	Written examination	Examination period			5
CH-234-B	Digital Systems and Computer Architecture Tutorial	Tutorial		During the semester			2.5
	Unit: CHOICE (own selection)					**1/2**	**15**
Students take two further CHOICE modules from those offered for all other study programs[2] if they intend to pursue a minor. If no minor will be pursued, take SDT-103 and one additional Choice module from another study program							
SDT-103	**Module:Development in JVM Languages**				**me**	**2**	**7.5**
SDT-103-A	Development in JVM Languages	Lecture	Written examination	Examination period			2.5
SDT-103-B	Development in JVM Languages	Tutorial	Program Code	During the semester			5
Year 2 - CORE *Take all CORE modules listed below or replace mandatory elective (me) modules with default CORE modules from minor study program*							**45**
	Unit: Advanced Computer Science I (default minor advanced modules)						**15**
CO-560	**Module: Databases**				**m**	**3**	**7.5**
CO-560-A	Databases	Lecture	Written examination	Examination period			5
CO-560-B	Databases- Project	Project	Project assessment	During the semester			2.5
CO-561	**Module: Software Engineering**				**m**	**4**	**7.5**
CO-561-A	Software Engineering	Lecture	Written examination	Examination period			2.5
CO-561-B	Software Engineering Project	Project	Project assessment	During the semester			5
	Unit: Advanced Computer Science II						**15**
CO-562	**Module: Operating Systems**				**m**	**3**	**7.5**
CO-562-A	Operating Systems	Lecture	Written examination	Examination period			
CO-563	**Module: Automata, Computability, and Complexity**				**m**	**4**	**7.5**
CO-563-A	Automata, Computability, and Complexity	Lecture	Written examination	Examination period			
	Unit: Advanced Computer Science III						**15**
SDT-202	**Module:-Functional Programming**				**me**	**3**	**5**
SDT-202-A	Functional Programming	Lecture	Written examination	Examination period			2.5
SDT-202-B	Functional Programming Tutorial	Tutorial	Program code	During the semester			2.5
CO-565	**Module: Legal and Ethical Aspects of Computer Science**				**me**	**3**	**2.5**
CO-565-A	Legal and Ethical Aspects of Computer Science	Lecture	Poster presentation	Examination period			
CO-541	**Module: Machine Learning**				**m**	**4**	**5**
CO-541-A	Machine Learning	Lecture	Written examination	Examination period			
CO-567-A	**Module: Academic Skills in Computer Science**				**me**	**4**	**2.5**
CO-567-A	Academic Skills in Computer Science	Seminar	Project assessment	Examination period			

	CONSTRUCTOR Track Modules (General Education)	Type	Assessment	Period	Status[1]	Sem.	CP
Year 1							**15**
	Unit: Methods						**10**
CTMS-MAT-24	**Module: Elements of Linear Algebra**				**me**	**1**	**5**
CTMS-24	Elements of Linear Algebra	Lecture	Written examination	Examination period			
CTMS-MAT-25	**Module: Elements of Calculus**				**me**	**2**	**5**
CTMS-25	Elements of Calculus	Lecture	Written examination	Examination period			
Students who have a strong mathematical background can also choose the following instead of CTMS-MAT-22 and CTMS-MAT-23:							
CTMS-MAT-22	**Module: Matrix Algebra & Advanced Calculus I**				**me**	**1**	**5**
CTMS-22	Matrix Algebra & Advanced Calculus I	Lecture	Written examination	Examination period			
CTMS-MAT-23	**Module: Matrix Algebra & Advanced Calculus II**				**me**	**2**	**5**
CTMS-23	Matrix Algebra & Advanced Calculus II	Lecture	Written examination	Examination period			
	Unit: German Language and Humanities (choose one module for each sememster)						**5**
German is default language and open to Non-German speakers (on campus and online).[4]							
CTLA-xxx	**Module: Language 1**				**me**	**1**	**2,5**
CTLA-xxx	Language 1	Seminar	Various	Various	me		
CTLA-xxx	**Module: Language 2**				**me**	**2**	**2,5**
CTLA-xxx	Language 2	Seminar	Various	Various	me		
CTHU-HUM-001	**Humanities Module: Introduction to Philosophical Ethics**				**me**	**1**	**2,5**
CTHU-001	Introduction into Philosophical Ethics	Lecture (online)	Written examination	Exam period	me		
CTHU-HUM-002	**Humanities Module: Introduction to the Philosophy of Science**				**me**	**2**	**2,5**
CTHU-002	Introduction to the Philosophy of Science	Lecture (online)	Written examination	Exam period	me		
CTHU-HUM-003	**Humanities Module: Introduction to Visual Culture**				**me**	**2**	**2,5**
CTHU-003	Introduction to Visual Culture	Lecture (online)	Written examination	Exam period	me		
Year 2							**15**
	Unit: Methods						**10**
CTMS-MAT-12	**Module: Probability and Random Processes**				**m**	**3**	**5**
CTMS-12	Probability and Random Processes	Lecture	Written examination	Examination period			5
Take one of the two listed mandatory elective methods modules:							
CTMS-MET-21	**Module: Statistics and Data Analysis**				**me**	**4**	**5**
CTMS-21	Statistics and Analysis	Lecture	Written examination	Examination period			
CTMS-MAT-13	**Module: Numerical Methods**				**me**	**4**	**5**
CTMS-13	Numerical Methods	Lecture	Written examination	Examination period			
	Unit: New Skills						**5**
Choose one of the two modules							
CTNS-NSK- 01	**Module: Logic (perspective I)**				**me**	**3**	**2,5**
CTNS-01	Logic (perspective I)	Online Lecture	Written Examination	Examination period			2,5
CTNS-NSK-02	**Module: Logic (perspective II)**						
CTNS-02	Logic (perspective II)	Online Lecture	Written Examination	Examination period			2,5
Choose one of the two modules							
CTNS-NSK-03	**Module: Correlation and Causation (perspective I)**				**me**	**4**	**2,5**
CTNS-03	Correlation and Causation (perspective I)	Online Lecture	Written Examination	Examination period			2,5
CTNS-NSK-04	**Module: Correlation and Causation (perspective II)**				**me**	**4**	**2,5**
CTNS-04	Correlation and Causation (perspective II)	Online Lecture	Written Examination	Examination period			2,5

Figure 3: Study and Examination Plan

Year 3 - CAREER							45
CA-INT-900	**Module: Summer Internship**				**m**	**4/5**	**15**
CA-INT-900-0	Summer Internship		Report/Business	During the 5th semester			
CA-CS-800	**Module: Thesis / Seminar CS**				**m**	**6**	**15**
CA-CS-800-T	Thesis CS	Thesis	Thesis	15th of May			12
CA-CS-800-S	Seminar CS	Seminar	Presentation	During the semester			3
	Unit: Specialization CS				**m**	**5/6**	**15**
Take a total of 15 CP Specialization Modules							
CA-S-CS-801	**Module: Computer Graphics**				**me**	**5**	**5**
CA-S-CS-801-A	Computer Graphics	Lecture	Written examination	Examination period			
CA-S-CS-802	**Module: Image Processing**				**me**	**6**	**5**
CA-S-CS-802-A	Image Processing	Lecture	Written examination	Examination period			5
CA-S-CS-803	**Module: Distributed Algorithms**				**me**	**6**	**5**
CA-S-CS-803-A	Distributed Algorithms	Lecture	Written examination	Examination period			5
CA-S-CS-804	**Module: Web Application Development□**				**me**	**6**	**5**
CA-S-CS-804-A	Web Application Development	Lecture	Written examination	Examination period			2.5
CA-S-CS-804-B	Web Application Development	Project	Project assessment	During the semester			2.5
CO-564	**Module: Computer Networks**				**me**	**5**	**5**
CO-564-A	Computer Networks	Lecture	Written examination	Examination period			
CO-566	**Module: Secure and Dependable Systems**				**me**	**5**	**5**
CO-566-A	Secure and Dependable Systems	Lecture	Written examination	Examination period			
CA-S-CS-807	**Module: Security Monitoring and Incident Response**				**me**	**6**	**5**
CA-S-CS-807	Security Monitoring and Incident Response	Lecture	Written examination	Examination period			
CA-CS-808	**Module: Ethical Hacking and Offensive Security**				**me**	**6**	**5**
CA-CS-808	Ethical Hacking and Offensive Security	Lab	Lab report	During the semester			
CA-S-CS-809	**Linux Kernel Internals**				**me**	**6**	**5**
CA-CS-809	Linux Kernel Internals	Lecture	Written examination	Examination period			
CA-S-CS-810	**Advanced Operating Systems**				**me**	**5**	**5**
CA-CS-810	Advanced Operating Systems	Lecture	Written examination	Examination period			
CA-S-xxx	**Specialization electives (from RIS, ECE, DE study programs)³**	Lecture	Written examination	Examination period	**me**	**5/6**	**5**
Total CP							

							15
	Unit: New Skills						**10**
Choose one of the two modules							
CTNS-NSK-05	**Module: Linear Model / Matrices**				**me**	**5**	**5**
CTNS-05	Linear Model/ Matrices	Seminar (online)	Written examination	Examination period			5
CTNS-NSK-06	**Module: Complex Problem Solving**				**me**	**5**	**5**
CTNS-06	Complex Problem Solving	Lecture (online)	Written examination	Examination period			5
Choose one of the two modules							5
CTNS-NSK-07	**Module: Argumentation, Data Visualization and Communication**				**me**	**5/6**	**5**
CTNS-07	Argumentation, Data Visualization and Communication (perspective I)	Online Lecture	Written examination	Examination period		**5**	5
CTNS-NSK-08	**Module: Argumentation, Data Visualization and Communication**				**me**	**5/6**	**5**
CTNS-08	Argumentation, Data Visualization and Communication (perspective II)	Online Lecture	Written examination	Examination period		**6**	5
Choose one of the two modules							
CTNS-NSK	**Module: Agency, Accountability & Leadership**				**me**	**6**	**5**
CTNS-09	Agency, Accountability & Leadership	Lecture (online)	Written examination	Examination period			5
CTNS-CIP-10	**Module: Community Impact Project**				**me**	**5/6**	**5**
CTNS-CIP-10	Community Impact Project	Project	Project	During the Sememster			5
							180

¹ Status (m = mandatory, me = mandatory elective)

² For a full listing of all CHOICE / CORE / CAREER / CONSTRUCTOR Track modules please consult the CampusNet online catalogue and /or the study program handbooks.

³ For details please see the CS program handbook.

⁴ German native speakers will have alternatives to the language courses (in the field of Humanities).

7 Computer Science Modules

7.1 Programming in C and C++

Module Name	Programming in C and C++
Module Code	2025-CH-230
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-CS-BSc 1 - 2025-RIS-BSc 1 - 2025-ECE-BSc 1 - 2025-SDT-BSc 1 - 2025-Minor-RIS-BSc 1 - 2025-Minor-CS-BSc 1 - 2025-Minor-Software Development 1 Mandatory Elective status for: None
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Dr. Kinga Lipskoch

Forms of Learning and Teaching	
Exam Preparation	20
Lecture	35
Tutorial	17
Independent Study	115
Workload Hours	187 hours

Module Components	Number	Type	CP
Programming in C and C++	CH-230-A	Lecture	5
Programming in C and C++ - Tutorial	CH-230-B	Tutorial	2.5

Module Description

This course offers an introduction to programming using the programming languages C and C++. After a short overview of the program development cycle (editing, preprocessing, compiling, linking, executing), the module presents the basics of C programming. Fundamental imperative programming concepts such as variables, loops, and function calls are introduced in a hands-on manner. Afterwards, basic data structures such as multidimensional arrays, structures, and pointers are introduced and dynamically allocated multidimensional arrays and linked lists and trees are used for solving simple practical problems. The relationships between pointers and arrays, pointers and structures, and pointers and functions are described, and they are illustrated using examples that also introduce recursive functions, file handling, and dynamic memory allocation.

The module then introduces basic concepts of object-oriented programming languages using the programming language C++ in a hands-on manner. Concepts such as classes and objects, data abstractions, and information hiding are introduced. C++ mechanisms for defining and using objects, methods, and operators are introduced and the relevance of constructors, copy constructors, and destructors for dynamically created objects is explained. Finally, concepts such as inheritance, polymorphism, virtual functions, and overloading are introduced. The learned concepts are applied by solving programming problems.

Recommended Knowledge

It is recommended that students install a suitable programming environment on their notebooks. It is recommended to install a Linux system such as Ubuntu, which comes with open-source compilers such as gcc and g++ and editors such as vim or emacs. Alternatively, the open-source Code: Blocks integrated development environment can be installed to solve programming problems

Usability and Relationship to other Modules

This module introduces the programming languages C and C++ and several other modules build on this foundation. Certain features of C++ such as templates and generic data structures and an overview of the standard template library will be covered in the Algorithms and Data Structures module.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain basic concepts of imperative programming languages such as variables, assignments, loops, and function calls.
2	Write	Write, test, and debug programs in the procedural programming language C using basic C library functions.
3	Demonstrate	Demonstrate how to use pointers to create dynamically allocated data structures such as linked lists.
4	Explain	Explain the relationship between pointers and arrays.
5	Illustrate	Illustrate basic object-oriented programming concepts such as objects, classes, information hiding, and inheritance.
6	Give	Give original examples of function and operator overloading and polymorphism.
7	Write	Write, test, and debug programs in the object-oriented programming language C++.

Indicative Literature

- Brian Kernighan, Dennis Ritchie: The C Programming Language, 2nd edition, PrenticeHall Professional Technical Reference, 1988.
- Steve Oualline: Practical C Programming, 3rd edition, O'Reilly Media, 1997.
- Bruce Eckel: Thinking in C++: Introduction to Standard C++, Prentice Hall, 2000.
- Bruce Eckel, Chuck Allison: Thinking in C++: Practical Programming, Prentice Hall, 2004.
- Bjarne Stroustrup: The C++ Programming Language, 4th edition, Addison Wesley, 2013.

- Michael Dawson: Beginning C++ Through Game Programming, 4th edition, Delmar Learning, 2014.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Programming in C and C++	Written Examination	120 minutes	67	45%	All theoretical ILOs
Programming in C and C++ - Tutorial	Program Code		33	45%	All Practical ILOs

Module Achievements: None

7.2 Algorithms and Data Structures

Module Name	Algorithms and Data Structures
Module Code	2025-CH-231
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-RIS-BSc 2 - 2025-CS-BSc 2 - 2025-Minor-CS-BSc 2 Mandatory Elective status for: None
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Dr. Kinga Lipskoch

Forms of Learning and Teaching	
Class Attendance	52.5
Independent Study	115
Exam Preparation	20
Workload Hours	187.5 hours

Module Components	Number	Type	CP
Algorithms and Data Structures	CH-231-A	Lecture	7.5

Module Description

Algorithms and data structures are the core of computer science. An algorithm is an effective description for calculations using a finite list of instructions that can be executed by a computer. A data structure is a concept for organizing data in a computer such that data can be used efficiently. This introductory module allows students to learn about fundamental algorithms for solving problems efficiently. It introduces basic algorithmic concepts; fundamental data structures for efficiently storing, accessing, and modifying data; and techniques that can be used for the analysis of algorithms and data structures with respect to their computational and memory complexities. The presented concepts and techniques form the basis of almost all computer programs.

Recommended Knowledge

Students should refresh their knowledge of the C and C++ programming language and be able to solve simple programming problems in C and C++. Students are expected to have a working programming environment.

Usability and Relationship to other Modules

Familiarity with basic algorithms and data structures is fundamental for almost all advanced modules in computer science. This module additionally introduces advanced concepts of the C++ programming language that are needed in advanced programming-oriented modules in the 2nd and 3rd years of the CS and RIS programs.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain asymptotic (time and memory) complexities and respective notations.
2	Able	Able to prove asymptotic complexities of algorithms.
3	Illustrate	Illustrate basic data structures such as arrays, lists, queues, stacks, trees, and hash tables.
4	Describe	Describe algorithmic design concepts and apply them to new problems.
5	Explain	Explain basic algorithms (sorting, searching, graph algorithms, computational geometry) and their complexities.
6	Summarize	Summarize and apply C++ templates and generic data structures provided by the standard C++ template library.

Indicative Literature

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein: Introduction to Algorithms, 3rd edition, MIT Press, 2009.
- Donald E. Knuth: The Art of Computer Programming: Fundamental Algorithms, volume 1, 3rd edition, Addison Wesley Longman Publishing, 1997.

Entry Requirements

Prerequisites	Programming in C and C++
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Algorithms and Data Structures	Written Examination	120 minutes	100	45%	1-6

Module Achievements: None

7.3 Mathematical Foundations of Computer Science

Module Name	Mathematical Foundations of Computer Science
Module Code	2025-CH-233
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-CS-BSc 1 - 2025-SDT-BSc 1 Mandatory Elective status for: None
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	35
Tutorial	17.5
Independent Study	115
Exam Preparation	20
Workload Hours	187.5 hours

Module Components	Number	Type	CP
Mathematical Foundations of Computer Science	CH-233-A	Lecture	5
Mathematical Foundations of Computer Science Tutorial	CH-233-B	Tutorial	2.5

Module Description

The module introduces students to the mathematical foundations of computer science. Students learn to reason logically and clearly. They acquire the skill to formalize arguments and to prove propositions mathematically using elementary logic. Students are also introduced to fundamental concepts of graph theory and elementary graph algorithms.

After establishing the concept of algorithms, the first part covers basic elements of discrete mathematics, leading to

Boolean algebra, propositional logic, and predicate logic. Students learn how to use fundamental proof techniques to prove (or disprove) simple propositions. The second part of the module introduces students to basic concepts of algebraic structures like groups, rings, and fields and different structure preserving maps (homomorphisms). Students study how these abstract concepts relate to problems in computer science. The last part of the module covers the basic elements of graph theory and the different representation of graphs. Elementary graph algorithms are introduced that have a wide range of applicability in computer science.

Recommended Knowledge

It is recommended that students revise mathematical concepts from their high school education.

Usability and Relationship to other Modules

This module introduces key mathematical concepts and teaches students to work with mathematical abstractions that are relevant for computer science. The acquired skills are relevant for subsequent courses covering theoretical or abstract

aspects of computer science.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain basic concepts and properties of algorithms.
2	Understand	Understand the concept of termination and complexity metrics.
3	Illustrate	Illustrate basic concepts of discrete math (sets, relations, functions).
4	Use	Use basic proof techniques and apply them to prove properties of algorithms.
5	Summarize	Summarize basic principles of Boolean algebra and propositional logic.
6	Use	Use predicate logic and outline concepts such as validity and satisfiability.
7	Distinguish	Distinguish abstract algebraic structures such as groups, rings and fields.
8	Classify	Classify different structure preserving maps (homomorphisms).
9	Understand	Understand calculations in finite fields and their applicability to computer science.
10	Explain	Explain elementary concepts of graph theory and use different graph representations.
11	Outline	Outline basic graph algorithms (e.g., traversal, search, spanning trees).

Indicative Literature

- Eric Lehmann, F. Thomson Leighton, Albert R. Meyer: Mathematics for Computer Science, online 2018.
- Winfried K. Grassmann, Jean-Paul Tremblay: Logic and Discrete Mathematics: A Computer Science Perspective, Pearson, 1996.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Mathematical Foundations of Computer Science	Written Examination	120 minutes	100	45%	All
Mathematical Foundations of Computer Science Tutorial					

Module Achievements: 50% of ten weekly assignments correctly solved. Two additional assignments are offered during the semester and another assignment is offered in January to makeup missing points.

7.4 Digital Systems and Computer Architecture

Module Name	Digital Systems and Computer Architecture
Module Code	2025-CH-234
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-ECE-BSc 2 - 2025-RIS-BSc 2 - 2025-CS-BSc 2 Mandatory Elective status for: - 2025-SDT-BSc 2
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Lecture	35
Tutorial	17.5
Independent Study	115
Exam Preparation	20
Workload Hours	187.5 hours

Module Components	Number	Type	CP
Digital Systems and Computer Architecture	CH-234-A	Lecture	5
Digital Systems and Computer Architecture Tutorial	CH-234-B	Tutorial	2.5

Module Description

The module introduces the essential hardware components of a digital computer system. Students will learn how useful digital circuits to add numbers or to store data can be constructed out of basic logic gates. Using these building blocks, the module will introduce how a simple processor can be constructed and how it interacts with memory systems and other components of a computer system. Students will practice the basics of assembler programming to understand program execution at the hardware level.

Usability and Relationship to other Modules

This module introduces students to the digital hardware components of a computer system. Students attain an understanding of program execution at the hardware level. Other modules requiring an understanding of program execution at the hardware level may require this module as a prerequisite.

Intended Learning Outcomes

No	Competence	ILO

1	Understand	Understand the architecture of a digital computer.
2	Explain	Explain the representation of numbers (integers and floats).
3	Summarize	Summarize basic laws of Boolean algebra.
4	Describe	Describe basic logic gates and which Boolean functions they implement.
5	Construct	Construct and analyze basic combinational digital circuits (e.g., adder, comparator, multiplexer).
6	Design	Design and analyze basic sequential digital circuits (e.g., latches, flip-flops).
7	Outline	Outline the basic structure of the von Neumann computer architecture.
8	Explain	Explain the execution of machine instructions on a von Neumann computer.
9	Develop	Develop simple programs in an assembler language such as the RISC-V.
10	Demonstrate	Demonstrate how function calls are executed and the role of the stack.
11	Understand	Understand microarchitectural concepts and the importance of the memory hierarchy.
12	Explain	Explain the purpose and principles of operation of the components of a computer system.

