CONSTRUCTOR UNIVERSITY

Study Program Handbook

Applied Computer Science

Bachelor of Science

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

The subject-specific examination regulations for Applied 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 5).

Upon graduation, students in this program will receive a Bachelor of Science (BSc) degree with a scope of 180 ECTS (for specifics see Chapter 3 of this handbook).

Version	Valid as of	Decision	Details
Fall 2024- V1.1		Apr 8, 2025	Editorial- corrected typo in name of module “Introduction to Cyber Physical Systems”.
Fall 2024 - V1	Sept. 1 2024		
		Feb 22, 2023	Academic Senate approval of study program name change from “Computer Science and Software Engineering” to “Applied Computer Science”
		Jan 25, 2023	Originally approved by Academic Senate

Contents

1 Program Overview

1.1 Concept

1.1.1 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 a 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 provide breadth while the university-wide general foundation and methods modules, optional German language and Humanity 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

Digitalization is a key driver of innovation and success across all industries. Applied Computer Science is obviously a key element in these processes. At the same time, there is a substantial change in the way daily work is organized and carried out. The share of home office and remote work increases, e.g., to collaborate with team members who are distributed around the world or to control, monitor, and maintain facilities and processes from a distance. While offering a lot of opportunities in terms of

convenience for employees and reduced costs for employers, this new normal of working also requires different skills and knowledge of the related tools and methods, which are addressed by this program.

Furthermore, online education is changing the higher education landscape in profound ways. It caters for specific needs and interests of students, especially in terms of the flexibility in which they can carry out their studies. And it is a natural option to prepare them for the new normal of remote work.

The bachelor program in Applied Computer Science uses online education with high amounts of flipped-classroom elements. This means that students participate in online courses with predominantly asynchronous lectures and exercise material, which are complemented by tutorials and hands-on sessions. Students are guided and supported by faculty as well as experienced tutors and lecturers to transfer the acquired knowledge into practice. The hands-on elements include high amounts of collaboration with other students, use of tools and concepts to engage in distributed work from different places in potentially different time-zones, and remote access to physical devices and set-ups.

The Computer Science core of the program is complemented with Management and Leadership modules in the second and third study years. Students will not only be trained in programming and software development, but will also acquire fundamental knowledge in business and learn how innovations can be transferred into a marketable product. Furthermore, they may take part in interdisciplinary courses in which problems are tackled from a wider perspective challenging them to think outside the boundaries of their discipline.

Overall, by completing their studies, students will be able to directly enter the job market or to continue their studies in a graduate program, for example the MSc in Computer Science and Software Engineering offered at Constructor University. Apart from the solid knowledge and skills obtained in Applied Computer Science, graduates are particularly well prepared for the demands of modern work, i.e. to work remotely and as part of a diverse team.

1.2 Specific Advantages of Applied Computer Science at Constructor University

The Applied Computer Science program at Constructor University aims to provide an application-oriented knowledge of Computer Science including a preparation for important aspects of modern professional life, namely remote work and life-long learning.

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 the challenges of digitalization and the related state of the art in research and development.

- Early involvement in software development project work is an essential aspect of the study program which further extends the already positively acknowledged educational approach in Computer Science at Constructor University.
- The Computer Science faculty's pedagogy, 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 is going to ensure that the program will be 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.
- Computer Science 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 participate 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. As the program features important elements remote collaborative software development, there is also the option for online students to participate in according activities if they are interested in them.

1.3 Program-specific Educational Aims

1.3.1 Qualification Aims

The program is an online program with optional blended elements, e.g., in summer. Lectures incorporate asynchronous material and primarily follow a flipped classroom model, i.e., including application components in the spirit of problem-based- as well as project-based-learning. Practical components, particularly labs, projects, and thesis are based on remote access, distributed development. Tutoring includes virtual study groups, peer evaluation and mentoring by faculty. Performance evaluation are conducted as online e-exams.

The remote work aspects include collaborative software development and remote access to physical devices for, e.g., control, monitoring and maintenance. Due to the aspects of independent, self-governed knowledge acquisition, the students are prepared for life-long learning, where additional knowledge and skills need to be acquired or updated in a regular fashion, especially in fast moving areas like Computer Science.

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

- Applied Computer Science competence
  Graduates are familiar with the foundations of Computer Science and they are able to design and develop software addressing a given application scenario. They are able to analyze and structure complex problems and they are able to address them using methods of Applied Computer Science. Graduates are able to construct and maintain complex computer systems using a structured, analytic, and creative approach. They are trained in developing software in collaborative teams in a remote fashion, i.e., independent of the location they live and work at.