Indicative Literature

- John L Hennessy, David A. Patterson: Computer Architecture: A Quantitative Approach, 6th edition, Morgan Kaufmann, 2017.
- Sarah Harris, David Harris: Digital Design and Computer Architecture: RISC-V Edition, Morgan Kaufmann, 2021.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Digital Systems and Computer Architecture	Written Examination	120 minutes	100	45%	1-12
Digital Systems and Computer Architecture Tutorial					1-12

Module Achievements: 50% of ten weekly assignments correctly solved. Two additional assignments are offered during the semester and another assignment is offered in August to makeup missing points.

7.5 Development in JVM Languages

Module Name	**Development in JVM Languages**
Module Code	2025-SDT-103
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-SDT-BSc 2 Mandatory Elective status for: -2025-CS-BSc 2
Duration	1 Semester
Program Affiliation	2025-SDT-BSc (Software, Data and Technology)
Module Coordinator(s)	Prof. Dr. Alexander Omelchenko

Forms of Learning and Teaching	
Class Attendance	35
Tutorial	35
Independent Study	97.5
Exam Preparation	20
Workload Hours	187.5 hours

Module Components	**Number**	**Type**	**CP**
Development in JVM Languages	SDT-103-A	Lecture	2.5
Development in JVM Languages	SDT-103-B	Tutorial	5

Module Description

In this module students will learn about the Kotlin programming language, a modern, powerful and expressive language that is used for various purposes from android development, web development to data science. Students will learn how to apply Kotlin to solve practical problems in software development and will learn about data types, variables and control flow, functions, object-oriented programming, exception handling, collections and generics, lambdas, and higher-order functions. They will also learn about the unique features of Kotlin such as null safety, extension functions and type inference.

Educational Aims:

- To provide students with a solid foundation in the Kotlin programming language

- To teach students how to apply Kotlin to solve practical problems in software development

- To enable students to write efficient, readable and maintainable code using Kotlin

- To familiarize students with the unique features of Kotlin such as null safety, extension functions, and type inference

- To prepare students for using Kotlin in Android Development.

- To give students a deeper understanding of the fundamental concepts of computer science, such as algorithms and data structures and how they can be applied to software development.

Recommended Knowledge

Students should refresh their knowledge of the C++ and Python programming language and be able to solve simple programming problems in C++ and Python. Students are expected to have a working programming environment.

Usability and Relationship to other Modules

Familiarity with Kotlin programming language is essential for students who wish to specialize in android development, web development or data science. This module will provide a solid foundation in Kotlin programming, including its unique features such as null safety, extension functions, and type inference. Additionally, this module will introduce advanced concepts of programming that are needed in advanced programming-oriented modules in the 2nd and 3rd years of the SDT program.

Intended Learning Outcomes

No	Competence	ILO
1	Write	Write, understand and debug Kotlin code effectively
2	Use	Use the unique features of Kotlin to write readable, maintainable and expressive code.
3	Use	Use Kotlin to solve practical problems in software development.
4	Design	Design and implement object-oriented programs in Kotlin.
5	Use	Use Kotlin collections and Generics in their programs.
6	Use	Use Lambdas and Higher-Order functions in Kotlin.
7	Use	Use Kotlin for android development.
8	Write	Write efficient and optimized code using Kotlin.
9	Use	Use Kotlin for web development.
10	Use	Use Kotlin for data science.

Indicative Literature

- Venkat Subramaniam: Programming Kotlin, Pragmatic Bookshelf, 2017.
- Hadi Hariri: Kotlin in Action, Manning Publications, 2017.
- Dmitry Jemerov and Svetlana Isakova: Kotlin in Practice, JetBrains, 2016.
- Antonio Leiva: Kotlin for Android Developers, Leanpub, 2015.
- Marcin Moskala: Kotlin Programming, Packt Publishing, 2018.

Entry Requirements

Prerequisites	Programming in C and C++
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Development in JVM Languages	Written Examination	60 Minutes	33	45%	All theoretical ILOs
Development in JVM Languages - Tutorial	Program Code		67	45%	All practical ILOs

Module Achievements: None

7.6 Databases

Module Name	Databases
Module Code	2025-CO-560
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-CS-BSc 3 - 2025-Minor-CS-BSc 3 Mandatory Elective status for: - 2025-RIS-BSc 3
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Peter Baumann

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	35
Project	97.5
Workload Hours	187.5 hours

Module Components	Number	Type	CP
Databases	CO-560-A	Lecture	5
Databases - Project	CO-560-B	Project	2.5

Module Description

This module offers an introduction to databases, with emphasis on practically applicable knowledge and skills. The course starts with conceptual database design using the Entity Relationship (ER) model, followed by the relational model and SQL for querying relations. On that occasion, structures for storing relations on disk are inspected. After that, tuning opportunities are discussed, including Normal Forms, indexing, transaction management, and views, and finally – based on a brief look at Relational Algebra query processing and optimization in the server. As today databases often are used for Web services an excursion is made to inspect the server side of Web request processing in the context of databases. This in turn prompts security considerations in databases. Concluding the relational part, the travel leads into NoSQL and NewSQL world. This widens the perspective towards data models beyond tables and redefined transaction concepts. Towards the semester end, OLAP datacubes are introduced as a practically important database application with special needs, concepts, and technology.

A hands-on group project complements the theoretical aspects: on a self-chosen topic, teams of 3 – 4 students implement the core of a web-accessible information system using python (or PHP), MariaDB, and Linux, in a guided sequence of homework assignments.

Recommended Knowledge

Working knowledge of basic data structures, such as trees, is required as well as familiarity with an object-oriented programming language such as C++. Basic knowledge of algebra is useful. For the project work, students benefit from having basic hands-on skills using Linux and, ideally, basic knowledge of a scripting language such as Python (the official Python documentation is available online).

Usability and Relationship to other Modules

This module introduces components that are widely used by modern applications and information systems. Students can apply their knowledge in the software engineering module. This module serves as a default advanced level minor module.

Intended Learning Outcomes

No	Competence	ILO
1	Read	Read and write ER diagrams.
2	Design	Design and normalize schemas for relational databases.
3	Write	Write SQL queries and understand their evaluation by a database server.
4	Know	Know common tuning methods in relational databases.
5	Explain	Explain the concept of transactions and how to use transactions in application design.
6	Explain	Explain core security and privacy issues in the context of databases.
7	Describe	Describe the differences of selected NoSQL data models and make a requirement-driven choice.
8	Describe	Describe the concept of datacubes and how databases can support it.
9	Develop	Develop database-backed Web-enabled information systems, considering security aspects.

Indicative Literature

- Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer D. Widom: Database Systems: The Complete Book. 2nd edition, Pearson, 2008.
- Elvis C. Foster, Shripad V. Godbole: Database Systems. O'Reilly, 2014.
- Miguel Grinberg: Flask Web Development: Developing Web Applications with Python. O'Reilly, 2018.
- Jon Duckett: PHP & MySQL: Server-side Web Development. Wiley, 2022.

Entry Requirements

Prerequisites	Algorithms and Data Structures
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs

Databases	Written Examination	120 minutes	67	45%	1-8
Databases - Project	Project Assessment		33	45%	9

Module Achievements: None

7.7 Software Engineering

Module Name	Software Engineering
Module Code	2025-CO-561
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-CS-BSc 4 - 2025-Minor-CS-BSc 4 Mandatory Elective status for: - 2025-RIS-BSc 4
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Peter Baumann

Forms of Learning and Teaching	
Class Attendance	35
Independent Study	10
Development Work	132.5
Exam Preparation	10
Workload Hours	187.5 hours

Module Components	Number	Type	CP
Software Engineering	CO-561-A	Lecture	2.5
Software Engineering Project	CO-561-B	Project	5

Module Description

This module is an introduction to software engineering and object-oriented software design. The lecture focuses on software quality and the methods to achieve and maintain it in environments of "multi-person construction of multi-version software." Based on their pre-existing knowledge of an object-oriented programming language, students are familiarized with software architectures, design patterns and frameworks, software components and middleware, Unified Modeling Language (UML)-based modelling, and validation by testing. Furthermore, the course addresses the more organizational topics of project management and version control.

The lectures are accompanied by a software project in which students have to develop a software solution to a given problem. The problem is described from the viewpoint of a customer and students working in teams have to execute a whole software project lifecycle. The teams have to create a suitable software architecture and software design, implement the components, and integrate the components. The teams have to ensure that basic quality requirements for the solution and the components are defined and satisfied. The students produce various artifacts such as design documents, source code, test cases and user documentation. All artifacts need to be maintained in a version control system and the commits should allow the instructor and other team members to track in a meaningful way the changes and who has been contributing them.

Recommended Knowledge

Students are expected to be able to develop software using an object-oriented programming language such as C++, and they should have access to a Linux system and associated software development tools.

Intended Learning Outcomes

No	Competence	ILO
1	Understand	Understand and apply object-oriented design patterns
2	Read	Read and write UML diagrams
3	Contrast	Contrast the benefits and drawbacks of different software development models
4	Design	Design and plan a larger software project involving a team development effort
5	Translate	Translate requirements formulated by a customer into computer science terminology
6	Evaluate	Evaluate the applicability of different software engineering models for a given software development project
7	Assess	Assess the quality of a software design and its implementation
8	Apply	Apply tools that assist in the various stages of a software development process
9	Work	Work effectively in a team toward the goals of the team

Indicative Literature

- Ian Sommerville: Software Engineering, Pearson, 2010.
- Roger Pressman: Software Engineering – a Practitioner's Approach, McGraw-Hill, 2014.

Entry Requirements

Prerequisites	Databases
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Software Engineering	Written Examination	60 minutes	33	45%	1-3
Software Engineering Project	Project Assessment		67	45%	4-9

Module Achievements: None

7.8 Operating Systems

Module Name	Operating Systems
Module Code	2025-CO-562
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-CS-BSc 3 - 2025-SDT-BSc 3 Mandatory Elective status for: None
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	52.5
Exam Preparation	20
Independent Study	115
Workload Hours	187.5 hours

Module Components	Number	Type	CP
Operating Systems	CO-562-A	Lecture	7.5

Module Description

This module introduces concepts and principles used by operating systems to provide programming abstractions that enable an efficient and robust execution of application programs. Students will gain an understanding of how an operating system kernel manages hardware components and how it provides abstractions such as processes, threads, virtual memory, file systems, and inter-process communication facilities. Students learn the principles of event-driven and concurrent programming and the mechanisms that are necessary to solve synchronization and coordination problems, thereby avoiding race conditions, deadlocks, and resource starvation. The Linux kernel and runtime system will be used throughout the course to illustrate how key ideas and concepts have been implemented and how application programs can use them.

Recommended Knowledge

Students are expected to understand data representation and program execution at the machine instruction level as covered in the module Digital Systems and Computer Architecture.

Students are expected to have a working Linux installation, which allows them to compile and run sample programs provided by the instructor and to implement their own solutions for homework assignments.

Usability and Relationship to other Modules

This module enables students to write programs that make efficient use of the services provided by the operating system kernel. This is particularly important for advanced modules on computer networks, robotics, and embedded systems.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain the differences between processes, threads, application programs, libraries, and operating system kernels.
2	Describe	Describe well-known mutual exclusion and coordination problems.
3	Use	Use semaphores to achieve mutual exclusion and solve coordination problems.
4	Use	Use mutual exclusion locks and condition variables to solve synchronization and coordination problems.
5	Illustrate	Illustrate how deadlocks can be avoided, detected, and resolved.
6	Summarize	Summarize the different mechanisms to realize virtual memory and their trade-offs.
7	Solve	Solve basic inter-process communication problems using signals and pipes.
8	Use	Use socket inter-process communication primitives.
9	Multiplex	Multiplex I/O activities using suitable system calls and libraries.
10	Describe	Describe file system programming interfaces and the design of file systems at the operating system kernel level.
11	Explain	Explain how memory mapping can improve I/O performance.
12	Restate	Restate the functionality of a linker and the difference between static linking and dynamic linking.
13	Outline	Outline how different device types are supported by Unix-like kernels.
14	Discuss	Discuss virtualization mechanisms such as containers or virtual machines.

Indicative Literature

- Abraham Silberschatz, Peter B. Galvin, Greg Gagne: Applied Operating System Concepts, John Wiley, 2000.
- Andrew S. Tanenbaum, Herbert Bos: Modern Operating Systems, Prentice Hall, 4th edition, Pearson, 2015.
- William Stallings: Operating Systems: Internals and Design Principles, 8th edition, Pearson, 2014.
- Robert Love: Linux Kernel Development, 3rd edition, Addison Wesley, 2010.
- Robert Love: Linux System Programming: Talking Directly to the Kernel and C Library, 2nd edition, O'Reilly, 2013.

Entry Requirements

Prerequisites	-Core Algorithms and Data Structures **OR** Algorithms and Data Structures -Digital Systems and Computer Architecture

Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Operating Systems	Written Examination	120 minutes	100	45%	1-14

Module Achievements: 50% of the assignments correctly solved. This module includes hands-on assignments so that students can develop their system programming skills. The module achievement ensures that a sufficient level of practical system programming skills has been obtained.

7.9 Machine Learning

Module Name	Machine Learning
Module Code	2025-CO-541
Module ECTS	5
Study Semester	Mandatory status for: - 2025-MMDA-BSc 4 - 2025-RIS-BSc 4 - 2025-SDT-BSc 4 -2025-Minor-Software Development 4 Mandatory Elective status for: - 2025-CS-BSc 4 - 2025-PHDS-BSc 4 - 2025-IEM-BSc 6
Duration	1 Semester
Program Affiliation	2025-RIS-BSc (Robotics and Intelligent Systems)
Module Coordinator(s)	Prof. Dr. Francesco Maurelli

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	Number	Type	CP
Machine Learning	CO-541-A	Lecture	5

Module Description

Machine learning (ML) concerns algorithms that are fed with (large quantities of) real-world data, and which return a compressed "model" of the data. An example is the "world model" of a robot; the input data are sensor data streams, from which the robot learns a model of its environment, which is needed, for instance, for navigation. Another example is a spoken language model; the input data are speech recordings, from which ML methods build a model of spoken English; this is useful, for instance, in automated speech recognition systems. There exist many formalisms in which such models can be cast, and an equally large diversity of learning algorithms. However, there is a relatively small number of fundamental challenges that are common to all of these formalisms and algorithms. The lectures introduce such fundamental concepts and illustrate them with a choice of elementary model formalisms (linear classifiers and regressors, radial basis function networks, clustering, online adaptive filters, neural networks, or hidden Markov models). Furthermore, the lectures also (re-)introduce required mathematical material from probability theory and linear algebra.

Recommended Knowledge

Usability and Relationship to other Modules

- This module gives a thorough introduction to the basics of machine learning. It complements the Artificial Intelligence module.

Intended Learning Outcomes

No	Competence	ILO
1	Understand	Understand the notion of probability spaces and random variables
2	Understand	Understand basic linear modeling and estimation techniques
3	Understand	Understand the fundamental nature of the "curse of dimensionality"
4	Understand	Understand the fundamental nature of the bias-variance problem and standard coping strategies
5	Use	Use elementary classification learning methods (linear discrimination, radial basis function networks, multilayer perceptrons)
6	Implement	Implement an end-to-end learning suite, including feature extraction and objective function optimization with regularization based on cross-validation

Indicative Literature

- T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd edition, Springer, 2008.
- S. Shalev-Shwartz, Shai Ben-David: Understanding Machine Learning, Cambridge University Press, 2014.
- C. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
- T.M. Mitchell, Machine Learning, Mc Graw Hill, India, 2017.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Machine Learning	Written Examination	120 minutes	100	45%	1-6

Module Achievements: None

7.10 Functional Programming

Module Name	Functional Programming
Module Code	2025-SDT-202
Module ECTS	5
Study Semester	Mandatory status for: - 2025-Minor-Software Development 3 Mandatory Elective status for: - 2025-CS-BSc 3 - 2025-SDT-BSc 3
Duration	1 Semester
Program Affiliation	2025-SDT-BSc (Software, Data and Technology)
Module Coordinator(s)	Prof. Dr. Alexander Omelchenko

Forms of Learning and Teaching	
Lecture	17.5
Tutorial	17.5
Independent Study	70
Exam Preparation	20
Workload Hours	125 hours

Module Components	Number	Type	CP
Functional Programming	SDT-202-A	Lecture	2.5
Functional Programming Tutorial	SDT-202-B	Tutorial	2.5

Module Description

The goal of this discipline is to provide students with a solid foundation in functional programming principles and techniques, focusing on the theoretical knowledge and practical skills required to effectively work with functional languages. The module explores the core concepts, language structures, syntax, and semantic constructs of functional programming languages, emphasizing their applicability in modern software development.

Content:

- Fundamentals of functional programming: lambda calculus and combinatory logic.

- Haskell programming language: syntax, semantics, standard library.

- Manage effects using applicative functors and monads.

- Type systems of functional languages.

Recommended Knowledge

It is recommended that students install a Linux system such as Ubuntu on their notebooks and that they become familiar with basic tools such as editors (vim or emacs) and the basics of a shell. The Glasgow Haskell Compiler (GHC) will be used for implementing Haskell programs.

Usability and Relationship to other Modules

Familiarity with functional programming concepts and principles is increasingly important in fields such as data science, artificial intelligence, and software development. This module provides a solid foundation in functional programming techniques and languages, which are essential for advanced modules in computer science and data science. Additionally, this module introduces advanced concepts of functional programming that are needed in advanced programming-oriented modules in the 2nd and 3rd years of the SDT program.

Intended Learning Outcomes

No	Competence	ILO
1	Collaborate	Collaborate effectively within a team in the IT field, utilizing project management tools, communication skills, and software for team project activities to jointly develop projects.
2	Compare	Compare and contrast the advantages and disadvantages of the functional programming paradigm, and apply functional programming techniques to solve applied problems using languages such as Haskell
3	Understand	Understand and utilize the basic type systems of functional languages and their extensions with polymorphic and recursive types to create efficient, well-structured code in a functional programming context
4	Choose	Choose between lazy and eager evaluation strategies based on the specific requirements of a problem or application, and implement software solutions using a functional programming paradigm.
5	Explain	Explain the computational model underlying functional programming and implement algorithms in functional languages using key concepts such as immutable data structures, recursion, and pattern matching
6	Employ	Employ generic annotations and type classes to describe interfaces and ensure static control, promoting code reusability and maintainability in functional programming projects

Indicative Literature

- Miran Lipovača. Learn You a Haskell for Great Good.
- O'Sullivan, Bryan, John Goerzen, and Don Stewart. Real World Haskell. O'Reilly Media, Inc., 2008.
- Hughes, John. "Why functional programming matters." The computer journal 32.2 (1989): 98-107.

Entry Requirements

Prerequisites	Core Algorithms and Data Structures **OR**

	Algorithms and Data Structures
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Functional Programming	Written Examination	60 Minutes	50	45%	All theoretical ILOs of the module
Functional Programming Tutorial	Program Code		50	45%	All practical ILOs of the module

Module Achievements: None

7.11 Automata, Computability, and Complexity

Module Name	Automata, Computability, and Complexity
Module Code	2025-CO-563
Module ECTS	7.5
Study Semester	Mandatory status for: - 2025-CS-BSc 4 Mandatory Elective status for: None
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jakob Suchan

Forms of Learning and Teaching	
Class Attendance	52.5
Exam Preparation	20
Independent Study	115
Workload Hours	187.5 hours

Module Components	Number	Type	CP
Automata, Computability, and Complexity	CO-563-A	Lecture	7.5

Module Description

This module introduces the mathematical theory of computation. Several types of abstract computational machines (called automata) are introduced together with the associated theory of formal languages. A formal language is a set of words over a defined alphabet that are well-formed according to a specific set of rules, called the grammar of the language. After studying the relationship between automata models and classes of formal languages, this course addresses the fundamental question "What problems can a computer possibly solve?'' by characterizing those solvable problems, equivalently, through Turing machines, random access machines, recursive functions and lambda calculus. A full answer to the related question, "How many computational resources are needed for solving a given problem?'' is not known today. However, the basic outlines of today's theory of computational complexity will be presented up to the most famous open problem in computer science, namely the "P = NP'' question: if a computer could guess the right answer to a computational problem (and only needs to check its correctness), would that computer be faster than another one that cannot guess the right solution? This may seem to be a ridiculously obvious case of a clear YES answer, but in fact it is considered by many to be the deepest open question in contemporary mathematics (and computer science, of course).

This module provides the core education in theoretical computer science. The material covered in this module gives students access to any field in computer science, which is based on discrete-mathematical formal foundations, such as the theory of automata and formal languages or compiler design.

Usability and Relationship to other Modules

This module provides the core education in theoretical computer science

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain discrete automata models (finite state machines, pushdown automata, Turing machines).
2	Describe	Describe the Chomsky hierarchy of formal languages and classify formal languages.
3	Characterize	Characterize classes of formal languages by automata models and grammars.
4	Define	Define formal models of computation such as Turing machines.
5	Explain	Explain the equivalences of formal models of computation.
6	Illustrate	Illustrate the nature and impact of the Church-Turing hypothesis.
7	Construct	Construct diagonalization arguments.
8	Give	Give examples of functions that are not computable.
9	Contrast	Contrast central complexity classes (L, P, NP, EXP...).
10	Apply	Apply reduction techniques both for decidability and complexity.
11	Create	Create a reduction-based check of whether a problem is NP-complete.

Indicative Literature

- Michael Sipser: Introduction to the Theory of Computation 2nd edition PWS Publishing Company 1997 (Primary Literature)
- John Hopcroft Rajeev Motwani Jeffrey Ullman: Introduction to Automata Theory Languages and Computation 3rd edition Pearson 2006

Entry Requirements

Prerequisites	**Mathematical Foundations of Computer Science**
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Automata, Computability, and Complexity	Written Examination	120 minutes	100	45%	1-11

Module Achievements: None

7.12 Legal and Ethical Aspects of Computer Science

Module Name	**Legal and Ethical Aspects of Computer Science**
Module Code	2025-CO-565
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 3
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	175
Independent Study	35
Poster Preparation	10
Workload Hours	220 hours

Module Components	**Number**	**Type**	**CP**
Legal and Ethical Aspects of Computer Science	CO-565-A	Lecture	2.5

Module Description

Information technology has a profound impact on society. This module introduces the legal and ethical frameworks that are relevant for computer scientists taking up qualified employment or joining advanced study programs leading to a career in education and research. The module provides an overview of intellectual property rights and their regulations, data protection regulations, and ethical frameworks defined by professional organizations. Students are confronted with a collection of case studies to develop sensitivity to legal and ethical dilemmas with which people are sometimes faced during the construction or operation of advanced information processing systems.

Intended Learning Outcomes

No	**Competence**	**ILO**
1	Recall	Recall principles of data protection regulations such as the European General Data Protection Regulation (GDPR).
2	Identify	Identify components of an IT system managing sensitive data that needs protection.
3	Summarize	Summarize regulations concerning intellectual property rights.
4	Analyze	Analyze the applicability of different closed-source and open-source software licensing models.

5	Describe	Describe computer science ethics and ethical frameworks defined by professional organizations.
6	Illustrate	Illustrate ethical dilemma resulting from the use of information processing systems.
7	Discuss	Discuss the interplay of legal frameworks and ethical principles and the design of information processing systems.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Legal and Ethical Aspects of Computer Science	Poster Presentation	10 minutes		45%	

Module Achievements: None

7.13 Academic Skills in Computer Science

Module Name	Academic Skills in Computer Science
Module Code	2025-CO-567
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 4
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Dr. Kinga Lipskoch

Forms of Learning and Teaching	
Class Attendance	7.5
Independent Study	25
Presentation	20
Workload Hours	52.5 hours

Module Components	Number	Type	CP
Academic Skills in Computer Science	CO-567-A	Seminar	2.5

Module Description

This module introduces students to basic skills in reading, understanding, and evaluating scientific articles, and in presenting scientific results in presentations and publications. During the seminar, students will study some classic computer science papers with a special focus on how the papers are organized, written and how they present scientific results. Students will develop and discuss guidelines for effective writing and they will learn about techniques and tools that can be used to effectively search for literature relevant to a certain topic. Finally, students will be introduced to peer review processes.

As a project, students will emulate the workflow of a scientific conference to demonstrate the academic skills they have learned.

Intended Learning Outcomes

No	Competence	ILO
1	Effectively	Effectively find research literature for a given topic.
2	Critically	Critically read and assess research papers.
3	Present	Present a research result in the structure of a scientific paper.
4	Describe	Describe how scientific peer review processes work.
5	Orally	Orally communicate research results effectively to a scientific community.

6	Describe	Describe common pitfalls in the presentation of data, algorithms, or math.
7	Discuss	Discuss ethical issues and guidelines related to scientific publications.

Indicative Literature

- Peter Zobel: Writing for Computer Science, 3rd edition, Springer, 2014.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Academic Skills in Computer Science	Project Assessment		100	45%	1-7

Module Achievements: None

7.14 Computer Networks

Module Name	Computer Networks
Module Code	2025-CO-564
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 5 - 2025-SDT-BSc 5
Duration	1 Semester
Program Affiliation	2025-SDT-BSc (Software, Data and Technology)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	35
Independent Study	70
Exam Preparation	20
Workload Hours	125 hours

Module Components	Number	Type	CP
Computer Networks	CO-564-A	Lecture	5

Module Description

Computer networks such as the Internet play a critical role in today's connected world. This module discusses the technology of Internet services in depth to enable students to understand the core issues involved in the design of modern computer networks. Fundamental algorithms and principles are explained in the context of existing protocols as they are used in today's Internet. Students taking this module should finally understand the technical complexity behind everyday online services such as Google or YouTube.

Students taking this module will understand how computer networks work, and they will be able to assess communication networks, including aspects such as performance but also robustness and security. Students will learn that the design of communication networks is not only influenced by technical constraints but also by the necessity to define common standards, which often require engineering decisions that reflect non-technical requirements.

Recommended Knowledge

Students are expected to be familiar with the C programming language and to learn basics of higher-level scripting languages such as Python (the official Python documentation is available on https://docs.python.org/)

Usability and Relationship to other Modules

The module should be taken together with the module Operating Systems, because a significant portion of the communication technology is implemented at the operating system level. An understanding of operating system concepts and abstractions will help students to understand how computer network technology is commonly implemented and made available to applications. The specialization module Distributed Algorithms discusses algorithms for solving problems commonly found in distributed systems that use computer networks to exchange information. The module Secure and Dependable Systems introduces cryptographic mechanisms that can be used to secure communication over computer networks.

Intended Learning Outcomes

No	Competence	ILO
1	Recall	Recall layering principles and the OSI reference model.
2	Articulate	Articulate the organization of the Internet and the organization involved in providing Internet services.
3	Describe	Describe media access control, flow control, and congestion control mechanisms
4	Explain	Explain how local area networks differ from global networks.
5	Illustrate	Illustrate how frames are forwarded in local area networks.
6	Contrast	Contrast addressing mechanisms and translations between addresses used at different layers.
7	Demonstrate	Demonstrate how the Internet network layer forwards packets.
8	Present	Present how routing algorithms and protocols are used to determine and select routes.
9	Describe	Describe how the Internet transport layer provides different end-to-end services.
10	Demonstrate	Demonstrate how names are resolved to addresses and vice versa.
11	Summarize	Summarize how application layer protocols send and access electronic mail or access resources on the world-wide web.
12	Design	Design and implement simple application layer protocols.
13	Recognize	Recognize to which extent computer networks are fragile and evaluate strategies to cope with the fragility.
14	Analyze	Analyze traffic traces produced by a given computer network.

Indicative Literature

- James F. Kurose, Keith W. Ross: Computer Networking: A Top-Down Approach Featuring the Internet, 3rd Edition, Addison-Wesley, 2004.
- Andrew S. Tanenbaum: Computer Networks, 4th Edition, Prentice Hall, 2002.

Entry Requirements

Prerequisites	Core Algorithms and Data Structures **OR** Algorithms and Data Structures
Co-requisites	Operating Systems
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Computer Networks	Written Examination	120 minutes	100	45%	1-14

Module Achievements: None

7.15 Secure and Dependable Systems

Module Name	**Secure and Dependable Systems**
Module Code	2025-CO-566
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 5 - Mandatory for Specialization Cybersecurity
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	**Number**	**Type**	**CP**
Secure and Dependable Systems	CO-566-A	Lecture	5

Module Description

This module introduces students to the fundamentals of computer security and techniques used to build and analyze dependable systems. This is an important topic given that computer systems are increasingly embedded in everyday objects (such as light bulbs) and taking over important control functions (such as driving cars). Furthermore, computer systems control complex communication systems that form critical infrastructure of the modern globalized world. Proper protection of information requires an applied understanding of cryptography and how cryptographic primitives are used to secure data and information exchanges. The aim of this module is to make students aware of what types of security vulnerabilities may arise in computing systems and how to prevent, identify, and fix them.

Recommended Knowledge

Usability and Relationship to other Modules

Intended Learning Outcomes

No	**Competence**	**ILO**
1	Recall	Recall dependability terminology and concepts.

2	Explain	Explain control flow attacks and injection attacks and defense mechanisms.
3	Describe	Describe network data plane and control plane attacks and defense mechanisms.
4	Understand	Understand symmetric and asymmetric cryptographic algorithms.
5	Explain	Explain how digital signatures and public key infrastructures work.
6	Analyze	Analyze key exchange protocols for weaknesses.
7	Describe	Describe secure network protocols (e.g., PGP, TLS, and SSH).
8	Recall	Recall anonymity terminology and concepts.
9	Discuss	Discuss information hiding mechanisms (e.g., steganography, and watermarking).
10	Illustrate	Illustrate anonymization techniques (mixes, onion routing).

<u>Indicative Literature</u>

- Bruce Schneier: Applied Cryptography 20th Anniversary Edition Wiley 2015
- WmA Conklin Gregory White: Principles of Computer Security 5th Edition McGraw-Hill 2018
- Simon Singh: The Code Book: Science of Secrecy from Ancient Egypt to Quantum Cryptography Anchor Books 2000

<u>Entry Requirements</u>

Prerequisites	Operating Systems
Co-requisites	None
Additional Remarks	None

<u>Assessment and Completion</u>

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Secure and Dependable Systems	Written Examination	120 minutes	100	45%	1-10

Module Achievements: None

7.16 Security Monitoring and Incident Response

Module Name	Security Monitoring and Incident Response
Module Code	2025-CA-S-CS-807
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6 - Mandatory for Specialization Cybersecurity
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	35
Independent Study	70
Exam Preparation	20
Workload Hours	125 hours

Module Components	Number	Type	CP
Security Monitoring and Incident Response	CA-CS-807	Lecture	5

Module Description

Cyber threats are an ever-present risk in today's interconnected world, making security monitoring and incident response essential disciplines in cybersecurity. This module explores the technologies, strategies, and frameworks used to detect, analyze, and respond to security incidents in modern IT environments. Students will gain an in-depth understanding of security monitoring tools, threat intelligence, and forensic techniques while learning how real-world cyberattacks unfold.

By taking this module, students will develop the ability to assess security events, investigate breaches, and implement effective response strategies. They will learn how to use open-source tools for log analysis, intrusion detection, and incident handling, gaining practical experience through hands-on exercises. The course also emphasizes the role of compliance, legal considerations, and communication in incident response. Students will understand that effective security monitoring and response require not only technical expertise but also collaboration, strategic decision-making, and adherence to regulatory standards.

Students are expected to be familiar with operating systems security mechanisms (including object ownership model, access control lists, and privilege elevation), to understand networking protocols and layers, and to be familiar with regular expressions. Taking the course "Computer Networks" is recommended.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain the need for security monitoring.
2	Understand	Understand the threat landscape and attack vectors.
3	Identify	Identify and analyze security threats using frameworks like MITRE ATT&CK and the Cyber Kill Chain.
4	Explain	Explain the purpose of Security Information and Event Management (SIEM) tools to monitor, detect, and correlate security events.
5	Conduct	Conduct log analysis and anomaly detection across network, endpoint, and cloud environments.
6	Implement	Implement incident response procedures based on established frameworks like NIST 800-61 and ISO 27035.
7	Apply	Apply containment, eradication, and recovery strategies to mitigate security incidents.
8	Anticipate	Anticipate and respond to cyber threats using threat intelligence sources.
9	Apply	Apply digital forensics techniques to investigate incidents, including memory, disk, and network traffic analysis.
10	Communicate	Communicate incident findings effectively to technical teams, executives, and regulatory bodies.
11	Understand	Understand legal, ethical, and regulatory considerations in incident response, including GDPR and ISO 27001:2022

Indicative Literature

- Richard Bejtlich, The Practice of Network Security Monitoring: Understanding Incident Detection and Response, No Starch Press, 2013.
- Eric Thompson, Cybersecurity Incident Response: How to Contain, Eradicate, and Recover from Incidents, Apress, 2018.

Entry Requirements

Prerequisites	Operating Systems
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Security Monitoring and Incident Response	Written Examination	120 minutes	100	45%	

Module Achievements: None

7.17 Ethical Hacking and Offensive Security

Module Name	Ethical Hacking and Offensive Security
Module Code	2025-CA-S-CS-808
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6 - Mandatory for Specialization Cybersecurity
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	17.5
Independent Study	47.5
Laboratory	35
Report Preparation	25
Workload Hours	125 hours

Module Components	Number	Type	CP
Ethical Hacking and Offensive Security	CA-CS-808	Laboratory	5

Module Description

This module introduces students to the principles and practices of ethical hacking, focusing on how attackers exploit weaknesses in systems, networks, and applications. It provides a structured and hands-on approach to offensive security, teaching students how to simulate cyberattacks in a controlled, lawful, and ethical manner. Core topics include reconnaissance, vulnerability scanning, exploitation, privilege escalation, and post-exploitation techniques, as well as specialized areas such as web application security, wireless attacks, and social engineering. The course aligns with industry frameworks like MITRE ATT&CK and emphasizes the responsible use of offensive tools, legal boundaries, and professional reporting standards.

The educational aim is to equip students with the technical skills and mindset needed to think like an attacker to better defend systems. Students will develop proficiency in using common penetration testing tools and gain experience in conducting end-to-end simulated attacks. Emphasis is placed on the ethical implications of offensive security, the importance of defined scope and consent, and the value of red teaming in real-world security assessments. By the end of the module, students will be prepared to contribute to penetration testing engagements or perform them under supervision.

Before taking a course in Ethical Hacking and Offensive Security, students are generally expected to have a foundational understanding of computer networks, be familiar with both Linux and Windows operating systems, command-line interfaces, file systems, and user permissions, and have a working knowledge of basic cybersecurity concepts, such as common attack types, system vulnerabilities, and defensive mechanisms like firewalls and intrusion detection systems. A reasonable level of scripting or programming experience, particularly in languages like Python or PowerShell, is important for automating tasks or writing simple exploits.

Intended Learning Outcomes

No	Competence	ILO
1	Understand	Understand the legal, ethical, and professional responsibilities involved in ethical hacking and penetration testing.
2	Conduct	Conduct reconnaissance and information-gathering activities using open-source intelligence (OSINT) techniques.
3	Perform	Perform vulnerability scanning and enumeration of systems, networks, and applications.
4	Exploit	Exploit common security weaknesses in networks, operating systems, and web applications using industry-standard tools.
5	Apply	Apply privilege escalation techniques and maintain access in simulated attack environments.
6	Evaluate	Evaluate and exploit vulnerabilities identified in web applications, referencing the OWASP Top 10.
7	Simulate	Simulate wireless and social engineering attacks within ethical boundaries and defined scopes.
8	Automate	Automate and conduct various stages of offensive security testing using tools such as Metasploit, Burp Suite, and Nmap.
9	Apply	Apply the MITRE ATT&CK framework to structure and document offensive security tactics and techniques.
10	Communicate	Communicate security findings effectively through structured technical reports and executive summaries.
11	Demonstrate	Demonstrate the ability to plan, execute, and document an end-to-end penetration test within a controlled lab environment.
12	Reflect	Reflect on the role of offensive security within broader cybersecurity strategies and defense mechanisms.

Indicative Literature

- Stuttard, D. & Pinto, M. (2017). The Web Application Hacker’s Handbook (2nd ed.). Wiley.
- Weidman, G. (2014). Penetration Testing: A Hands-On Introduction to Hacking. No Starch Press.

Entry Requirements

Prerequisites	Secure and Dependable Systems
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Ethical Hacking and Offensive Security	Laboratory Report		100	45%	1-12

Module Achievements: None

7.18 Advanced Operating Systems

Module Name	Advanced Operating Systems
Module Code	2025-CA-S-CS-810
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 5
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	

Forms of Learning and Teaching	
Class Attendance	35
Independent Study	70
Exam Preparation	20
Workload Hours	125 hours

Module Components	Number	Type	CP
Advanced Operating Systems	CA-CS-810	Lecture	5

Module Description

This advanced course explores the fundamental principles and modern advances in operating systems beyond introductory material, focusing on the architectural choices, performance trade-offs, and system-level abstractions that are at the basis of robust and scalable systems. The course is centered on the analytical and experimental evaluation of algorithms and design strategies adopted in state-of-the-art systems, including multicore, distributed, and real-time systems.

The course topics include:

- recap of OS architectures, kernel modes;

- boot sequences and initialization of the kernel and user-space initialization;

- advanced process and thread models and context switching for multicore systems;

- real-time scheduling, multiprocessor scheduling, load balancing;

- wait-free/lock-free synchronization techniques and read-copy-update mechanisms;

- modern efficient inter-process communication facilities;

- memory management supporting translation lookaside buffers, huge pages, non-uniform memory access;

- mandatory and discretionary access control, memory safety, isolation, namespaces, and container security.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain the fundamental concepts and principles of modern operating system design, including multicore and multithreading architectures, resource management, and process synchronization.
2	Analyze	Analyze the trade-offs and design decisions involved in developing efficient and scalable operating system components, considering factors such as performance, reliability, and security.
3	Evaluate	Evaluate the performance and effectiveness of various scheduling algorithms, memory management techniques, and synchronization mechanisms in the context of real-world operating system scenarios.
4	Determine	Determine the appropriate strategies for managing hardware resources and devices, such as interrupt handling, device drivers, and system calls, to ensure optimal system performance and functionality.
5	Describe	Describe the challenges and solutions associated with developing and debugging complex operating system components, including techniques for troubleshooting and optimizing system behavior.
6	Ability	Ability to design, implement, and optimize advanced operating system components, such as process schedulers, memory managers, and synchronization primitives, using modern programming languages and tools.
7	Communicate	Communicate effectively, both orally and in writing, the key concepts, design decisions, and implementation details of advanced operating system components to technical and non-technical audiences.

Indicative Literature

- Silberschatz, Galvin, Gagne: Operating System Concepts, 8th edition, John Wiley & Sons, 2008.
- Robert Love: Linux Kernel Development, 3rd edition, Addison-Wesley Professional, 2010.
- Anderson, Dahlin: Operating Systems: Principles and Practice.
- Selected recent papers (SIGOPS, USENIX, EuroSys, ACM Queue).

Entry Requirements

Prerequisites	**Operating Systems**
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs

Advanced Operating Systems	Written Examination	120 minutes	100	45%	1-7

Module Achievements: None

7.19 Linux Kernel Internals

Module Name	Linux Kernel Internals
Module Code	2025-CA-S-CS-809
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	

Forms of Learning and Teaching	
Class Attendance	35
Independent Study	70
Exam Preparation	20
Workload Hours	125 hours

Module Components	Number	Type	CP
Linux Kernel Internals	CA-CS-809	Lecture	5

Module Description

This course provides an in-depth, technical, and hands-on investigation of the internal structure and evolution of the Linux kernel. Students are guided through the key kernel subsystems, design rationales, and interfaces. Students will learn how the kernel manages processes, memory, filesystems, devices, and networking. The module has the approach of presenting the overall architectural idea and then going deep into the source code to promote a concrete and practical understanding of kernel programming, debugging, and performance profiling.

In particular, the course will cover:

- overall Linux kernel architecture;

- Linux kernel source code and the kernel build system;

- Linux boot and initialization sequence;

- task management covering processes, threads, kernel threads, and their scheduling;

- interrupts and exceptions, bottom halves and tasklets;

- memory management beyond simple physical/virtual mappings, slab/slub allocators, NUMA support;

- Linux virtual file system layer, filesystems (e.g., ext4, Btrfs), journaling, pseudo filesystems (e.g., proc, udev, sysfs);

- Linux socket layer, protocol stacks, packet filtering and address translation (e.g., netfilter);

- synchronization in the kernel: spinlocks, semaphores, read-copy-update implementations;

- Linux debugging/profiling techniques such as printk, dmesg, kprobes, perf, ftrace

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain the modular or monolithic architectures of the Linux kernel, including the advantages and disadvantages of each approach and their impact on kernel design and performance
2	Analyze	Analyze the mechanisms used by the Linux kernel to manage hardware resources, such as device drivers, interrupt handling, and system calls, and evaluate their effectiveness in ensuring efficient and reliable system operation.
3	Evaluate	Evaluate the Linux kernel's support for multi-tasking, concurrency, and inter-process communication, and determine the appropriate synchronization primitives and techniques for developing reliable and efficient kernel-level code.
4	Describe	Describe the Linux kernel's memory management, file systems, and networking subsystems, and explain how they interact with each other and with user-space applications to provide essential system services.
5	Demonstrate	Demonstrate the ability to develop, debug, and optimize kernel modules and device drivers using the Linux kernel's programming interfaces and tools, such as the Linux Device Model and the kernel's debugging facilities

Indicative Literature

- Robert Love: Linux Kernel Development, 3rd edition, Addison-Wesley Professional, 2010.
- Daniel P. Bovet and Marco Cesati: Understanding the Linux Kernel, 3rd edition, O'Reilly Media, 2005.
- Greg Kroah-Hartman: Linux Kernel in a Nutshell, O'Reilly and Associates, 2007.
- Linux kernel source documentation, LWN.net articles, kernel mailing lists

Entry Requirements

Prerequisites	**Operating Systems** **Computer Networks**
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Linux Kernel Internals	Written Examination	120 minutes	100	45%	1-5

Module Achievements: None

7.20 Computer Graphics

Module Name	Computer Graphics
Module Code	2025-CA-S-CS-801
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-RIS-BSc 5 - 2025-CS-BSc 5
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	35
Independent Study	70
Exam Preparation	20
Workload Hours	125 hours

Module Components	Number	Type	CP
Computer Graphics	CA-CS-801	Lecture	5

Module Description

This module deals with the digital synthesis and manipulation of visual content. The creation process of computer graphics spans from the creation of a three-dimensional (3D) scene to displaying or storing it digitally. Prominent tasks in computer graphics are geometry processing, rendering, and animation. Geometry processing is concerned with object representations such as surfaces and their modeling. Rendering is concerned with transforming a model of the virtual world into a set of pixels by applying models of light propagation and sampling algorithms. Animation is concerned with descriptions of objects that move or deform over time. This is an introductory module covering the concepts and techniques of 3D (interactive) computer graphics. It covers mathematical foundations, basic algorithms and principles, and some advanced methods and concepts. An introduction to the implementation of simple programs using a mainstream computer graphics library completes this module.