- 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 remote 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 Applied 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.

- Management competence
  Graduates have obtained basic business and management knowledge supporting them to reflect their core discipline against the background of a corporate environment and to incorporate a business perspective into computer science and software development.

1.3.2 Intended Learning Outcomes

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

1. acquire Applied Computer Science knowledge in an independent, self-governed way;
2. work in teams distributed around the globe to analyze complex problems, to evaluate them, and to derive solutions;
3. comprehend the processes and tools of Software Engineering for collaborative, remote software and systems development;
4. program software in C/C++ and understand algorithms;
5. be able to use libraries and to generate software in core Computer Science areas;
6. apply suited mathematical methods;
7. understand operating systems, databases, and web services;
8. comprehend methods from Artificial Intelligence and Machine Learning;
9. understand the relation between software and its links to the physical world;
10. analyze data and to extract insights from it;
11. apply the acquired Software Engineering skills and Computer Science knowledge in collaborative, remote projects;
12. use academic or scientific methods as appropriate in the field of Applied Computer Science such as defining research questions, justifying methods, collecting, assessing and interpreting relevant information, and drawing scientifically-founded conclusions that consider social, scientific and ethical insights;
13. develop and advance solutions to problems and arguments in their subject area and defend these in discussions with specialists and non-specialists;

14. engage ethically with academic, professional and wider communities and to actively contribute to a sustainable future, reflecting and respecting different views;
15. take responsibility for their own learning, personal and professional development and role in society, evaluating critical feedback and self-analysis;
16. apply their knowledge and understanding to a professional context;
17. take on responsibility in a diverse team;
18. adhere to and defend ethical, scientific and professional standards.

1.4 Career Options and Support

Digitalization is affecting all areas of business, industry, daily life, and society. There is accordingly a very high demand for graduates with a background in Applied Computer Science in general. In addition, students have been trained to be able to work in a remote, collaborative fashion and being able to engage in life-long learning, i.e., to acquire or update knowledge and skills in the fast-moving areas of Computer Science in an independent and self-governed way. This offers not only increased flexibility for graduates to engage in professional opportunities worldwide, it is also a substantial benefit for potential employers as they may select from an increased pool of talented candidates, whom they do not need to relocate to work on their job.

The areas of employment are almost unlimited as digitalization is important in business, industry, daily life, and society. Within these areas, research & development or management tracks can be taken. The job market includes jobs such as software engineer, information systems manager, data analyst, computer systems engineer, application developer, IT consultant, remote maintenance manager, and system analyst.

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 global network which is useful 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 on the study program, please contact the Study Program Coordinator:

Prof. Dr. Andreas Birk

Professor of Electrical Engineering & Computer Science

Email: abirk@constructor.university

or visit our program website: https://constructor.university/programs/online-programs/applied-computer-science

For more information on Student Services, please visit:

https://constructor.university/student-life/student-services

2 The Curricular Structure

2.1 General

The curricular structure provides multiple elements for enhancing employability, interdisciplinarity, and internationality. Additionally, a mandatory internship (or work in a start-up) of at least two months after the second year of study gives students opportunities 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 a certain degree of flexibility regarding their individual study path and which ensures that they can 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/academic-policies).

2.2 The Curriculum

2.2.1 Year 1

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 take introductory modules for a total of 60 CP from the Year 1 area. 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.

Applied Computer Science students take the following mandatory (m) modules in the first semester (30 CP)

- CHOICE Module: Introduction to Computer Science (m, 7.5 CP)
- CHOICE Module: Programming in C/C++ (m, 7.5 CP)
- CHOICE Module: Introduction to Data Science (m, 7.5 CP)
- CHOICE Module: Distributed Development (m, 2.5 CP)
- Methods Module: Calculus and Elements of Linear Algebra I (m, 5 CP)

and the following modules in the second semester (30 CP):

- CHOICE Module: Algorithms and Data Structures (m, 7.5 CP)
- CHOICE Module: Introduction to Cyber Physical Systems (m, 7.5 CP)
- CHOICE Module: Software Design and Prototyping (m, 7.5 CP)
- CHOICE Module: Distributed Development (m, 2.5 CP)
- Methods Module: Calculus and Elements of Linear Algebra II (m, 5 CP)

The 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

Introduction to Computer Science module discusses abstract and concrete notions of computing machines and algorithms, and the representation of information. Students are also exposed to a pure functional programming language. The Software Design and Prototyping module deals with prototyping software, also known as mockup systems. It is complemented by the Distributed Development module that deals with practical aspects of remotely developing software in teams distributed at different physical locations. The module Introduction to Cyber Physical Systems deals with the relations and interfaces of software to computer hardware, embedded systems, sensors and actuators, and networking. Relevant mathematical content is covered in the Matrix Algebra and Advanced Calculus modules and in the Introduction to Data Science module.

2.2.2 Year 2

In their second year, students take a total of 50 CP from a selection of in-depth, discipline-specific modules. Building on the introductory Year 1 modules and applying the methods and skills students have already acquired so far, 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.

In Year 2, Applied Computer Science students acquire the following disciplinary and methods mandatory modules (50 CP in total):

- CORE Module: Databases and Web Services (m, 7.5 CP)
- CORE Module: Operating Systems (m, 7.5 CP)
- CORE Module: Data Analytics and Modeling (m, 7.5 CP)
- CORE Module: Software Engineering (m, 7.5 CP)
- CORE Module: Artificial Intelligence (m, 7.5 CP)
- CORE Module: Machine Learning (m, 7.5 CP)
- Methods Module: Probability and Random Processes (m, 5 CP)

In the second year, core areas of Computer Science with a high relevance to modern software development are covered in the modules Databases and Web Services, Operating Systems, Artificial Intelligence, and Machine Learning. Knowledge in Software Engineering is deepened in the according module. Relevant mathematical aspects are covered in the modules Probability and Random Processes and Data Analytics, where the latter – together with Artificial Intelligence and Machine Learning – also deepens the knowledge related to Data Science. Multiple modules include practical software development aspects, namely Software Engineering, Databases and Web Services, Artificial Intelligence, Machine Learning, Machine Learning Tools and Data Analytics and Modeling.

Additionally, the students will take two mandatory "new skills" modules from the university-wide CONSTRUCTOR Track which is dedicated to multidisciplinary content dedicated to methods as well as intellectual skills (5 CP in total) (please also see 2.2.3.4):

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

The remaining 5 CP must be chosen from the Management Elective area (mandatory elective, me), which includes Management oriented modules that provide basic business and management knowledge.

- Management Module: Digital Business Models and Functions (me, 5 CP)
- Management Module: Marketing and Methods (me, 5 CP)

An updated list of all modules in this Elective area will be available in the online course catalogue at the start of the second academic year.

2.2.3 Year 3

During their third year, students prepare for and make decisions about their career after graduation. To explore available choices fitting individual interests, and to gain professional experience, students take a mandatory summer internship (see 2.2.3.1). The third year of studies allows Applied Computer Science students to take ACS Specialization modules, two new skills modules and two further Management Elective modules (as described in Chapter 2.2.3.3). Finally, the 6th semester is dedicated to fostering the students' research experience by involving them in a Bachelor thesis project.

2.2.3.1 Internship/Startup 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 personal role in employment and society, and develop 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 their business plans.

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

2.2.3.2 ACS Specialization Modules

In the third year of their studies, students take 15 CP of advanced ACS 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 among which students can select freely during their fifth and sixth semester. The default module size is 5 CP, with smaller 2.5 CP modules being possible as justified exceptions.

Applied Computer Science students take at least 15 CP from the following abridged list of ACS Specialization Modules:

- ACS Specialization Module: Computer Graphics (me, 5 CP)
- ACS Specialization Module: Computer Networks (me, 5 CP)

- ACS Specialization Module: Web Application Development (me, 5 CP)
- ACS Specialization Module: Human Computer Interaction (me, 5 CP)

An updated list of all modules in the ACS Specialization area will be available in the online course catalogue at the start of the third academic year.

2.2.3.3 Management Modules

Students take 5 CP from the Management area to acquire valuable knowledge in the field of business and management. Modules in this area aim to bridge the gap from software development to marketable software products and to prepare students interested in a management-oriented career track. A broad spectrum of topics is tackled, such as product development, innovation, marketing, leadership, general business, and change management. An updated list of all modules in the Management area will be available in the online course catalogue at the start of the third academic year.

2.2.3.4 Collaborative Software Project

In the collaborative software project (m, 5 CP), the students deepen their knowledge and skills in one or multiple areas of the first and especially second year. They are exposed to state-of-the-art research with the goal to derive ideas and strategies to address application-oriented problems and to develop software for them. Students learn how to organize and execute an application-oriented research and development (R&D) project. Students are expected to organize themselves in group work under the guidance of the instructor.