Usability and Relationship to other Modules

Students with a strong interest in graphical user interfaces are encouraged to also select the Human–Computer Interaction specialization module, which discusses among other things how computer graphics can be used as a component of interactive graphical user interfaces.

Intended Learning Outcomes

No	Competence	ILO

1	Construct	Construct 3D geometry representations.
2	Apply	Apply 3D transformations.
3	Understand	Understand the algorithms and optimizations applied by graphics rendering systems.
4	Explain	Explain the stages of modern computer graphics programmable pipelines
5	Implement	Implement simple computer graphics applications using graphics frameworks such as OpenGL.
6	Illustrate	Illustrate the techniques used to create animations.

Indicative Literature

- John Hughes, Andries van Dam, Morgan McGuire, David F. Sklar, James D. Foley, Steven K. Feiner, Kurt Akeley, Computer Graphics - Principles and Practice, 3rd edition, Addison-Wesley, 2013.
- Peter Shirley, Steve Marschner, Fundamentals of Computer Graphics, 4th edition, Taylor and Francis Ltd, 2016.
- Matt Pharr, Wenzel Jakob, Greg Humphreys, Physically Based Rendering: From Theory to Implementation, 3rd edition, Morgan Kaufmann, 2016.

Entry Requirements

Prerequisites	Core Algorithms and Data Structures **OR** Algorithms and Data Structures
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Computer Graphics	Written Examination	120 minutes	100	45%	1-6

Module Achievements: None

7.21 Image Processing

Module Name	Image Processing
Module Code	2025-CA-S-CS-802
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Markus Wenzel

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	Number	Type	CP
Image Processing	CA-CS-802	Lecture	5

Module Description

The module provides a foundation of the theory and applications of digital image processing. The first part concentrates on morphological image processing, which is one of the most basic yet powerful tool sets in dealing with digital images, and it is the backbone of many of today's high-performance image analysis systems. The module starts by introducing concepts such as dilation, erosion, geodesic transformations, morphological filtering, and the watershed transform. It then develops into advanced strategies for image segmentation and texture analysis. The second part of the module will concentrate on understanding problems from real-world applications, such as in biomedical imaging, and provides an overview of the broader field of image processing. The course can be combined with other courses on machine learning and signal analysis. Homework assignments will cover C/C++ implementations of basic and combined image processing algorithms.

Recommended Knowledge

Students are required to have a good understanding of data structures and algorithms, e.g. as provided in “Algorithms and Data Structures” or “Core Algorithms and Data Structures”. Familiarity with Python programming, and preferably with Python or other image processing libraries, is a great advantage.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain the theory and concepts of image processing.

2	Illustrate	Illustrate concepts such as dilation, erosion, geodesic transformations, and morphological filtering.
3	Analyze	Analyze image segmentation and texture analysis algorithms.
4	Design	Design and implement their own image processing algorithms in C/C++.

Indicative Literature

- Milan Sonka Vaclav Hlavac Roger Boyle: Image Processing Analysis and Machine Vision 3rd edition Nelson Engineering 2007
- Pierre Soille Morphological Image Analysis: Principles and Applications 2nd edition Springer 2004

Entry Requirements

Prerequisites	Core Algorithms and Data Structures **OR** Algorithms and Data Structures
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Image Processing	Written Examination	120 minutes	100	45%	1-4

Module Achievements: None

7.22 Distributed Algorithms

Module Name	Distributed Algorithms
Module Code	2025-CA-S-CS-803
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6 - 2025-RIS-BSc 6 - 2025-SDT-BSc 6
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Dr. Kinga Lipskoch

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	Number	Type	CP
Distributed Algorithms	CA-CS-803	Lecture	5

Module Description

Distributed algorithms are the foundation of modern distributed computing systems. They are characterized by a lack of knowledge of a global state, a lack of knowledge of a global time, and inherent non-determinism in their execution. The course introduces basic distributed algorithms using an abstract formal model, which is centered on the notion of a transition system. The topics covered are logical clocks, distributed snapshots, mutual exclusion algorithms, wave algorithms, election algorithms, reliable broadcast algorithms, and distributed consensus algorithms. Process algebras are introduced as another formalism to describe distributed and concurrent systems.

The distributed algorithms introduced in this module form the foundation of computing systems that have to be scalable and fault-tolerant, e.g., large-scale distributed non-standard databases or distributed file systems. The course is recommended for students interested in the design of scalable distributed computing systems.

Recommended Knowledge

Students should refresh their knowledge of the C, C++ and Python programming language and be able to solve simple programming problems in C, C++ and Python. Students are expected to have a working programming environment.

Intended Learning Outcomes

No	Competence	ILO
1	Describe	Describe and analyze distributed algorithms using formal methods such as transition systems.
2	Explain	Explain different algorithms to solve election problems.
3	Illustrate	Illustrate the limitations of time to order events and how logical clocks and vector clocks overcome these limitations.
4	Apply	Apply distributed algorithms to produce consistent snapshots of distributed computations.
5	Describe	Describe the differences among wave algorithms for different topologies.
6	Analyze	Analyze and implement distributed consensus algorithms such as Paxos and Raft.
7	Use	Use a process algebra such as communicating sequential processes or -calculus to model distributed algorithms.

<u>Indicative Literature</u>

- Maarten van Steen, Andrew S. Tanenbaum: Distributed Systems, 3rd edition, Pearson Education, 2017.
- Nancy A. Lynch: Distributed Algorithms, Morgan Kaufmann, 1996.

<u>Entry Requirements</u>

Prerequisites	Core Algorithms and Data Structures **OR** Algorithms and Data Structures
Co-requisites	None
Additional Remarks	

<u>Assessment and Completion</u>

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Distributed Algorithms	Written Examination	120 minutes	100	45%	1-7

Module Achievements: None

7.23 Web Application Development

Module Name	Web Application Development
Module Code	2025-CA-S-CS-804
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6 - 2025-RIS-BSc 6
Duration	1 Semester
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Prof. Dr. Jürgen Schönwälder

Forms of Learning and Teaching	
Class Attendance	17.5
Independent Study	40
Project Work	50
Exam Preparation	17.5
Workload Hours	125 hours

Module Components	Number	Type	CP
Web Application Development	CA-CS-804-A	Lecture	2.5
Web Application Development - Project	CA-CS-804-B	Project	2.5

Module Description

A web application is a client-server computer program where the client provides the user interface and the client-side logic runs in a web browser or as an app running on a mobile device such as a smart phone or a tablet. A key characteristic is that more complex application logic and data storage is realized by a server offering a web application programming interface.

This module focuses on the client side of web application and introduces technologies that can be used to implement interactive user interfaces and client-side logic. It builds on the module databases and web services, which covers the data storage components and server-side logic of web applications.

This module consists of a lecture and an associated project. The lecture component introduces programming languages and frameworks that are widely used for implementing the client side of web applications such as Java, Kotlin, Swift, JavaScript and frameworks built on top of them. In the project component, students develop web applications and test them on existing and openly accessible web services.

Intended Learning Outcomes

No	Competence	ILO

1	Explain	Explain the document object model behind HTML and its relation to CSS.
2	Discuss	Discuss the principles and basic mechanisms of reactive website design.
3	Analyze	Analyze the interactions between web applications and web services.
4	Use	Use languages such as Java, Kotlin, or Swift to implement mobile web applications.
5	Use	Use web standards such as HTML, CSS, and JavaScript to implement web applications running in standard web browsers.

Indicative Literature

- Stoyan Stefanov: JavaScript Patterns, O'Reilly Media, 2010.
- Alexey Soshin: Hands-on Design Patterns with Kotlin, Packt Publishing, 2018.
- Alex Banks, Eve Porcello: Learning React: Functional Web Development. With React and Flux, O'Reilly, 2017.

Entry Requirements

Prerequisites	Databases
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Web Application Development	Written Examination	120 minutes	50	45%	1-3
Web Application Development - Project	Project Assessment		50	45%	4-5

Module Achievements: None

7.24 Computer Vision

Module Name	Computer Vision
Module Code	2025-CO-546
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 3 - 2025-RIS-BSc 3
Duration	1 Semester
Program Affiliation	2025-RIS-BSc (Robotics and Intelligent Systems)
Module Coordinator(s)	Prof. Dr. Francesco Maurelli

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	Number	Type	CP
Computer Vision	CO-546-A	Lecture and Laboratory	5

Module Description

Computer Vision algorithms are used in a variety of real-world applications that include surveillance and object tracking, 3D model building (photogrammetry), and object recognition. Apart from their visual appeal, these algorithms also represent elegant applications of linear algebra and optimization techniques. Topics covered in this course include a recapitulation of relevant linear algebra, introduction to face-recognition, camera calibration, stitched panoramas, edge and blob visual features, structure from motion, color-spaces, segmentation, and an introduction to object-recognition.

Recommended Knowledge

- Refresh basic programming skills in MATLAB and/or Python

- Basic knowledge of robotics middleware (RIS Lab I)

Usability and Relationship to other Modules

Giving the foundation of computer vision, this module is important for RIS project and for advanced specialization courses. This module serves as a third year Specialization module for CS major students.

Intended Learning Outcomes

No	Competence	ILO
1	Describe	Describe image formation and camera models.
2	Calibrate	Calibrate cameras.
3	Compute	Compute image histograms, and basic image processing.
4	Discriminate	Discriminate among visual features (e.g., corner, edge, blob).
5	Properly	Properly use computer vision libraries.
6	Implement	Implement computer vision applications.

Indicative Literature

- D.A. Forsyth and J. Ponce, Computer Vision: A Modern Approach. 2nd edition, 2011.
- R. Szeliski, Computer Vision: Algorithms and Applications, Springer, 2010.
- Ma et al., An Invitation to 3 D Vision: From Images to Geometric Models, Springer, 2004.

Entry Requirements

Prerequisites	Programming in C and C++
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Computer Vision	Written Examination	120 minutes	100	45%	1-6

Module Achievements: 50% if the assignments correctly solved.

7.25 Human Computer Interaction

Module Name	Human Computer Interaction
Module Code	2025-CA-S-RIS-802
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-RIS-BSc 5 - 2025-CS-BSc 5
Duration	1 Semester
Program Affiliation	2025-RIS-BSc (Robotics and Intelligent Systems)
Module Coordinator(s)	Prof. Dr. Francesco Maurelli

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	Number	Type	CP
Human Computer Interaction	CA-RIS-802	Lecture	5

Module Description

Computer systems often interact with human beings. The design of a good human-computer interface is often crucial for the acceptance and the success of a software system. Human-computer interface designs have to satisfy several requirements such as usability, learnability, efficiency, accessibility, and safety. The module discusses the evolution of human-computer interaction models and introduces design principles for graphical user interfaces and other types of interaction (e.g., visual, voice, gesture). Human-computer interaction designs are often evaluated using prototypes or mockups that can be given to test candidates to evaluate the effectiveness of the design. The module introduces evaluation strategies as well as tools and techniques that can be used to prototype human-computer interfaces.

Usability and Relationship to other Modules

Students with a strong interest in graphical user interfaces are encouraged to also select the Computer Graphics specialization module, which introduces methods and technologies for creating computer graphics and animations.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain the evolution of human-computer interaction models.

2	Design	Design and implement simple graphical user interfaces.
3	Explain	Explain ergonomic principles guiding the design of user interfaces.
4	Illustrate	Illustrate different types of interaction (e.g., visual, voice, gestures) and their usability aspects.
5	Evaluate	Evaluate aspects of and tradeoffs between usability, learnability, efficiency, and safety.
6	Apply	Apply scientific methods to evaluate interfaces with respect to their usability and other desirable properties.
7	Use	Use prototyping tools that can be employed to create mockups of user interfaces during the early stages of a software project.

Indicative Literature

- Alan Dix, Janet Finlay, Gregory D., and Russell Beale: Human-Computer Interaction, 3rd edition, Pearson, 2004.
- Ben Shneiderman, Catherine Plaisant, Maxine Cohen, Steven Jacobs, Niklas Elmqvist, Nicholas Diakopoulos: Designing the User Interface: Strategies for Effective Human-Computer Interaction, 6th edition, Pearson, 2016.
- Céline Jost, Brigitte Le Pévédic, Tony Belpaeme, Cindy Bethel, Dimitrios Chrysostomou, Nigel Crook, Marine Grandgeorge, Nicole Mirnig, Human-Robot Interaction, Evaluation Methods and Their Standardization, Springer 2020 ISBN: 978-3-030- 42306-3.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Human Computer Interaction	Project Assessment	120 minutes	100	45%	1-7

Module Achievements: None

7.26 Artificial Intelligence

Module Name	Artificial Intelligence
Module Code	2025-CO-547
Module ECTS	5
Study Semester	Mandatory status for: - 2025-RIS-BSc 4 - 2025-Minor-RIS-BSc 4 Mandatory Elective status for: - 2025-CS-BSc 4 - 2025-SDT-BSc 4
Duration	1 Semester
Program Affiliation	2025-RIS-BSc (Robotics and Intelligent Systems)
Module Coordinator(s)	Dr. Dmitry Kropotov

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	Number	Type	CP
Artificial Intelligence	CO-547-A	Lecture	5

Module Description

Artificial Intelligence (AI) is an important subdiscipline of Computer Science that deals with technologies to automate the performance of tasks that are usually associated with intelligence. AI methods have a significant application potential, as there is an increasing interest and need to generate artificial systems that can carry out complex missions in unstructured environments without permanent human supervision. The module teaches a selection of the most important methods in AI. In addition to general-purpose techniques and algorithms, it also includes aspects of methods that are especially targeted for physical systems such as intelligent mobile robots or autonomous cars.

Recommended Knowledge

Revise content of the pre-requisite modules.

Usability and Relationship to other Modules

This module gives an introduction to Artificial Intelligence (AI) excluding the aspects of machine learning (ML), which are covered in a dedicated module that complements this one.

Intended Learning Outcomes

No	Competence	ILO

1	Outline	Outline and explain the history, general developments, and application areas of AI.
2	Apply	Apply the basic concepts and methods of behavior-oriented AI.
3	Use	Use concepts and methods of search algorithms for problem-solving.
4	Explain	Explain the basic concepts of path-planning as an application example for domain-specific search.
5	Apply	Apply basic path-planning algorithms and compare their relations to general search algorithms.
6	Write	Write and explain concepts of propositional and first-order logic.
7	Use	Use logic representations and inference for basic examples of artificial planning systems.

Indicative Literature

- S. Russell and P. Norvig Artificial Intelligence: A Modern Approach, Prentice Hall, 2009.
- S.M. LaValle, Planning Algorithms. Cambridge University Press, 2006.
- J.-C. Latombe, Robot Motion Planning, Springer, 1991.

Entry Requirements

Prerequisites	Core Algorithms and Data Structures **OR** Algorithms and Data Structures
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Artificial Intelligence	Written Examination	120 minutes	100	45%	1-7

Module Achievements: None

7.27 Robotics

Module Name	Robotics
Module Code	2025-CO-540
Module ECTS	5
Study Semester	Mandatory status for: - 2025-RIS-BSc 3 -2025-Minor-RIS-BSc 3 Mandatory Elective status for: - 2025-CS-BSc 3
Duration	1 Semester
Program Affiliation	2025-RIS-BSc (Robotics and Intelligent Systems)
Module Coordinator(s)	Prof. Dr. Andreas Birk

Forms of Learning and Teaching	
Class Attendance	35
Exam Preparation	20
Independent Study	70
Workload Hours	125 hours

Module Components	Number	Type	CP
Robotics	CO-540-A	Lecture	5

Module Description

Robotics is an area that is driven by dreams from science fiction and the reality of engineering. The module intends to provide an understanding of the formal foundations of this area as well as its technological state of the art and future directions. The course accordingly gives an introduction to the core algorithmic, mathematical, and engineering concepts and methods of robotics. This includes concepts and methods that are used for well-established tools of factory automation, especially in the form of robot-arms, as well as increasingly relevant intelligent mobile systems such as autonomous cars or autonomous transport systems.

Recommended Knowledge

Revise content of the pre-requisite modules.

Usability and Relationship to other Modules

- This module serves as a third Year Specialization module for CS major students.

- This module gives an introduction to Robotics, which is a core discipline of Robotics and Intelligent System (RIS) and an important area of possible future employment.

Intended Learning Outcomes

No	Competence	ILO

1	Outline	Outline and explain the history, general developments, and application areas of robotics
2	Apply	Apply the concepts and methods to describe space and motions therein including homogeneous coordinates and transforms as well as quaternions
3	Use	Use the spatial concepts and methods for the forward kinematics (FK) of robot-arms
4	Explain	Explain basic concepts of simple actuators, including electrical motors and gear systems
5	Apply	Apply concepts and methods to derive the inverse kinematics of robot-arms and related systems such as legs in analytical and numerical forms
6	Apply	Apply concepts and methods of wheeled locomotion including FK and IK of the differential and of the omni-directional drive
7	Use	Use basic concepts and methods of dynamics
8	Explain	Explain and use core concepts and methods of global localization, e.g., multilateration and multidimensional scaling
9	Use	Use the basic concepts and methods of error propagation estimation in the context of relative localization with dead-reckoning
10	Outline	Outline and compare the basic concepts and methods of mapping

Indicative Literature

- J. J. Craig, Introduction to robotics - Mechanics and control, Prentice Hall, 2005.
- G. Dudek and M. Jenkin, Computational Principles of Mobile Robotics, Cambridge University Press, 2000.
- R.Siegwart and I.R. Nourbakhsh, Introduction to Autonomous Mobile Robots, The MIT Press, 2004.
- S.Thrun, W. Burgard and D. Fox, Probabilistic Robotics, MIT Press, 2005.
- H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki and S. Thrun, Principles of Robot Motion, MIT Press, 2005.

Entry Requirements

Prerequisites	Programming in C and C++ Mathematical and Physical Foundations of Robotics I
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Robotics	Written Examination	120 minutes	100	45%	All

Module Achievements: None

7.28 Digital Design

Module Name	Digital Design
Module Code	2025-CA-S-ECE-803
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-RIS-BSc 5 - 2025-CS-BSc 5 - 2025-ECE-BSc 5
Duration	1 Semester
Program Affiliation	2025-ECE-BSc (Electrical and Computer Engineering)
Module Coordinator(s)	Dr. Fangning Hu

Forms of Learning and Teaching	
Independent Study	90
Lecture/Laboratory	35
Workload Hours	125 hours

Module Components	Number	Type	CP
Digital Design	CA-ECE-803	Lecture and Laboratory	5

Module Description

The current trend of digital system design is towards hardware description languages (HDLs) that allow compact description of very complex hardware constructs. The module provides a sound introduction to basic components of a digital system such as logic gates, multiplexers, decoders, flip-flops and registers as well as VHDLs such as types, signals, sequential and concurrent statements. Methods and principle of designing complex digital systems such as finite state machines, hierarchical design, pipelined design, RTL design methodology and parameterized design will also be introduced. Students will learn VHDL for programming FPGA boards to realize small digital systems in hardware (i.e. on FPGA boards). Such digital systems could be adders, multiplexers, control units, multipliers, asynchronous serial communication modules (UART). At the end of the module, the students should be able to design a simple digital system by VHDL on an FPGA board.

Recommended Knowledge

Students may prepare themselves with books like “Brent E. Nelson, Designing Digital Systems, 2005” and “Pong P. Chu, RTL Hardware Design Using VHDL, A John Wiley & Sons, Inc, Publication, 2006”.

Usability and Relationship to other Modules

This module introduces how to design digital systems and how to realize them on a FPGA board which could also serve as a specialization module for students from Computer Science and Robotics and Intelligent Systems.