2.2.3.5 New Skills

This part of the curriculum constitutes the intellectual and conceptual tool kit, and is designed to cultivate and nurture the capacity for a particular set of intellectual dispositions – curiosity, imagination, critical thought, transferability – as well as a range of individual and societal capacities – self-reflection, argumentation and communication – and to introduce 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 as mentioned in 2.2.2:

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

In the third year, students take two 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 modules are mandatory for all students in the third year:

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

2.2.3.6 Bachelor Thesis

This module is a mandatory graduation requirement for all undergraduate students. 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 their major-specific 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 Faculty Supervisor, the Bachelor Thesis can also have an interdisciplinary nature.

3 Applied Computer Science Undergraduate Program Regulations

3.1 Scope of these Regulations

The regulations in this handbook are valid for all students who entered the Applied Computer Science undergraduate program at Constructor University in Fall 2024. In case of a 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/academic-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). Constructor University Bremen reserves therefore the right to modify the regulations of the program handbook.

3.2 Degree

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

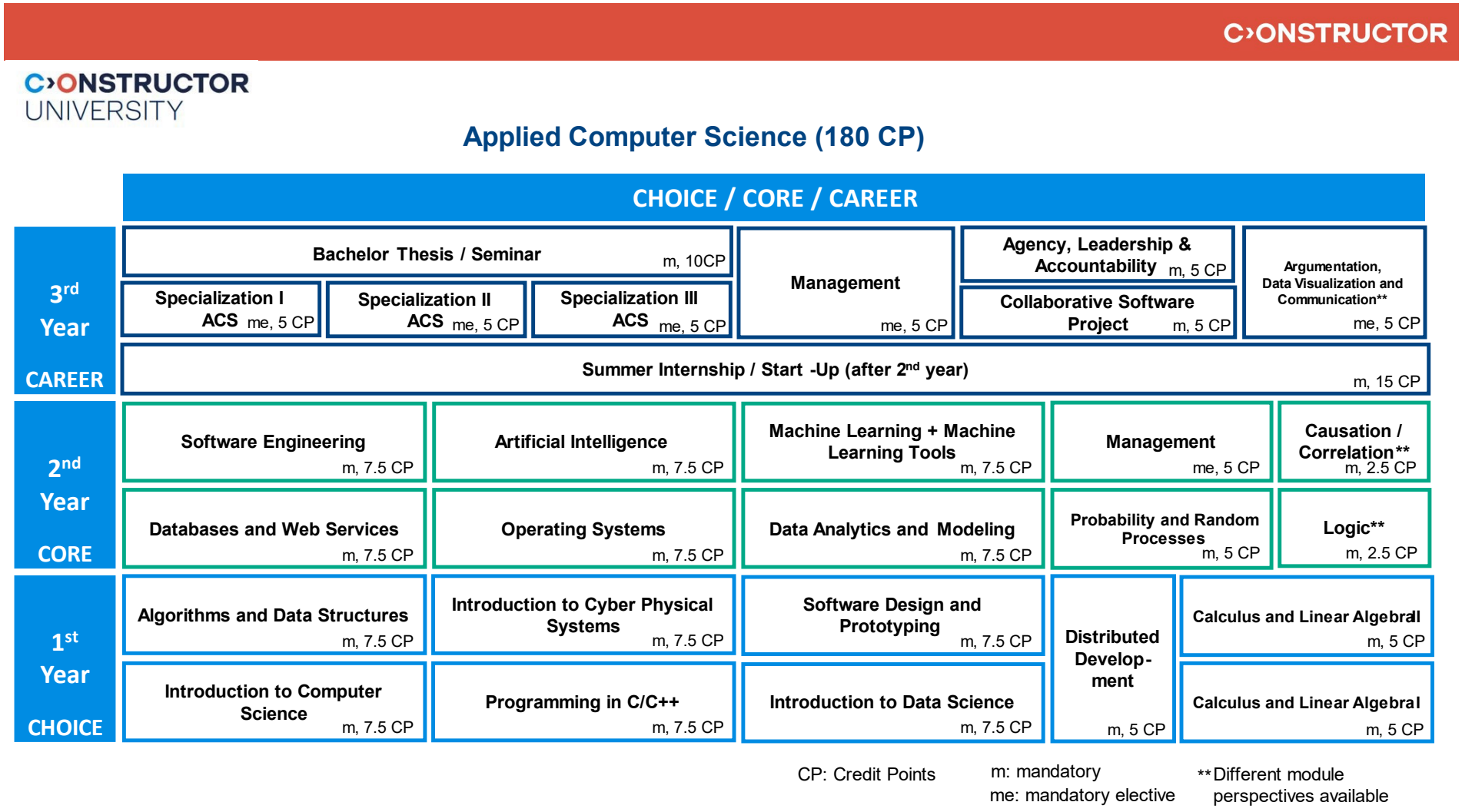
3.3 Graduation Requirements

In order 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 5 of this handbook.