Intended Learning Outcomes

No	Competence	ILO
1	Understand	Understand the principle of digital system design based on standard building blocks and components.
2	Design	Design a complex digital system.
3	Understand	Understand the limitations of a given hardware platform (here FPGAs), modify algorithms where necessary, and structure them suitably in order to optimize performance and complexity.
4	Use	Use a typical development system.
5	Program	Program in VHDL.
6	Program	Program an FPGA board.

Indicative Literature

- Brent E. Nelson, Designing Digital Systems with SystemVerilog, 2018, ISBN-13: 978-1980926290.
- Pong P. Chu, RTL Hardware Design Using VHDL, Wiley-IEEE Press, 2006, ISBN-13: 978-0471720928.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Digital Design	Written Examination	120 minutes	100	45%	1-6

Module Achievements: None

7.29 Information Theory

Module Name	Information Theory
Module Code	2025-CO-525
Module ECTS	5
Study Semester	Mandatory status for: - 2025-ECE-BSc 4 Mandatory Elective status for: - 2025-CS-BSc 4 - 2025-PHDS-BSc 4 - 2025-RIS-BSc 4
Duration	1 Semester
Program Affiliation	2025-ECE-BSc (Electrical and Computer Engineering)
Module Coordinator(s)	Prof. Dr.-Ing. Mojtaba Joodaki

Forms of Learning and Teaching	
Independent Study	90
Lecture	35
Workload Hours	125 hours

Module Components	Number	Type	CP
Information Theory	CO-525-A	Lecture	5

Module Description

Information theory serves as the most important foundation for communication systems. The module provides an analytical framework for modeling and evaluating point-to-point and multi-point communication. After a short rehearsal of probability and random variables and some excursion to random number generation, the key concept of information content of a signal source and information capacity of a transmission medium are precisely defined, and their relationships to data compression algorithms and error control codes are examined in detail. The module aims to install an appreciation for the fundamental capabilities and limitations of information transmission schemes and to provide the mathematical tools for applying these ideas to a broad class of communications systems.

The module contains also a coverage of different source-coding algorithms like Huffman, Lempel-Ziv-(Welch), Shannon-Fano-Elias, Arithmetic Coding, Runlength Encoding, Move-to-Front transform, PPM, and Context Tree Weighting. In Channel coding, finite fields, some basic block and convolutional codes, and the concept of iterative decoding will be introduced. Aside from source and channel aspects, an introduction to security is given, including public-key cryptography. Information theory is a standard module in every communications-oriented Bachelor's program.

Recommended Knowledge

- Signals and Systems contents, such as DFT and convolution

- Notion of probability, combinatorics basics as taught in Methods module “Probability and Random Processes"

- Some basic knowledge of communications and sound understanding of probability is recommended. Hence, it is strongly advised to take the methods and skills course Probability and Random Processes prior to this module. Nevertheless, probability basics will also be revised within the module.

Usability and Relationship to other Modules

- Although not a mandatory prerequisite, this module is ideally taken before Coding Theory (CA-ECE-802)

- All communications-related modules are naturally based on information theory

- Students from Computer Science or related programs, also students taking Bio-informatics modules, profit from information-theoretic knowledge and source coding (compression) algorithms. Students from Computer Science would also be interested in the algebraic basics for error-correcting codes and cryptology, fields which area also introduced shortly.

- Serves as a mandatory elective 3rd year Specialization module for CS and RIS major students.

Intended Learning Outcomes

No	Competence	ILO
1	Explain	Explain what is understood as the information content of data and the corresponding limits of data compression algorithms.
2	Design	Design and apply fundamental algorithms in data compression.
3	Explain	Explain the information theoretic limits of data transmission.
4	Apply	Apply the mathematical basics of channel coding and cryptography.
5	Implement	Implement some channel coding schemes.
6	Differentiate	Differentiate the principles of encryption and authentication schemes and implement discussed procedures.

Indicative Literature

- Thomas M. Cover, Joy A. Thomas, Elements of Information Theory, 2nd ed., Wiley, Sept. 2006.
- David Salomon, Data Compression, The Complete Reference, 4th ed., Springer, 2007.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Information Theory	Written Examination	120 minutes	100	45%	1-6

Module Achievements: None

7.30 Internship / Startup and Career Skills

Module Name	Internship / Startup and Career Skills
Module Code	2025-CA-INT-900
Module ECTS	15
Study Semester	Mandatory status for: - 2025-CS-BSc 5 - 2025-CS-BSc 6 Mandatory Elective status for: None
Duration	1 Semester
Program Affiliation	Career ()
Module Coordinator(s)	Dr. Tanja Woebs Clémentine Senicourt

Forms of Learning and Teaching	
Internship	308
Internship Event	2
Independent Study	32
Interactive Learning	33
Workload Hours	375 hours

Module Components	Number	Type	CP
Internship	CA-INT-900-0	Internship	15

Module Description

The aims of the internship module are reflection, application, orientation, and development: for students to reflect on their interests, knowledge, skills, their role in society, the relevance of their major subject to society, to apply these skills and this knowledge in real life whilst getting practical experience, to find a professional orientation, and to develop their personality and in their career. This module supports the programs' aims of preparing students for gainful, qualified employment and the development of their personality.

The full-time internship must be related to the students' major area of study and extends lasts a minimum of two consecutive months, normally scheduled just before the 5th semester, with the internship event and submission of the internship report in the 5th semester. Upon approval by the SPC and SCS, the internship may take place at other times, such as before teaching starts in the 3rd semester or after teaching finishes in the 6th semester. The Study Program Coordinator or their faculty delegate approves the intended internship a priori by reviewing the tasks in either the Internship Contract or Internship Confirmation from the respective internship institution or company. Further regulations as set out in the Policies for Bachelor Studies apply.

Students will be gradually prepared for the internship in semesters 1 to 4 through a series of mandatory information sessions, seminars, and career events.

The purpose of the Career Services Information Sessions is to provide all students with basic facts about the job market in general, and especially in Germany and the EU, and services provided by the Student Career Support.

In the Career Skills Seminars, students will learn how to engage in the internship/job search, how to create a competitive application (CV, Cover Letter, etc.), and how to successfully conduct themselves at job interviews and/or assessment centers. In addition to these mandatory sections, students can customize their skill set regarding application challenges and their intended career path in elective seminars.

Finally, during the Career Events organized by the Career Service Center (e.g. the annual Constructor Career Fair and single employer events on and off campus), students will have the opportunity to apply their acquired job market skills in an actual internship/job search situation and to gain their desired internship in a high-quality environment and with excellent employers.

As an alternative to the full-time internship, students can apply for the StartUp Option. Following the same schedule as the full-time internship, the StartUp Option allows students who are particularly interested in founding their own company to focus on the development of their business plan over a period of two consecutive months. Participation in the StartUp Option depends on a successful presentation of the student's initial StartUp idea. This presentation will be held at the beginning of the 4th semester. A jury of faculty members will judge the student's potential to realize their idea and approve the participation of the students. The StartUp Option is supervised by the Faculty StartUp Coordinator. At the end of StartUp Option, students submit their business plan. Further regulations as outlined in the Policies for Bachelor Studies apply.

The concluding Internship Event will be conducted within each study program (or a cluster of related study programs) and will formally conclude the module by providing students the opportunity to present on their internships and reflect on the lessons learned within their major area of study. The purpose of this event is not only to self-reflect on the whole internship process, but also to create a professional network within the academic community, especially by entering the Alumni Network after graduation. It is recommended that all three classes (years) of the same major are present at this event to enable networking between older and younger students and to create an educational environment for younger students to observe the "lessons learned" from the diverse internships of their elder fellow students.

Recommended Knowledge

- Information provided on CSC

- Major specific knowledge and skills

- Please see the section "Knowledge Center" at JobTeaser Career Center for information on Career Skills seminar and workshop offers and for online tutorials on the job market preparation and the application process. For more information, please see https://constructor.university/student-life/career-services

- Participating in the internship events of earlier classes

Usability and Relationship to other Modules

This module applies skills and knowledge acquired in previous modules to a professional environment and provides an opportunity to reflect on their relevance in employment and society. It may lead to thesis topics.

Intended Learning Outcomes

No	Competence	ILO
1	Describe	Describe the scope and the functions of the employment market and personal career development.
2	Apply	Apply professional, personal, and career-related skills for the modern labor market, including self-organization, initiative and responsibility, communication, intercultural sensitivity, team and leadership skills, etc.
3	Independently	Independently manage their own career orientation processes by identifying personal interests, selecting appropriate internship locations or start-up opportunities, conducting interviews, succeeding at pitches or assessment centers, negotiating related employment, managing their funding or support conditions (such as salary, contract, funding, supplies, work space, etc.).
4	Apply	Apply specialist skills and knowledge acquired during their studies to solve problems in a professional environment and reflect on their relevance in employment and society.
5	Justify	Justify professional decisions based on theoretical knowledge and academic methods.
6	Reflect	Reflect on their professional conduct in the context of the expectations of and consequences for employers and their society.
7	Reflect	Reflect on and set their own targets for the further development of their knowledge, skills, interests, and values.
8	Establish	Establish and expand their contacts with potential employers or business partners, and possibly other students and alumni, to build their own professional network to create employment opportunities in the future.
9	Discuss	Discuss observations and reflections in a professional network.

Entry Requirements

Prerequisites	**Internship / Startup and Career Skills**
Co-requisites	None
Additional Remarks	At least 15 CP from CORE modules in the major

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Internship	Project Report	3500 words	100	45%	1-9

Module Achievements: None

7.31 Bachelor Thesis and Seminar CS

Module Name	Bachelor Thesis and Seminar CS
Module Code	2025-CA-CS-800
Module ECTS	15
Study Semester	Mandatory status for: - 2025-CS-BSc 6 Mandatory Elective status for: None
Duration	14-week lecture period
Program Affiliation	2025-CS-BSc (Computer Science)
Module Coordinator(s)	Study Program Chair

Forms of Learning and Teaching	
Independent Study	350
Seminar	25
Workload Hours	375 hours

Module Components	Number	Type	CP
Thesis CS	CA-CS-800-T	Thesis	12
Thesis Seminar CS	CA-CS-800-S	Seminar	3

Module Description

This module is a mandatory graduation requirement for all undergraduate students to demonstrate their ability to address a problem from their respective major subject independently using academic/scientific methods within a set time frame. Although supervised, this module requires students to be able to work independently and systematically and set their own goals in exchange for the opportunity to explore a topic that excites and interests them personally and that a faculty member is interested in supervising. Within this module, students apply their acquired knowledge about their major discipline and their learned skills and methods for conducting research, ranging from the identification of suitable (short-term) research projects, preparatory literature searches, the realization of discipline-specific research, and the documentation, discussion, interpretation, and communication of research results.

This module consists of two components, an independent thesis and an accompanying seminar. The thesis component must be supervised by a Constructor University faculty member and requires short-term research work, the results of which must be documented in a comprehensive written thesis including an introduction, a justification of the methods, results, a discussion of the results, and a conclusion. The seminar provides students with the opportunity to practice their ability to present, discuss, and justify their and other students' approaches, methods, and results at various stages of their research in order to improve their academic writing, receive and reflect on formative feedback, and therefore grow personally and professionally.

Recommended Knowledge

- Identify an area or a topic of interest and discuss this with your prospective supervisor in a timely manner.

- Create a research proposal including a research plan to ensure timely submission.

- Ensure you possess all required technical research skills or are able to acquire them on time.

- Review the University's Code of Academic Integrity and Guidelines to Ensure Good Academic Practice

Usability and Relationship to other Modules

This module builds on all previous modules in the undergraduate program. Students apply the knowledge, skills, and competencies they have acquired and practiced during their studies, including research methods and their ability to acquire additional skills independently as and if required.

Intended Learning Outcomes

No	Competence	ILO
1	Apply	Apply their knowledge and understanding to a context of their choice.
2	Independently	Independently plan and organize advanced learning processes.
3	Design	Design and implement appropriate research methods, taking full account of the range of alternative techniques and approaches.
4	Collect	Collect, assess, and interpret relevant information.
5	Draw	Draw scientifically-founded conclusions that consider social, scientific, and ethical factors.
6	Develop	Develop, formulate, and advance solutions to problems and debates within their subject area, and defend these through argument.
7	Discuss	Discuss information, ideas, problems, and solutions with specialists and non-specialists.

Indicative Literature

- Justin Zobel, Writing for Computer Science, 3rd edition, Springer, 2015.

Entry Requirements

Prerequisites	Bachelor Thesis and Seminar CS
Co-requisites	None
Additional Remarks	Students must have taken and successfully passed a total of at least 30 CP from advanced modules, and of those, at least 20 CP from advanced modules in the major.

Assessment and Completion

Components	Examination Type	Duration/ Length	Weight (%)	Minimum	ILOs

Thesis CS	Thesis	6,000-8,000 Words (15-25 Pages) excluding front and back matter.	80	45%	All, mainly 1-6.
Thesis Seminar CS	Presentation	15- 30 minutes	20	45%	All, Mainly 6-7

Module Achievements: None

8 Constructor Track Modules

8.1 Methods Modules

8.1.1 Elements of Linear Algebra

Module Name	**Elements of Linear Algebra**
Module Code	2025-CTMS-MAT-24
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-RIS-BSc 1 - 2025-CS-BSc 1 - 2025-SDT-BSc 1 - 2025-F-ACS-BSc 1
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Keivan Mallahi Karai

Forms of Learning and Teaching	
Lecture	35
Independent Study	90
Workload Hours	125 hours

Module Components	**Number**	**Type**	**CP**
Elements of Linear Algebra	CTMS-24	Lecture	5

Module Description

This module is the first in a sequence introducing mathematical methods at the university level in a form relevant for study and research in the quantitative natural sciences, engineering, Computer Science. The emphasis in these modules is on training

operational skills and recognizing mathematical structures in a problem context. Mathematical rigor is used where appropriate. However, a full axiomatic treatment of the subject is provided in the first-year modules “Analysis” and “Linear Algebra”.

The lecture comprises the following topics:

- Review of elementary analytic geometry

- Vector spaces, linear independence, bases, coordinates

- Matrices and matrix algebra

- Solving linear systems by Gauss elimination, structure of general solution

- LU decomposition and matrix inverse

- Linear maps and connection to matrices

- Determinant

- Eigenvalues and eigenvectors

- Hermitian and skew-Hermitian matrices

- Orthonormal bases, Gram-Schmidt orthonormalization and QR decomposition

- Fourier transform

- Singular value decomposition

- Principal Component Analysis and best low rank approximations

Recommended Knowledge

- Knowledge of Pre-Calculus at High School level (Functions, inverse functions, sets, real numbers, trigonometric functions, parametric equations, tangent lines, graphs, elementary methods for solving systems of linear and nonlinear equations)

- Knowledge of Analytic Geometry at High School level (vectors, lines, planes, reflection, rotation, translation, dot product, cross product, normal vector, polar coordinates)

- Review all of higher-level High School Mathematics, in particular the topics explicitly named in "Entry Requirements – Knowledge, Ability, or Skills" above.

Usability and Relationship to other Modules

A rigorous treatment of this topic is provided in the module "Linear Algebra."

Intended Learning Outcomes

No	Competence	ILO
1	Apply	Apply the methods described in the content section of this module description to the extent that they can solve standard textbook problems reliably and with confidence.
2	Recognize	Recognize the mathematical structures in an unfamiliar context and translate them into a mathematical problem statement.
3	Recognize	Recognize common mathematical terminology and concepts used in textbooks and research papers in computer science, engineering, and mathematics to the extent that they fall into the content categories covered in this module.
4	Independently	Independently prove results which are direct consequences of those proved in the lectures
5	Understand	Understand and use fundamental mathematical terminology to communicate mathematical ideas.

Indicative Literature

- Gilbert Strang, Introduction to Linear Algebra, Fifth Edition (2016).
- S.A. Leduc Linear Algebra. Hoboken: Wiley (2003).

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Elements of Linear Algebra	Written Examination	120 minutes	100	45%	1-5

Module Achievements: None

8.1.2 Elements of Calculus

Module Name	Elements of Calculus
Module Code	2025-CTMS-MAT-25
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-RIS-BSc 2 - 2025-CS-BSc 2 - 2025-SDT-BSc 2 - 2025-F-ACS-BSc 2
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Keivan Mallahi Karai

Forms of Learning and Teaching	
Lecture	35
Independent Study	90
Workload Hours	125 hours

Module Components	Number	Type	CP
Elements of Calculus	CTMS-25	Lecture	5

Module Description

This module is the second in a sequence introducing mathematical methods at the university level in a form relevant for study and research in the quantitative natural sciences, engineering, Computer Science. The emphasis in these modules is on training operational skills and recognizing mathematical structures in a problem context. Mathematical rigor is used where appropriate. However, a full axiomatic treatment of the subject is provided in the first-year modules “Analysis”.

The lecture comprises the following topics:

- Sets, basic operations, and relations

- Functions, basic operations, compositions of functions, graphs of functions

- Brief introduction to real and complex numbers

- Limits for sequences and functions

- Continuity

- Derivatives of functions and its geometric interpretations

- Computing derivatives: linearity, product rule, chain rule

- Applications of derivatives, optimization for one-variable functions

- Introduction to Integration and the Fundamental Theorem of Calculus

- Differential equations, modeling simple dynamical systems

- Discrete derivative, summations, difference equations

- Functions of several variables, representations using graphs and level curves

- Basic ideas of multivariable calculus

- Partial derivatives and directional derivatives, total derivative

- Optimization in several variables, gradient descent, Lagrange multipliers

- Ordinary differential equations with several variables, simple examples

- Linear constant-coefficient ordinary differential equations

- Fourier series and their applications

Recommended Knowledge

- Knowledge of Pre-Calculus at High School level (Functions, inverse functions, sets, real numbers, polynomials, rational functions, trigonometric functions, logarithm and exponential function, parametric equations, tangent lines, graphs.)

- Knowledge of Analytic Geometry at High School level (vectors, lines, planes, reflection, rotation, translation, dot product, cross product, normal vector, polar coordinates)

- Some familiarity with elementary Calculus (limits, derivative) is helpful, but not strictly required.

- Review the content of Linear Algebra

Usability and Relationship to other Modules

A rigorous treatment of this topic is provided in the module "Analysis"

Intended Learning Outcomes

No	Competence	ILO
1	Apply	Apply the methods described in the content section of this module description to the extent that they can solve standard textbook problems reliably and with confidence.
2	Recognize	Recognize the mathematical structures in an unfamiliar context and translate them into a mathematical problem statement.
3	Recognize	Recognize common mathematical terminology and concepts used in textbooks and research papers in computer science, engineering, and mathematics to the extent that they fall into the content categories covered in this module.
4	Independently	Independently prove results which are direct consequences of those proved in the lectures.
5	Understand	Understand and use fundamental mathematical terminology to communicate mathematical ideas.

Indicative Literature

- James Stewart, Calculus: Early Transcendentals, (2015).
- S.I. Grossman, Calculus of one variable, 2nd edition, (2014).

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Elements of Calculus	Written Examination	120 minutes	100	45%	1-5

Module Achievements: None

8.1.3 Probability and Random Processes

Module Name	Probability and Random Processes
Module Code	2025-CTMS-MAT-12
Module ECTS	5
Study Semester	Mandatory status for: - 2025-CS-BSc 3 Mandatory Elective status for: None
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Keivan Mallahi Karai

Forms of Learning and Teaching	
Independent Study	90
Lecture	35
Workload Hours	125 hours

Module Components	Number	Type	CP
Probability and random processes	CTMS-12	Lecture	5

Module Description

This module aims to provide a basic knowledge of probability theory and random processes suitable for students in engineering, Computer Science, and Mathematics. The module provides students with basic skills needed for formulating real-world problems dealing with randomness and probability in mathematical language, and methods for applying a toolkit to solve these problems. Mathematical rigor is used where appropriate. A more advanced treatment of the subject is deferred to the third-year module Stochastic Processes.

The lecture comprises the following topics:

- Brief review of number systems, elementary functions, and their graphs

- Outcomes, events and sample space

- Combinatorial probability

- Conditional probability and Bayes' formula

- Binomials and Poisson-Approximation

- Random Variables, distribution and density functions

- Independence of random variables

- Conditional Distributions and Densities

- Transformation of random variables

- Joint distribution of random variables and their transformations

- Expected Values and Moments, Covariance

- High dimensional probability: Chebyshev and Chernoff bounds

- Moment-Generating Functions and Characteristic Functions

- The Central limit theorem

- Random Vectors and Moments, Covariance matrix, Decorrelation

- Multivariate normal distribution. Markov chains, stationary distributions.

Recommended Knowledge

- Review all of the first-year calculus and linear algebra modules as indicated in "Entry Requirements - Knowledge, Ability, or Skills" above.

- Knowledge of calculus at the level of a first year calculus module (differentiation, integration with one and several variables, trigonometric functions, logarithms and exponential functions).

- Knowledge of linear algebra at the level of a first year university module (eigenvalues and eigenvectors, diagonalization of matrices).

- Some familiarity with elementary probability theory at the high school level.

Usability and Relationship to other Modules

Students taking this module are expected to be familiar with basic tools from calculus and linear algebra.

Intended Learning Outcomes

No	Competence	ILO
1	Command	Command the methods described in the content section of this module description to the extent that they can solve standard text-book problems reliably and with confidence.
2	Recognize	Recognize the probabilistic structures in an unfamiliar context and translate them into a mathematical problem statement.
3	Recognize	Recognize common mathematical terminology used in textbooks and research papers in the quantitative sciences, engineering, and mathematics to the extent that they fall into the content categories covered in this module.

Indicative Literature

- J. Hwang and J.K. Blitzstein (2019). Introduction to Probability, second edition. London: Chapman & Hall.
- S. Ghahramani. Fundamentals of Probability with Stochastic Processes, fourth edition. Upper Saddle River: Prentice Hall.

Entry Requirements

Prerequisites	Elements of Linear Algebra Elements of Calculus Matrix Algebra and Advanced Calculus I Matrix Algebra and Advanced Calculus II
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Probability and random processes	Written Examination	120 minutes	100	45%	1-3

Module Achievements: None

8.1.4 Statistics and Data Analytics

Module Name	**Statistics and Data Analytics**
Module Code	2025-CTMS-MET-21
Module ECTS	5
Study Semester	Mandatory status for: - 2025-MMDA-BSc 4 - 2025-PHDS-BSc 4 - 2025-SDT-BSc 4 Mandatory Elective status for: - 2025-CS-BSc 4
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Dr. Ivan Ovsyannikov

Forms of Learning and Teaching	
Independent Study	105
Lecture	35
Workload Hours	140 hours

Module Components	**Number**	**Type**	**CP**
Statistics and Data Analytics	CTMS-21	Lecture	5

Module Description

The aim of this module is to introduce students to basic ideas and methods used for analysing large and complex datasets. While the first modern statistical toolkits date back to the beginning of the twentieth century, the advent of the computer age and the availability of fast computations has led to dramatic changes in the field.

Statistical models have found applications in many areas ranging from business and healthcare to astrophysics and speech recognition. Such models are used to make predictions, draw inferences and support policy decisions in all these areas.

This module draws on students' knowledge from the module Probability and Random Processes to help them build and analyze statistical models, ranging in their degree of sophistication from basis to more advanced ones, and apply them to real-world situations.

The module will cover the following topics:

- Classical statistics: descriptive and inferential modes, parameter estimation and hypothesis testing.

- Linear regressions, multiple linear regressions

- Classification: logistic regression, generative models for classification

- Resampling methods, bootstrap

- Non-linear models, splines

- Support vector machines

- Basic ideas of deep learning

Recommended Knowledge

- Good command of basic probability

- Recap Probability and Random Processes

Usability and Relationship to other Modules

- This module is part of the core education in Mathematics, Modeling and Data Analytics and Physics and Data Science.

- It is also valuable for students in Computer Science, RIS, and ECE, either as part of a minor in Mathematics, or as an elective module.

Intended Learning Outcomes

No	Competence	ILO
1	Formulate	Formulate statistical models for real world problems.
2	Describe	Describe statistical methods for analyzing real world problems.
3	Explain	Explain the importance of linear and non-linear models.
4	Recognize	Recognize different solution methods for modeling problems.
5	Illustrate	Illustrate the use of regressions, resampling, support vector machines and other statistical tools to describe phenomena in the real world.
6	Describe	Describe basic ideas of deep learning.

Indicative Literature

- James, Witten, Hastie, Tibshirani. An introduction to Statistical learning, second edition.

Entry Requirements

Prerequisites	Probability and Random Processes
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Statistics and Data Analytics	Written Examination	120 minutes	100	45%	1-6

Module Achievements: None

8.1.5 Numerical Methods

Module Name	**Numerical Methods**
Module Code	2025-CTMS-MAT-13
Module ECTS	5
Study Semester	Mandatory status for: - 2025-ECE-BSc 4 Mandatory Elective status for: - 2025-CS-BSc 4 - 2025-RIS-BSc 4
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Keivan Mallahi Karai

Forms of Learning and Teaching	
Independent Study	90
Lecture	35
Workload Hours	125 hours

Module Components	**Number**	**Type**	**CP**
Numerical Methods	CTMS-13	Lecture	5

Module Description

This module covers calculus-based numerical methods, in particular root finding, interpolation, approximation, numerical differentiation, numerical integration (quadrature), and a first introduction to the numerical solution of differential equations.

The lecture comprises the following topics:

- number representations

- Gaussian elimination

- LU decomposition

- Cholesky decomposition

- iterative methods

- bisection method

- Newton’s method

- secant method

- polynomial interpolation

- Aitken’s algorithm

- Lagrange interpolation

- Newton interpolation

- Hermite interpolation

- Bezier curves

- De Casteljau's algorithm

- piecewise interpolation

- Spline interpolation

- B-Splines

- Least-squares approximation

- polynomial regression

- difference schemes

- Richardson extrapolation

- Quadrature rules

- Monte Carlo integration

- time stepping schemes for ordinary differential equations

- Runge Kutta schemes

- finite difference method for partial differential equations

Recommended Knowledge

- Taking Calculus and Elements of Linear Algebra II before taking this module is recommended, but not required. A thorough review of Calculus and Elements of Linear Algebra, with emphasis on the topics listed below is recommended.

- Knowledge of Calculus (functions, inverse functions, sets, real numbers, sequences and limits, polynomials, rational functions, trigonometric functions, logarithm and exponential function, parametric equations, tangent lines, graphs, derivatives, anti-derivatives, elementary techniques for solving equations).

- Knowledge of Linear Algebra (vectors, matrices, lines, planes, n-dimensional Euclidean vector space, rotation, translation, dot product (scalar product), cross product, normal vector, eigenvalues, eigenvectors, elementary techniques for solving systems of linear equations).

Usability and Relationship to other Modules

This module is a co-recommendation for the module "Applied Dynamical Systems Lab", in which the actual implementation in a high-level programming language of the learned methods will be covered.

Intended Learning Outcomes

No	Competence	ILO

1	Describe	Describe the basic principles of discretization used in the numerical treatment of continuous problems.
2	Command	Command the methods described in the content section of this module description to the extent that they can solve standard text-book problems reliably and with.
3	Recognize	Recognize mathematical terminology used in textbooks and research papers on numerical methods in the quantitative sciences, engineering, and mathematics to the extent that they fall into the content categories covered in this module.
4	Implement	Implement simple numerical algorithms in a high-level programming language.
5	Understand	Understand the documentation of standard numerical library code and understand the potential limitations and caveats of such algorithms.

Indicative Literature

- D. Kincaid and W. Cheney (1991). Numerical Analysis: Mathematics of Scientific Computing. Pacific Grove: Brooks/Cole Publishing.
- W Boehm and H Prautzsch (1993). Numerical Methods. Natick: AK Peters.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Numerical Methods	Written Examination	120 minutes	100	45%	1-5

Module Achievements: None

8.1.6 Matrix Algebra and Advanced Calculus I

Module Name	**Matrix Algebra and Advanced Calculus I**
Module Code	2025-CTMS-MAT-22
Module ECTS	5
Study Semester	Mandatory status for: - 2025-PHDS-BSc 1 - 2025-ECE-BSc 1 - 2025-PHDS-BSc 2 Mandatory Elective status for: - 2025-RIS-BSc 1 - 2025-CS-BSc 1 - 2025-SDT-BSc 1
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Keivan Mallahi Karai

Forms of Learning and Teaching	
Independent Study	90
Lecture	35
Workload Hours	125 hours

Module Components	**Number**	**Type**	**CP**
Matrix Algebra and Advanced Calculus I	CTMS-22	Lecture	5

Module Description

This module is the first in a sequence including advanced mathematical methods at the university level at a level higher than the course Calculus and Linear Algebra I.

The course comprises the following topics:

- Number systems, complex numbers

- The concept of function, composition of functions, inverse functions

- Basic ideas of calculus: Archimedes to Newton

- The notion of limit for functions and sequences and series

- Continuous function and their basic properties

- Derivatives: rate of change, velocity and applications

- Mean value theorem and estimation, maxima and minima, convex functions

- Integration, change of variables, Fundamental Theorem of Calculus

- Applications of the integral: work, area, average value, centre of mass

- Improper Integrals, Mean value theorem for integrals

- Taylor series

- Ordinary differential equations, examples, solving first order linear differential equations

- Basic ideas of numerical analysis, Newton's method, asymptotic formulas

- Review of elementary analytic geometry, lines, conics

- Vector spaces, linear independence, bases, coordinates

- Linear maps, matrices and their algebra, matrix inverses

- Gaussian elimination, solution space

- Determinants

Recommended Knowledge

- Knowledge of pre-calculus ideas (sets and functions, elementary functions, polynomials) and analytic geometry (equations of lines, systems of linear equations, dot product, polar coordinates) at High School level. Familiarity with ideas of calculus is helpful.

- Review of high school mathematics.

Usability and Relationship to other Modules

- Calculus and Linear Algebra I can be substituted with this module after consulting academic advisor

- A more advanced treatment of multi-variable Calculus, in particular, its applications in Physics and Mathematics, is provided in the second-semester module "Applied Mathematics". All students taking "Applied Mathematics" are expected to take this module as well as the module topics are closely synchronized.

- The second-semester module "Linear Algebra" provides a complete proof-driven development of the theory of Linear Algebra. Diagonalization is covered more abstractly, with particular emphasis on degenerate cases. The Jordan normal form is also covered in "Linear Algebra", not in this module.

Intended Learning Outcomes

No	Competence	ILO
1	Apply	Apply the methods described in the content section of this module description to the extent that they can
2	Solve	Solve standard text-book problems reliably and with confidence
3	Recognize	Recognize the mathematical structures in an unfamiliar context and translate them into a mathematical problem statement
4	Recognize	Recognize common mathematical terminology used in textbooks and research papers in the quantitative sciences, engineering, and mathematics to the extent that they fall into the content categories covered in this module

Indicative Literature

- Advanced Calculus, G.B. Folland (Pearson 2002).
- Linear Algebra, S. Lang (Springer Verlag 1986).
- Mathematical Methods for Physics and Engineering
- K. Riley, M. Hobson, S. Bence (Cambridge University Press 2006).

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Matrix Algebra and Advanced Calculus I	Written Examination	120 minutes	100	45%	1-4

Module Achievements: None

8.1.7 Matrix Algebra and Advanced Calculus II

Module Name	**Matrix Algebra and Advanced Calculus II**
Module Code	2025-CTMS-MAT-23
Module ECTS	5
Study Semester	Mandatory status for: - 2025-ECE-BSc 2 - 2025-PHDS-BSc 2 Mandatory Elective status for: - 2025-RIS-BSc 2 - 2025-CS-BSc 2 - 2025-SDT-BSc 2
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Keivan Mallahi Karai

Forms of Learning and Teaching	
Independent Study	90
Lecture	35
Workload Hours	125 hours

Module Components	**Number**	**Type**	**CP**
Matrix Algebra and Advanced Calculus II	CTMS-23	Lecture	5

Module Description

- Coordinate systems, functions of several variables, level curves, polar coordinates

- Continuity, directional derivatives, partial derivatives, chain rule (version I)

- derivative as a matrix, chain rule (version II), tangent planes and linear approximation, gradient, repeated partial derivatives

- Minima and Maxima of functions of several variables, Lagrange multipliers

- Multiple integrals, iterated integrals, integration over standard regions, change of variables formula

- Vector fields, parametric representation of curves, line integrals and arc length, conservative vector fields

- Potentials, Green's theorem in the plane

- Parametric representation of surfaces

- Vector products and normal surface integrals

- Integral theorems by Stokes and Gauss, physical interpretations

- Basics of differential forms and their calculus, connection to gradient, curl, and divergence

- Eigenvalues and eigenvectors, diagonalisable matrices

- Inner product spaces, Hermitian and unitary matrices

- Matrix factorizations: Singular value decomposition with applications, LU decomposition, QR decomposition

- Linear constant-coefficient ordinary differential equations, application to mechanical vibrations and electrical

oscillations

- Periodic functions, Fourier series

Recommended Knowledge

Review the content of Matrix Algebra and Advanced Calculus I

Usability and Relationship to other Modules

- This module can substitute Calculus and Linear Algebra II after consulting academic advisor.

- Methods of this course are applied in the module Mathematical Modeling.

- The second-semester module Linear Algebra provides a more rigorous and more abstract treatment of some of the notions discussed in this module.

Intended Learning Outcomes

No	Competence	ILO
1	Understand	Understand the definitions of continuity, derivative of a function as a linear transformation, multivariable integrals, eigenvalues and eigenvectors and associated notions.
2	Apply	Apply the methods described in the content section of this module description to the extent that they can.
3	Evaluate	Evaluate multivariable integrals using definitions or by applying Green and Stokes theorem.
4	Evaluate	Evaluate various decompositions of matrices.
5	Solve	Solve standard text-book problems reliably and with confidence.
6	Recognize	Recognize the mathematical structures in an unfamiliar context and translate them into a mathematical problem statement.
7	Recognize	Recognize common mathematical terminology used in textbooks and research papers in the quantitative sciences, engineering, and mathematics to the extent that they fall into the content categories covered in this module.

Indicative Literature

- Advanced Calculus GB Folland (Pearson 2002).
- Linear Algebra S Lang (Springer Verlag 1986).
- Mathematical Methods for Physics and Engineering.

- K Riley M Hobson S Bence (Cambridge University Press 2006).
- Vector Calculus Linear Algebra and Differential Forms: A Unified.
- Approach JH Hubbard B Hubbard (Pearson 1998).

Entry Requirements

Prerequisites	Matrix Algebra and Advanced Calculus I
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Matrix Algebra and Advanced Calculus II	Written Examination	120 minutes	100	45%	1-7

Module Achievements: None

8.2 New Skills

8.2.1 Logic (perspective I)

Module Name	Logic (perspective I)
Module Code	2025-CTNS-NSK-01
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 3
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Jules Coleman

Forms of Learning and Teaching	
Independent Study	45
Online Lecture	17.5
Workload Hours	62.5 hours

Module Components	Number	Type	CP
Logic (perspective I)	CTNS-01	Lecture (Online)	2.5

Module Description

Suppose a friend asks you to help solve a complicated problem? Where do you begin? Arguably, the first and most difficult task you face is to figure out what the heart of the problem actually is. In doing that you will look for structural similarities between the problem posed and other problems that arise in different fields that others may have addressed successfully. Those similarities may point you to a pathway for resolving the problem you have been asked to solve. But it is not enough to look for structural similarities. Sometimes relying on similarities may even be misleading. Once you've settled tentatively on what you take to be the heart of the matter, you will naturally look for materials, whether evidence or arguments, that you believe is relevant to its potential solution. But the evidence you investigate of course depends on your formulation of the problem, and your formulation of the problem likely depends on the tools you have available - including potential sources of evidence and argumentation. You cannot ignore this interactivity, but you can't allow yourself to be hamstrung entirely by it. But there is more. The problem itself may be too big to be manageable all at once, so you will have to explore whether it can be broken into manageable parts and if the information you have bears on all or only some of those parts. And later you will face the problem of whether the solutions to the particular sub problems can be put together coherently to solve the entire problem taken as a whole.

What you are doing is what we call engaging in computational thinking. There are several elements of computational thinking illustrated above. These include: Decomposition (breaking the larger problem down into smaller ones); Pattern recognition (identifying structural similarities); Abstraction (ignoring irrelevant particulars of the problem): and Creating Algorithms), problem-solving formulas.

But even more basic to what you are doing is the process of drawing inferences from the material you have. After all, how else are you going to create a problem-solving formula, if you draw incorrect inferences about what information has shown and what, if anything follows logically from it. What you must do is apply the rules of logic to the information to draw inferences that are warranted.

We distinguish between informal and formal systems of logic, both of which are designed to indicate fallacies as well as warranted inferences. If I argue for a conclusion by appealing to my physical ability to coerce you, I prove nothing about the truth of what I claim. If anything, by doing so I display my lack of confidence in my argument. Or if the best I can do is berate you for your skepticism, I have done little more than offer an ad hominem instead of an argument. Our focus will be on formal systems of logic, since they are at the heart of both scientific argumentation and computer developed algorithms. There are in fact many different kinds of logic and all figure to varying degrees in scientific inquiry. There are inductive types of logic, which purport to formalize the relationship between premises that if true offer evidence on behalf of a conclusion and the conclusion and are represented as claims about the extent to which the conclusion is confirmed by the premises. There are deductive types of logic, which introduce a different relationship between premise and conclusion. These variations of logic consist in rules that if followed entail that if the premises are true then the conclusion too must be true.

There are also modal types of logic which are applied specifically to the concepts of necessity and possibility, and thus to the relationship among sentences that include either or both those terms. And there is also what are called deontic logic, a modification of logic that purport to show that there are rules of inference that allow us to infer what we ought to do from facts about the circumstances in which we find ourselves. In the natural and social sciences most of the emphasis has been placed on inductive logic, whereas in math it is placed on deductive logic, and in modern physics there is an increasing interest in the concepts of possibility and necessity and thus in modal logic. The humanities, especially normative discussions in philosophy and literature are the province of deontic logic.

This module will also take students through the central aspects of computational thinking, as it is related to logic; it will introduce the central concepts in each, their relationship to one another and begin to provide the conceptual apparatus and practical skills for scientific inquiry and research.

Intended Learning Outcomes

No	Competence	ILO
1	Apply	Apply the various principles of logic and expand them to computational thinking.
2	Understand	Understand the way in which logical processes in humans and in computers are similar and different at the same time.
3	Apply	Apply the basic rules of first-order deductive logic and employ them rules in the context of creating a scientific or social scientific study and argument.
4	Employ	Employ those rules in the context of creating a scientific or social scientific study and argument.

Indicative Literature

- Frege, Gottlob (1879), Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens [Translation: A Formal Language for Pure Thought Modeled on that of Arithmetic],Halle an der Salle: Verlag von Louis Nebert.
- Gödel, Kurt (1986), Russels mathematische Logik. In: Alfred North Whitehead, Bertrand Russell: Principia Mathematica. Vorwort, S. V–XXXIV. Suhrkamp.
- Leeds, Stephen. "George Boolos and Richard Jeffrey. Computability and logic. Cambridge University Press, New York and London1974, x+ 262 pp." The Journal of Symbolic Logic 42.4 (1977): 585-586.
- Kubica, Jeremy. Computational fairy tales. Jeremy Kubica, 2012.
- McCarthy, Timothy. "Richard Jeffrey. Formal logic: Its scope and limits. of XXXVIII 646. McGraw-Hill Book Company, New York etc. 1981, xvi+ 198 pp." The Journal of Symbolic Logic 49.4 (1984): 1408-1409.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Logic (perspective I)	Written Examination	60 minutes	100	45%	All

Module Achievements: None

8.2.2 Logic (perspective II)

Module Name	Logic (perspective II)
Module Code	2025-CTNS-NSK-02
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 3
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Jules Coleman

Forms of Learning and Teaching	
Independent Study	45
Online Lecture	17.5
Workload Hours	62.5 hours

Module Components	Number	Type	CP
Logic (perspective II)	CTNS-02	Lecture (Online)	2.5

Module Description

The focus of this module is on formal systems of logic, since they are at the heart of both scientific argumentation and computer developed algorithms. There are in fact many kinds of logic and all figure to varying degrees in scientific inquiry. There are inductive types of logic, which purport to formalize the relationship between premises that if true offer evidence on behalf of a conclusion and the conclusion and are represented as claims about the extent to which the conclusion is confirmed by the premises. There are deductive types of logic, which introduce a different relationship between premise and conclusion. These variations of logic consist in rules that if followed entail that if the premises are true then the conclusion too must be true.

This module introduces logics that go beyond traditional deductive propositional logic and predicate logic and as such it is aimed at students who are already familiar with basics of traditional formal logic. The aim of the module is to provide an overview of alternative logics and to develop a sensitivity that there are many different logics that can provide effective tools for solving problems in specific application domains.

The module first reviews the principles of a traditional logic and then introduces many-valued logics that distinguish more than two truth values, for example true, false, and unknown. Fuzzy logic extends traditional logic by replacing truth values with real numbers in the range 0 to 1 that are expressing how strong the believe into a proposition is. Modal logics introduce modal operators expressing whether a proposition is necessary or possible. Temporal logics deal with propositions that are qualified by time. One can view temporal logics as a form of modal logics where propositions are qualified by time constraints. Interval temporal logic provides a way to reason about time intervals in which propositions are true.

The module will also investigate the application of logic frameworks to specific classes of problems. For example, a special subset of predicate logic, based on so-called Horn clauses, forms the basis of logic programming languages such as Prolog. Description logics, which are usually decidable logics, are used to model relationships and they have applications in the semantic web, which enables search engines to reason about resources present on the Internet.

Intended Learning Outcomes

No	Competence	ILO
1	Apply	Apply the various principles of logic.
2	Explain	Explain practical relevance of non-standard logic.
3	Describe	Describe how many-valued logic extends basic predicate logic.
4	Apply	Apply basic rules of fuzzy logic to calculate partial truth values.
5	Sketch	Sketch basic rules of temporal logic.
6	Implement	Implement predicates in a logic programming language.
7	Prove	Prove some simple non-standard logic theorems.

Indicative Literature

- Bergmann, Merry. “An Introduction to Many-Valued and Fuzzy Logic: Semantics, Algebras, and Derivation Systems”, Cambridge University Press, April 2008.
- Sterling, Leon S., Ehud Y. Shapiro, Ehud Y. "The Art of Prolog", 2nd edition, MIT Press, March 1994.
- Fisher, Michael. "An Introduction to Practical Formal Methods Using Temporal Logic", Wiley, Juli 2011.
- Baader, Franz. "The Description Logic Handbook: Theory Implementation and Applications", Cambridge University Press, 2nd edition, May 2010.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Logic (perspective II)	Written Examination	60 minutes	100	45%	All

Module Achievements: None

8.2.3 Causation and Correlation (perspective I)

Module Name	Causation and Correlation (perspective I)
Module Code	2025-CTNS-NSK-03
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 4
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Jules Coleman

Forms of Learning and Teaching	
Independent Study	45
Online Lecture	17.5
Workload Hours	62.5 hours

Module Components	Number	Type	CP
Causation and Correlation	CTNS-03	Lecture (Online)	2.5

Module Description

In many ways, life is a journey. And also, as in other journeys, our success or failure depends not only on our personal traits and character, our physical and mental health, but also on the accuracy of our map. We need to know what the world we are navigating is actually like, the how, why and the what of what makes it work the way it does. The natural sciences provide the most important tool we have developed to learn how the world works and why it works the way it does. The social sciences provide the most advanced tools we have to learn how we and other human beings, similar in most ways, different in many others, act and react and what makes them do what they do. In order for our maps to be useful, they must be accurate and correctly reflect the way the natural and social worlds work and why they work as they do.

The natural sciences and social sciences are blessed with enormous amounts of data. In this way, history and the present are gifts to us. To understand how and why the world works the way it does requires that we are able to offer an explanation of it. The data supports a number of possible explanations of it. How are we to choose among potential explanations? Explanations, if sound, will enable us to make reliable predictions about what the future will be like, and also to identify many possibilities that may unfold in the future. But there are differences not just in the degree of confidence we have in our predictions, but in whether some of them are necessary future states or whether all of them are merely possibilities? Thus, there are three related activities at the core of scientific inquiry: understanding where we are now and how we got here (historical); knowing what to expect going forward (prediction); and exploring how we can change the paths we are on (creativity).

At the heart of these activities are certain fundamental concepts, all of which are related to the scientific quest to uncover immutable and unchanging laws of nature. Laws of nature are thought to reflect a causal nexus between a previous event and a future one. There are also true statements that reflect universal or nearly universal connections between events past and present that are not laws of nature because the relationship they express is that of a correlation between events. A working thermostat accurately allows us to determine or even to predict the temperature in the room in which it is located, but it does not explain why the room has the temperature it has. What then is the core difference between causal relationships and correlations? At the same time, we all recognize that given where we are now there are many possible futures for each of us, and even had our lives gone just the slightest bit differently than they have, our present state could well have been very different than it is. The relationship between possible pathways between events that have not materialized but could have is expressed through the idea of counterfactual.

Creating accurate roadmaps, forming expectations we can rely on, making the world a more verdant and attractive place requires us to understand the concepts of causation, correlation, counterfactual explanation, prediction, necessity, possibility, law of nature and universal generalization. This course is designed precisely to provide the conceptual tools and intellectual skills to implement those concepts in our future readings and research and ultimately in our experimental investigations, and to employ those tools in various disciplines.

Intended Learning Outcomes

No	Competence	ILO
1	Formulate	Formulate testable hypotheses that are designed to reveal causal connections and those designed to reveal interesting, important and useful correlations.
2	Distinguish	Distinguish scientifically interesting correlations from unimportant ones.
3	Apply	Apply critical thinking skills to evaluate information.
4	Understand	Understand when and why inquiry into unrealized possibility is important and relevant.

Indicative Literature

- Thomas S. Kuhn: The Structure of Scientific Revolutions. Nelson, fourth edition, 2012.
- Goodman, Nelson. Fact, fiction, and forecast. Harvard University Press, 1983.
- Quine Willard, Van Orman, and Joseph Silbert Ullian. The web of belief. Vol 2. New York: Random house, 1978.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimu m	ILOs

Causation and Correlation	Written Examination	60 minutes	100	45%	1-4

Module Achievements: None

8.2.4 Causation and Correlation (perspective II)

Module Name	Causation and Correlation (perspective II)
Module Code	2025-CTNS-NSK-04
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 4
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Dr. Eoin Ryan Dr. Irina Chiaburu Prof. Dr. Keivan Mallahi Karai

Forms of Learning and Teaching	
Independent Study	45
Online Lecture	17.5
Workload Hours	62.5 hours

Module Components	Number	Type	CP
Causation and Correlation (perspective II)	CTNS-04	Lecture (Online)	2.5

Module Description

Causality or causation is a surprisingly difficult concept to understand. David Hume famously noted that causality is a concept that our science and philosophy cannot do without, but it is equally a concept that our science and philosophy cannot describe. Since Hume, the problem of cause has not gone away, and sometimes seems to get even worse (e.g., quantum mechanics confusing previous notions of causality). Yet, ways of doing science that lessen our need to explicitly use causality have become very effective (e.g., huge developments in statistics). Nevertheless, it still seems that the concept of causality is at the core of explaining how the world works, across fields as diverse as physics, medicine, logistics, the law, sociology, and history - and ordinary daily life - through all of which, explanations and predictions in terms of cause and effect remain intuitively central.

Causality remains a thorny problem but, in recent decades, significant progress has occurred, particularly in work by or inspired by Judea Pearl. This work incorporates many 20th century developments, including statistical methods - but with a reemphasis on finding the why, or the cause, behind statistical correlations -, progress in understanding the logic, semantics and metaphysics of conditionals and counterfactuals, developments based on insights from the likes of philosopher Hans Reichenbach or biological statistician Sewall Wright into causal precedence and path analysis, and much more. The result is a new toolkit to identify causes and build causal explanations. Yet even as we get better at identifying causes, this raises new (or old) questions about causality, including metaphysical questions about the nature of causes (and effects, events, objects, etc), but also questions about what we really use causality for (understanding the world as it is or just to glean predictive control of specific outcomes), about how causality is used differently in different fields and activities (is cause in physics the same as that in history?), and about how other crucial concepts relate to our concept of cause (space and time seem to be related to causality, but so do concepts of legal and moral responsibility).

This course will introduce students to the mathematical formalism derived from Pearl's work, based on directed acyclic graphs and probability theory. Building upon previous work by Reichenbach and Wright, Pearl defines a "a calculus of interventions" of "do-calculus" for talking about interventions and their relation to causation and counterfactuals. This model has been applied in various areas ranging from econometrics to statistics, where acquiring knowledge about causality is of great importance.

At the same time, the course will not forget some of the metaphysical and epistemological issues around cause, so that students can better critically evaluate putative causal explanations in their full context. Abstractly, such issues involve some of the same philosophical questions Hume already asked, but more practically, it is important to see how metaphysical and epistemological debates surrounding the notion of cause affect scientific practice, and equally if not more importantly, how scientific practice pushes the limits of theory. This course will look at various ways in which empirical data can be transformed into explanations and theories, including the variance approach to causality (characteristic of the positivistic quantitative paradigm), and the process theory of causality (associated with qualitative methodology). Examples and case studies will be relevant for students of the social sciences but also students of the natural/physical world as well.

Recommended Knowledge

Basic probability theory

Intended Learning Outcomes

No	Competence	ILO
1	Have	Have a clear understanding of the history of causal thinking.
2	Form	Form a critical understanding of the key debates and controversies surrounding the idea of causality.
3	Recognize	Recognize and apply probabilistic causal models.
4	Explain	Explain how understanding of causality differs among different disciplines.
5	Demonstrate	Demonstrate how theoretical thinking about causality has shaped scientific practices.

Indicative Literature

- Paul, L. A. and Ned Hall. Causation: A User’s Guide. Oxford University Press 2013.

- Pearl, Judea. Causality: Models, Reasoning and Inference. Cambridge University Press 2009.
- Pearl, Judea, Glymour Madelyn and Jewell, Nicolas. Causal Inference in Statistics: A Primer. Wiley 2016.
- llari, Phyllis McKay and Federica Russo. Causality: Philosophical Theory Meets Scientific Practice. Oxford University Press 2014.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Causation and Correlation (perspective II)	Written Examination	60 minutes	100	45%	1-5

Module Achievements: None

8.2.5 Linear Model and Matrices

Module Name	**Linear Model and Matrices**
Module Code	2025-CTNS-NSK-05
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 5
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Marc-Thorsten Hütt

Forms of Learning and Teaching	
Independent Study	90
Online Lecture	35
Workload Hours	125 hours

Module Components	**Number**	**Type**	**CP**
Linear model and matrices	CTNS-05	Seminar (Online)	5

Module Description

There are no universal 'right skills'. But the notion of linear models and the avenue to matrices and their properties can be useful in diverse disciplines to implement a quantitative, computational approach. Some of the most popular data and systems analysis strategies are built upon this framework. Examples include principal component analysis (PCA), the optimization techniques used in Operations Research (OR), the assessment of stable and unstable states in nonlinear dynamical systems, as well as aspects of machine learning.

Here we introduce the toolbox of linear models and matrix-based methods embedded in a wide range of transdisciplinary applications (part 1). We describe its foundation in linear algebra (part 2) and the range of tools and methods derived from this conceptual framework (part 3). At the end of the course, we outline applications to graph theory and machine learning (part 4). Matrices can be useful representations of networks and of system of linear equations. They are also the core object of linear stability analysis, an approach used in nonlinear dynamics. Throughout the course, examples from neuroscience, social sciences, medicine, biology, physics, chemistry, and other fields are used to illustrate these methods.

A strong emphasis of the course is on the sensible usage of linear approaches in a nonlinear world. We will critically reflect the advantages as well as the disadvantages and limitations of this method. Guiding questions are: How appropriate is a linear approximation of a nonlinear system? What do you really learn from PCA? How reliable are the optimal states obtained via linear programming (LP) techniques?

This debate is embedded in a broader context: How does the choice of a mathematical technique confine your view on the system at hand? How, on the other hand, does it increase your capabilities of analyzing the system (due to software available for this technique, the ability to compare with findings from other fields built upon the same technique and the volume of knowledge about this technique)?

In the end, students will have a clearer understanding of linear models and matrix approaches in their own discipline, but they will also see the full transdisciplinarity of this topic. They will make better decisions in their choice of data analysis methods and become mindful of the challenges when going from linear to nonlinear thinking.

Intended Learning Outcomes

No	Competence	ILO
1	Apply	Apply the concept of linear modeling in their own discipline.
2	Distinguish	Distinguish between linear and nonlinear interpretation strategies and understand the range of applicability of linear models.
3	Make	Make use of data analysis / data interpretation strategies from other disciplines, which are derived from linear algebra.
4	Be	Be aware of the ties that linear models have to machine learning and network theory,
5	Note	Note that these four ILOs can be loosely associated with the four parts of the course indicated above.

Indicative Literature

- Part 1: material from Linear Algebra for Everyone, Gilbert Strang, Wellesley-Cambridge Press, 2020.
- Part 2: material from Introduction to Linear Algebra (5th Edition), Gilbert Strang, Cambridge University Press, 2021.
- Part 3: Mainzer, Klaus. "Introduction: from linear to nonlinear thinking." Thinking in Complexity: The Computational Dynamics of Matter, Mind and Mankind (2007): 1-16.; material from Mathematics of Big Data: Spreadsheets, Databases, Matrices, and Graphs, Jeremy Kepner, Hayden Jananthan, The MIT Press, 2018.; material from Introduction to Linear Algebra (5th Edition), Gilbert Strang, Cambridge University Press, 2021.
- Part 4: material from Linear Algebra and Learning from Data, Gilbert Strang, Wellesley-Cambridge Press, 2019.

Entry Requirements

Prerequisites	Logic (perspective I) Causation and Correlation (perspective I) Causation and Correlation (perspective II) Logic (perspective II)
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Linear model and matrices	Written Examination	120 minutes	100	45%	1-4

Module Achievements: None

8.2.6 Complex Problem Solving

Module Name	Complex Problem Solving
Module Code	2025-CTNS-NSK-06
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 5
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Marco Verweij

Forms of Learning and Teaching	
Independent Study	90
Online Lecture	35
Workload Hours	125 hours

Module Components	Number	Type	CP
Complex Problem Solving	CTNS-06	Lecture (Online)	5

Module Description

Complex problems are, by definition, non-linear and/or emergent. Some fifty years ago, scholars such as Herbert Simon began to argue that societies around the world had developed an impressive array of tools with which to solve simple and even complicated problems, but still needed to develop methods with which to address the rapidly increasing number of complex issues. Since then, a variety of such methods has emerged. These include 'serious games' developed in computer science, 'multisector systems analysis' applied in civil and environmental engineering, 'robust decision-making' proposed by the RAND Corporation, 'design thinking' developed in engineering and business studies, 'structured problem-solving' used by McKinsey & Co., 'real-time technology assessment' advocated in science and technology studies, and 'deliberative decision-making' emanating from political science.

In this course, students first learn to distinguish between simple, complicated and complex problems. They also become familiar with the ways in which a particular issue can sometimes shift from one category into another. In addition, the participants learn to apply several tools for resolving complex problems. Finally, the students are introduced to the various ways in which natural and social scientists can help stakeholders resolve complex problems. Throughout the course examples and applications will be used. When possible, guest lectures will be offered by experts on a particular tool for tackling complex issues. For the written, take-home exam, students will have to select a specific complex problem, analyse it and come up with a recommendation - in addition to answering several questions about the material learned.

Recommended Knowledge

- Being able to read primary academic literature

- Willingness to engage in teamwork

- Camillus, J. (2008). Strategy as a wicked problem. Harvard Business Review 86: 99-106;

- Rogers, P. J. (2008). Using programme theory to evaluate complicated and complex aspects of interventions. Evaluation, 14, 29–48.

Intended Learning Outcomes

No	Competence	ILO
1	Identify	Identify a complex problem.
2	Develop	Develop an acceptable recommendation for resolving complex problems.
3	Understand	Understand the roles that natural and social scientists can play in helping stakeholders resolve complex problems.

Indicative Literature

- Camillus, J. (2008). Strategy as a wicked problem. Harvard Business Review 86: 99-106; Rogers, P. J. (2008). Using programme theory to evaluate complicated and complex aspects of interventions. Evaluation, 14, 29–48.
- Chia, A. (2019). Distilling the essence of the McKinsey way: The problem-solving cycle. Management Teaching Review 4(4): 350-377.
- Den Haan, J., van der Voort, M.C., Baart, F., Berends, K.D., van den Berg, M.C., Straatsma, M.W., Geenen, A.J.P., & Hulscher, S.J.M.H. (2020). The virtual river game: Gaming using models to collaboratively explore river management complexity, Environmental Modelling & Software 134, 104855.
- Folke, C., Carpenter, S., Elmqvist, T., Gunderson, L., Holling, C.S., & Walker, B. (2002). Resilience and sustainable development: Building adaptive capacity in a world of transformations. AMBIO: A Journal of the Human Environment 31(5): 437-440.
- Ostrom, E. (2010). Beyond markets and states: Polycentric governance of complex economic systems. American Economic Review 100(3): 641-72.
- Pielke, R. Jr. (2007). The honest broker: Making sense of science in policy and politics. Cambridge: Cambridge University Press.
- Project Management Institute (2021). A guide to the project management body of knowledge (PMBOK® guide).
- Schon, D. A., & Rein, M. (1994). Frame reflection: Toward the resolution of intractable policy controversies. New York: Basic Books.
- Simon, H. A. (1973). The structure of ill structured problems. Artificial Intelligence 4(3-4): 181-201.
- Verweij, M. & Thompson, M. (Eds.) (2006). Clumsy solutions for a complex world. London: Palgrave Macmillan.

Entry Requirements

Prerequisites	Logic (perspective I) Causation and Correlation (perspective I) Causation and Correlation (perspective II) Logic (perspective II)

Co-requisites	None
Additional Remarks	

__Assessment and Completion__

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Complex Problem Solving	Written Examination	120 minutes	100	45%	1-3

Module Achievements: None

8.2.7 Argumentation, Data Visualization and Communication (perspective I)

Module Name	Argumentation, Data Visualization and Communication (perspective I)
Module Code	2025-CTNS-NSK-07
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 5
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Arvid Kappas Prof. Dr. Jules Coleman

Forms of Learning and Teaching	
Independent Study	90
Online Lecture	35
Workload Hours	125 hours

Module Components	Number	Type	CP
Argumentation, Data Visualization and Communication (perspective I)	CTNS-07	Lecture (Online)	5

Module Description

One must be careful not to confuse argumentation with being argumentative. The latter is an unattractive personal attribute, whereas the former is a requirement of publicly holding a belief, asserting the truth of a proposition, the plausibility of a hypothesis, or a judgment of the value of a person or an asset. It is an essential component of public discourse. Public discourse is governed by norms and one of those norms is that those who assert the truth of a proposition or the validity of an argument or the responsibility of another for wrongdoing open themselves up to good faith requests to defend their claims. In its most general meaning, argumentation is the requirement that one offer evidence in support of the claims they make, as well as in defense of the judgments and assessments they reach. There are different modalities of argumentation associated with different contexts and disciplines. Legal arguments have a structure of their own as do assessments of medical conditions and moral character. In each case, there are differences in the kind of evidence that is thought relevant and, more importantly, in the standards of assessment for whether a case has been successfully made. Different modalities of argumentation require can call for different modes of reasoning. We not only offer reasons in defense of or in support of beliefs we have, judgments we make and hypotheses we offer, but we reason from evidence we collect to conclusions that are warranted by them.

Reasoning can be informal and sometimes even appear unstructured. When we recognize some reasoning as unstructured yet appropriate what we usually have in mind is that it is not linear. Most reasoning we are familiar with is linear in character. From A we infer B, and from A and B we infer C, which all together support our commitment to D. The same form of reasoning applies whether the evidence for A, B or C is direct or circumstantial. What changes in these cases is perhaps the weight we give to the evidence and thus the confidence we have in drawing inferences from it.

Especially in cases where reasoning can be supported by quantitative data, wherever quantitative data can be obtained either directly or by linear or nonlinear models, the visualization of the corresponding data can become key in both, reasoning and argumentation. A graphical representation can reduce the complexity of argumentation and is considered a must in effective scientific communication. Consequently, the course will also focus on smart and compelling ways for data visualization - in ways that go beyond what is typically taught in statistics or mathematics lectures. These tools are constantly developing, as a reflection of new software and changes in state of the presentation art. Which graph or bar chart to use best for which data, the use of colors to underline messages and arguments, but also the pitfalls when presenting data in a poor or even misleading manner. This will also help in readily identifying intentional mis-representation of data by others, the simplest to recognize being truncating the ordinate of a graph in order to exaggerate trends. This frequently leads to false arguments, which can then be readily countered.

There are other modalities of reasoning that are not linear however. Instead they are coherentist. We argue for the plausibility of a claim sometimes by showing that it fits in with a set of other claims for which we have independent support. The fit is itself the reason that is supposed to provide confidence or grounds for believing the contested claim.

Other times, the nature of reasoning involves establishing not just the fit but the mutual support individual items in the evidentiary set provide for one another. This is the familiar idea of a web of interconnected, mutually supportive beliefs. In some cases, the support is in all instances strong; in others it is uniformly weak, but the set is very large; in other cases, the support provided each bit of evidence for the other is mixed: sometimes strong, sometimes weak, and so on.

There are three fundamental ideas that we want to extract from this segment of the course. These are (1) that argumentation is itself a requirement of being a researcher who claims to have made findings of one sort or another; (2) that there are different forms of appropriate argumentation for different domains and circumstances; and (3) that there are different forms of reasoning on behalf of various claims or from various bits of evidence to conclusions: whether those conclusions are value judgments, political beliefs, or scientific conclusions. Our goal is to familiarize you with all three of these deep ideas and to help you gain facility with each.

Intended Learning Outcomes

No	Competence	ILO
1	Distinguish	Distinguish among different modalities of argument, e.g. legal arguments, vs. scientific ones.
2	Construct	Construct arguments using tools of data visualization.
3	Communicate	Communicate conclusions and arguments concisely, clearly and convincingly.

Indicative Literature

- Tufte, E.R. (1985). The visual display of quantitative information. The Journal for Healthcare Quality (JHQ), 7(3), 15.

- Cairo, A (2012). The Functional Art: An introduction to information graphics and visualization. New Ridders.
- Knaflic, C.N. (2015). Storytelling with data: A data visualization guide for business professionals. John Wiley & Sons.

Entry Requirements

Prerequisites	Logic (perspective I) Causation and Correlation (perspective I) Causation and Correlation (perspective II) Logic (perspective II)
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Argumentation, Data Visualization and Communication (perspective I)	Written Examination	120 minutes	100	45%	1-3

Module Achievements: None

8.2.8 Argumentation, Data Visualization and Communication (perspective II)

Module Name	Argumentation, Data Visualization and Communication (perspective II)
Module Code	2025-CTNS-NSK-08
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Arvid Kappas Prof. Dr. Jules Coleman

Forms of Learning and Teaching	
Independent Study	80
Online Lecture	35
Tutorial	10
Workload Hours	125 hours

Module Components	Number	Type	CP
Argumentation, Data Visualization and Communication (perspective II)	CTNS-08	Lecture (Online)	5

Module Description

Humans are a social species, and interaction is crucial throughout the entire life span. While much of human communication involves language, there is a complex multichannel system of nonverbal communication that enriches linguistic content, provides context, and is also involved in structuring dynamic interaction. Interactants achieve goals by encoding information that is interpreted in the light of current context in transactions with others. This complexity implies also that there are frequent misunderstandings as a sender's intention is not fulfilled. Students in this course will learn to understand the structure of communication processes in a variety of formal and informal contexts. They will learn what constitutes challenges to achieving successful communication and to how to communicate effectively, taking the context and specific requirements for a target audience into consideration. These aspects will be discussed also in the scientific context, as well as business, and special cases, such as legal context - particularly with view to argumentation theory.

Communication is a truly transdisciplinary concept that involves knowledge from diverse fields such as biology, psychology, neuroscience, linguistics, sociology, philosophy, communication and information science. Students will learn what these different disciplines contribute to an understanding of communication and how theories from these fields can be applied in the real world. In the context of scientific communication, there will also be a focus on visual communication of data in different disciplines. Good practice examples will be contrasted with typical errors to facilitate successful communication also with view to the Bachelor's thesis.

Recommended Knowledge

- Ability and openness to engage in interactions

- Media literacy, critical thinking and a proficient handling of data sources

- Own research in academic literature

Intended Learning Outcomes

No	Competence	ILO
1	Analyze	Analyze communication processes in formal and informal contexts.
2	Identify	Identify challenges and failures in communication.
3	Design	Design communications to achieve specified goals to specific target groups.
4	Understand	Understand the principles of argumentation theory.
5	Use	Use data visualization in scientific communications.

Indicative Literature

- Joseph A. DeVito: The Interpersonal Communication Book (Global edition, 16th edition), 2022.
- Steven L. Franconeri, Lace M. Padilla, Priti Shah, Jeffrey M. Zacks, and Jessica Hullman: The Science of Visual Data Communication: What Works Psychological Science in the Public Interest, 22(3), 110–161, 2022.
- Douglas Walton: Argumentation Theory – A Very Short Introduction. In: Simari, G., Rahwan, I. (eds) Argumentation in Artificial Intelligence. Springer, Boston, MA, 2009.

Entry Requirements

Prerequisites	Logic (perspective I) Logic (perspective II) Causation and Correlation (perspective I) Causation and Correlation (perspective II)
Co-requisites	None
Additional Remarks	

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Argumentation, Data Visualization and Communication (perspective II)	Presentation	Digital submission (Asynchronous)	100	45%	1-5

Module Achievements: Asynchronous presentation on a topic relating to the major of the student, including a reflection including concept outlining the rationale for how arguments are selected and presented based on a particular target group for a particular purpose. The presentation shall be multimedial and include the presentation of data. The module achievement ensures sufficient knowledge about key concepts of effective communication including a reflection on the presentation itself.

8.2.9 Agency, Leadership, and Accountability

Module Name	Agency, Leadership, and Accountability
Module Code	2025-CTNS-NSK-09
Module ECTS	5
Study Semester	Mandatory status for: - 2025-S-ACS-BSc 5 Mandatory Elective status for: - 2025-CS-BSc 6
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Prof. Dr. Jules Coleman

Forms of Learning and Teaching	
Independent Study	90
Online Lecture	35
Workload Hours	125 hours

Module Components	Number	Type	CP
Agency, Leadership, and Accountability	CTNS-09	Lecture (Online)	5

Module Description

Each of us is judged by the actions we undertake and held to account for the consequences of them. Sometimes we may be lucky and our bad acts don't have harmful effects on others. Other times we may be unlucky and reasonable decisions can lead to unexpected or unforeseen adverse consequences for others. We are therefore held accountable both for choices and for outcomes. In either case, accountability expresses the judgment that we bear responsibility for what we do and what happens as a result. But our responsibility and our accountability in these cases is closely connected to the idea that we have agency.

Agency presumes that we are the source of the choices we make and the actions that result from those choices. For some, this may entail the idea that we have free will. But there is scientific world view that holds that all actions are determined by the causes that explain them, which is the idea that if we knew the causes of your decisions in advance, we would know the decision you would make even before you made it. If that is so, how can your choice be free? And if it is not free, how can you be responsible for it? And if you cannot be responsible, how can we justifiably hold you to account for it?

These questions express the centuries old questions about the relationship between free will and a determinist world view: for some, the conflict between a scientific world view and a moral world view.

But we do not always act as individuals. In society we organize ourselves into groups: e.g. tightly organized social groups, loosely organized market economies, political societies, companies, and more. These groups have structure. Some individuals are given the responsibility of leading the group and of exercising authority. But one can exercise authority over others in a group merely by giving orders and threatening punishment for non-compliance.

Exercising authority is not the same thing as being a leader? For one can lead by example or by encouraging others to exercise personal judgment and authority. What then is the essence of leadership?

The module has several educational goals. The first is for students to understand the difference between actions that we undertake for which we can reasonably held accountable and things that we do but which we are not responsible for. For example, a twitch is an example of the latter, but so too may be a car accident we cause as a result of a heart attack we had no way of anticipating or controlling. This suggests the importance of control to responsibility. At the heart of personal agency is the idea of control. The second goal is for students to understand what having control means. Some think that the scientific view is that the world is deterministic, and if it is then we cannot have any personal control over what happens, including what we do. Others think that the quantum scientific view entails a degree of indeterminacy and that free will and control are possible, but only in the sense of being unpredictable or random. But then random outcomes are not ones we control either. So, we will devote most attention to trying to understand the relationships between control, causation and predictability.

But we do not only exercise agency in isolation. Sometimes we act as part of groups and organizations. The law often recognizes ways in which groups and organizations can have rights, but is there a way in which we can understand how groups have responsibility for outcomes that they should be accountable for. We need to figure out then whether there is a notion of group agency that does not simply boil down to the sum of individual actions. We will explore the ways in which individual actions lead to collective agency.

Finally we will explore the ways in which occupying a leadership role can make one accountable for the actions of others over which one has authority.

Intended Learning Outcomes

No	Competence	ILO
1	Understand	Understand and reflect how the social and moral world views that rely on agency and responsibility are compatible, if they are, with current scientific world views.
2	Understand	Understand how science is an economic sector, populated by large powerful organizations that set norms, fund research agendas.
3	Identify	Identify the difference between being a leader of others or of a group - whether a research group or a lab or a company - and being in charge of the group.
4	Learn	Learn to be a leader of others and groups. Understand that when one graduates one will enter not just a field of work but a heavily structured set of institutions and that one's agency and responsibility for what happens, what work gets done, its quality and value, will be affected accordingly.

Indicative Literature

- Hull, David L. "Science as a Process." Science as a Process. University of Chicago Press, 2010.
- Feinberg, Joel. "Doing & deserving; essays in the theory of responsibility." (1970).

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Agency, Leadership, and Accountability	Written Examination	120 minutes	100	45%	1-4

Module Achievements: None

8.2.10 Community Impact Project

Module Name	Community Impact Project
Module Code	2025-CTNS-CIP-10
Module ECTS	5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 6
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	CIP Faculty Coordinator

Forms of Learning and Teaching	
Introductory, Accompanying, and Final Events	10
Self-Organized Teamwork	115
Workload Hours	125 hours

Module Components	Number	Type	CP
Community Impact Project	CTNS-10	Project	5

Module Description

CIPs are self-organized, major-related, and problem-centered applications of students' acquired knowledge and skills. These activities will ideally be connected to their majors so that they will challenge the students' sense of practical relevance and social responsibility within the field of their studies. Projects will tackle real issues in their direct and/or broader social environment. These projects ideally connect the campus community to other communities, companies, or organizations in a mutually beneficial way.

Students are encouraged to create their own projects and find partners (e.g., companies, schools, NGOs), but will get help from the CIP faculty coordinator team and faculty mentors to do so. They can join and collaborate in interdisciplinary groups that attack a given issue from different disciplinary perspectives.

Student activities are self-organized but can draw on the support and guidance of both faculty and the CIP faculty coordinator team.

Usability and Relationship to other Modules

Students who have accomplished their CIP (6th semester) are encouraged to support their fellow students during the development phase of the next year's projects (4th semester).

Recommended Knowledge

- Basic knowledge of the main concepts and methodological instruments of the respective disciplines.

- Develop or join a community impact project before the 5th or 6th semester based on the introductory events during the 4th semester by using the database of projects, communicating with fellow students and faculty, and finding potential companies, organizations, or communities to target.

Intended Learning Outcomes

No	Competence	ILO
1	The	The Community Impact Project is designed to convey the required personal and social competencies for enabling students to finish their studies at Constructor University as socially conscious and responsible graduates (part of the Constructor University's mission) and to convey social and personal abilities to the students, including a practical awareness of the societal context and relevance of their academic discipline.
2	Understand	Understand the real-life issues of communities, organizations, and industries and relate them to concepts in their own discipline.
3	Enhance	Enhance problem-solving skills and develop critical faculty, create solutions to problems, and communicate these solutions appropriately to their audience.
4	Apply	Apply media and communication skills in diverse and non-peer social contexts.
5	Develop	Develop an awareness of the societal relevance of their own scientific actions and a sense of social responsibility for their social surroundings.
6	Reflect	Reflect on their own behavior critically in relation to social expectations and consequences.
7	Work	Work in a team and deal with diversity, develop cooperation and conflict skills, and strengthen their empathy and tolerance for ambiguity.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	At least 15 CP from CORE modules in the major.

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Community Impact Project	Project Assessment		100	Graded as pass/fail	All

Module Achievements: None

8.3 Language and Humanities Modules

8.3.1 Languages

The descriptions of the language modules are provided in a separate document, the “Language Module Handbook” that can be accessed from the Constructor University’s Language & Community Center internet sites (https://constructor.university/student-life/language-community-center/learning-languages).

8.3.2 Humanities

8.3.2.1 Introduction to the Philosophy of Science

Module Name	Introduction to the Philosophy of Science
Module Code	2025-CTHU-HUM-002
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 1 - 2025-CS-BSc 2
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Dr. Eoin Ryan

Forms of Learning and Teaching	
Independent Study	45
Online Lecture	17.5
Workload Hours	62.5 hours

Module Components	Number	Type	CP
Introduction to the Philosophy of Science	CTHU-002	Lecture (Online)	2.5

Module Description

This humanities module will introduce students to some of the central ideas in philosophy of science. Topics will include distinguishing science from pseudo-science, types of inference and the problem of induction, the pros and cons of realism and anti-realism, the role of explanation, the nature of scientific change, the difference between natural and social sciences, scientism and the values of science, as well as some examples from philosophy of the special sciences (e.g., physics, biology).

The course aims to give students an understanding of how science produces knowledge, and some of the various contexts and issues which mean this process is never entirely transparent, neutral, or unproblematic. Students will gain a critical understanding of science as a human practice and technology; this will enable them both to better understand the importance and success of science, but also how to properly critique science when appropriate.

Intended Learning Outcomes

No	Competence	ILO
1	Understand	Understand key ideas from the philosophy of science.
2	Discuss	Discuss different types of inference and rational processes.

3	Describe	Describe differences between how the natural sciences, social sciences and humanities discover knowledge.
4	Identify	Identify ways in which science can be more and less value-laden.
5	Illustrate	Illustrate some important conceptual leaps in the history of science.

Indicative Literature

- Peter Godfrey-Smith Theory and Reality (2021)
- James Ladyman, Understanding Philosophy of Science (2002).
- Paul Song, Philosophy of Science: Perspectives from Scientists (2022).

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Introduction to the Philosophy of Science	Written Examination	60 minutes	100	45%	1-5

Module Achievements: None

8.3.2.2 Introduction to Philosophical Ethics

Module Name	**Introduction to Philosophical Ethics**
Module Code	2025-CTHU-HUM-001
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 1 - 2025-CS-BSc 2
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Dr. Eoin Ryan

Forms of Learning and Teaching	
Independent Study	45
Online Lecture	17.5
Workload Hours	62.5 hours

Module Components	**Number**	**Type**	**CP**
Introduction to Philosophical Ethics	CTHU-001	Lecture (Online)	2.5

Module Description

The nature of morality - how to lead a life that is good for yourself, and how to be good towards others - has been a central debate in philosophy since the time of Socrates, and it is a topic that continues to be vigorously discussed. This course will introduce students to some of the key aspects of philosophical ethics, including leading normative theories of ethics (e.g. consequentialism or utilitarianism, deontology, virtue ethics, natural law ethics, egoism) as well as some important questions from metaethics (are useful and generalizable ethical claims even possible; what do ethical speech and ethical judgements actually do or explain) and moral psychology (how do abstract ethical principles do when realized by human psychologies). The course will describe ideas that are key factors in ethics (free will, happiness, responsibility, good, evil, religion, rights) and indicate various routes to progress in understanding ethics, as well as some of their difficulties.

Intended Learning Outcomes

No	**Competence**	**ILO**
1	Describe	Describe normative ethical theories such as consequentialism, deontology and virtue ethics.
2	Discuss	Discuss some metaethical concerns.
3	Analyze	Analyze ethical language.
4	Highlight	Highlight complexities and contradictions in typical ethical commitments.

5	Indicate	Indicate common parameters for ethical discussions at individual and social levels.
6	Analyze	Analyze notions such as objectivity, subjectivity, universality, pluralism, value.

Indicative Literature

- Simon Blackburn Being Good (2009).
- Russ Shafer-Landay A Concise Introduction to Ethics (2019).
- Mark van Roojen Metaethicas: A Contemporary Introduction (2015).

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	Examination Type	Duration /Length	Weight (%)	Minimum	ILOs
Introduction to Philosophical Ethics	Written Examination	60 minutes	100	45%	1-6

Module Achievements: None

8.3.2.3 Introduction to Visual Culture

Module Name	Introduction to Visual Culture
Module Code	2025-CTHU-HUM-003
Module ECTS	2.5
Study Semester	Mandatory status for: None Mandatory Elective status for: - 2025-CS-BSc 1 - 2025-CS-BSc 2
Duration	1 Semester
Program Affiliation	2025-CT ()
Module Coordinator(s)	Dr. Irina Chiaburu

Forms of Learning and Teaching	
Independent Study	45
Online Lecture	17.5
Workload Hours	62.5 hours

Module Components	Number	Type	CP
Introduction to Visual Culture	CTHU-003	Lecture (Online)	2.5

Module Description

Of the five senses, the sense of sight has for a long time occupied the central position in human cultures. As John Berger has suggested this could be because we can see and recognize the world around us before we learn how to speak. Images have been with us since the earliest days of the human history. In fact, the earliest records of human history are images found on cave walls across the world. We use images to capture abstract ideas, to catalogue and organize the world, to represent the world, to capture specific moments, to trace time and change, to tell stories, to express feelings, to better understand, to provide evidence and more. At the same time, images exert their power on us, seducing us into believing in their 'innocence', that is into forgetting that as representations they are also interpretations, i.e., a particular version of the world.

The purpose of this course is to explore multiple ways in which images and the visual in general mediate and structure human experiences and practices from more specialized discourses, e.g., scientific discourses, to more informal and personal day-to-day practices, such as self-fashioning in cyberspace. We will look at how social and historical contexts affect how we see, as well as what is visible and what is not. We will explore the centrality of the visual to the intellectual activity, from early genres of scientific drawing to visualizations of big data. We will examine whether one can speak of visual culture of protest, look at the relationship between looking and subjectivity and, most importantly, ponder the relationship between the visual and the real.

Intended Learning Outcomes

No	Competence	ILO

1	Understand	Understand a range of key concepts pertaining to visual culture, art theory and cultural analysis.
2	Understand	Understand the role visuality plays in development and maintenance of political, social, and intellectual discourses.
3	Think	Think critically about images and their contexts.
4	Reflect	Reflect critically on the connection between seeing and knowing.

Indicative Literature

- Berger, J., Blomberg, S., Fox, C., Dibb, M., & Hollis, R. (1973). Ways of seeing.
- Foucault, M. (2002). The order of things: an archaeology of the human sciences (Ser. Routledge classics). Routledge.
- Hunt, L. (2004). Politics, culture, and class in the French revolution: twentieth anniversary edition, with a new preface (Ser. Studies on the history of society and culture, 1). University of California Press.
- Miller, V. (2020). Understanding digital culture (Second). SAGE.
- Thomas, N. (1994). Colonialism's culture: anthropology, travel and government. Polity Press.

Entry Requirements

Prerequisites	None
Co-requisites	None
Additional Remarks	None

Assessment and Completion

Components	**Examination Type**	**Duration /Length**	**Weight (%)**	**Minimum**	**ILOs**
Introduction to Visual Culture	Written Examination	60 minutes	100	45%	1-4

Module Achievements: None

9 Appendix

9.1 Intended Learning Outcomes Assessment-Matrix

Computer Science (BSc.)					Mathematical Foundations of Computer Science	Digital Systems and Computer Architecture	Programming in C and C++	Algorithms and Data Structures	Development in JVM Languages	Databases	Software Engineering	Operating Systems	Automata, Computability, and Complexity	Functional Programming	Legal and Ethical Aspects of Computer Science	Machine Learnig	Academic Skills in Computer Science	Computer Graphics	Image Processing	Distributed Algorithms	Web Application Development	Computer Networks	Secure and Dependable Systems	Bachelor Thesis	Elements of Linear Algebra	Elements of Calculus	Probability and Random Processes	Numerical Methods/Statistics and Data Analytics	Internship	CT New Skills	CT German language and Humanities
Semester					1	2	1	2	2	3	4	3	4	3	3	4	4	5	6	6	6	5	6	6	1	2	3	4	5	3-6	1-2
Mandatory/mandatory elective					m	m	m	m	me	m	m	m	m	me	me	me	me	me	me	me	me	me	me	m	me	me	m	me	m	m	me
Credits					7.5	7.5	7.5	7.5	7.5	7.5	7.5	7.5	7.5	5	2.5	5	2.5	5	5	5	5	5	5	15	5	5	5	5	15	20	5
	Competencies*																														
Program Learning Outcomes	**A**	**E**	**P**	**S**																											
Work professionally in the highly dynamic computer science field and enter graduate programs related to computer science.	x	x	x		x	x	x	x		x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
Apply fundamental concepts of computer science while solving problems.	x	x			x	x	x	x		x	x	x	x	x		x		x	x	x	x	x	x	x					x	x	
Think in an analytic way at multiple levels of abstraction.	x	x	x	x	x	x	x	x		x	x	x	x	x		x	x	x	x	x	x	x	x	x	x	x	x	x		x	
Develop, analyze and implement algorithms using modern software engineering methods.	x	x			x		x	x		x	x					x		x	x		x		x	x							
Understand the characteristics of a range of computing platforms and their advantages and limitations.	x	x				x						x		x		x		x	x			x		x							
Choose from multiple programming paradigms, languages and algorithms in order to solve a given problem adequately.	x	x				x	x	x		x	x	x		x		x		x	x	x	x	x	x	x							
Describe the fundamental theory of computation and computability.	x				x			x					x																		
Apply the necessary mathematical methods.	x				x																				x	x	x	x			
Recognize the context in which computer systems operate, including interactions with people and the physical world.	x	x	x	x		x						x			x	x	x	x	x	x	x		x	x					x	x	
Describe the state of published knowledge in their field or a specialization within it.	x	x	x							x	x	x	x	x		x	x	x	x	x	x	x	x	x							
Analyze and model real-life scenarios in organizations and industries using contemporary techniques of computer science, also taking methods and insights of other disciplines into account.	x	x	x							x	x	x		x		x		x	x	x	x	x	x	x					x	x	
Appropriately communicate solutions of problems in computer science in both spoken and written form to specialists and non-specialists.	x	x	x	x							x						x							x					x	x	
Draw scientifically-founded conclusions that consider social, professional, scientific and ethical aspects.	x	x	x	x						x	x	x		x	x	x	x	x	x	x	x	x	x	x					x	x	
Work effectively in a diverse team and take responsibility in a team.	x	x	x	x							x																		x	x	x
Take responsibility for their own learnig, personal and professional development and role in society, reflecting on their practice and evaluating critical feedback.															x		x							x					x	x	
Adhere to and defend ethical, scientific and professional standards.	x	x	x	x							x				x	x	x						x						x	x	
Assessment Type																															
Written examination					x	x	x	x	x	x	x	x	x	x		x		x	x	x	x	x	x		x	x	x	x		x	x
Term paper																															
Essay																															x
Project report																													x		
Poster presentation															x																
Laboratory report																															
Program Code							x		x					x																	
Oral examination																															x
Presentation																								x						x	x
Practical Assessments																															
Project Assessment										x	x						x				x									x	
Portfolio Assessments																															
Bachelor Thesis																								x							
Module achievements					x	x						x																			x

*Competencies: A-scientific/academic proficiency; E-competence for qualified employment; P-development of personality; S-competence for engagement in society