4 Schematic Study Plan for Applied Computer Science

Figure 1 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.



5 Study and Examination Plan

Applied Computer Science (ACS) BSc

Matriculation Fall 2024

		Type	Assessment	Period	Status[1]	Sem.	CP
Year 1							**60**
Take all the mandatory YEAR 1 modules listed below, as this is a requirement for the Applied Computer Science program.							
	Unit: Program-Specific Modules						
ACS-101	**Module: Introduction to Computer Science**				**m**	**1**	**7.5**
ACS-101-A	Introduction to Computer Science	Lecture (online)	Written examination	Examination period			
ACS-102	**Module: Programming in C and C++**				**m**	**1**	**7.5**
ACS-102-A	Programming in C and C++	Lecture (online)	Written examination	Examination period			2.5
ACS-102-B	Programming in C and C++ Tutorial	Tutorial (online)	Practical assessment	During the semester			5
CH-700	**Module: Introduction to Data Science**				**m**	**1**	**7.5**
CH-700-A	Introduction to Data Science	Lecture (online)	Written examination	Examination period			
ACS-103	**Module: Algorithms and Data Structures**				**m**	**2**	**7.5**
ACS-103-A	Algorithms and Data Structures	Lecture (online)	Written examination	Examination period			
ACS-104	**Module: Introduction to Cyber Physical Systems**				**m**	**2**	**7.5**
ACS-104-A	Introduction to Cyber Physical Systems (CPS) Lecture	Lecture (online)	Written examination	Examination period			5
ACS-104-B	Introduction to Cyber Physical Systems (CPS) Tutorial	Tutorial (online)	Practical assignments	During the semester			2.5
ACS-105	**Module: Software Design and Prototyping**				**m**	**2**	**7.5**
ACS-105-A	Software Design and Prototyping Lecture	Lecture (online)	Written examination	Examination period			5
ACS-105-B	Software Design and Prototyping Tutorial	Tutorial (online)	Project Assignment	During the semester			2.5
ACS-106	**Module: Distributed Development**				**m**	**1/2**	**5**
ACS-106-A	Distributed Development I	Lecture & Lab (online)	Practical assessment	During the semester		1	2.5
ACS-106-B	Distributed Development II	Lecture & Lab (online)	Practical assessment	During the semester		2	2.5
	Unit: Methods						
CTMS-MAT-09	**Module: Calculus and Elements of Linear Algebra I**				**me**	**1**	**5**
CTMS-09	Calculus and Elements of Linear Algebra I	Lecture (online)	Written examination	Examination period			
CTMS-MAT-10	**Module: Calculus and Elements of Linear Algebra II**				**me**	**2**	**5**
CTMS-10	Calculus and Elements of Linear Algebra II	Lecture (online)	Written examination	Examination period			
Year 2							**60**
Take all the mandatory YEAR 2 modules listed below (50 CP), as this is a requirement for the Applied Computer Science program. Additionally, take the mandatory New Skills modules as listed below (2 x 2,5 CP). Further, please choose 5 CP of Management Electives.							
	Unit: Program-Specific Modules						
ACS-201	**Module: Databases and Web Services**				**m**	**3**	**7.5**
ACS-201-A	Databases and Web Services	Lecture (online)	Written examination	Examination period			5
ACS-201-B	Databases and Web Services - Project	Project (online)	Project Assessment	During the semester			2.5
ACS-202	**Module: Operating Systems**				**m**	**3**	**7.5**
ACS-202-A	Operating Systems	Lecture (online)	Written examination	Examination period			
CO-710	**Module: Data Analytics and Modeling**				**m**	**3**	**7.5**
CO-710-A	Data Analytics and Modeling	Lecture (online)	Written examination	Examination period			
ACS-203	**Module: Software Engineering**				**m**	**4**	**7.5**
ACS-203-A	Software Engineering	Lecture (online)	Written examination	Examination period			2.5
ACS-203-B	Software Engineering Project	Project (online)	Project Assessment	During the semester			5
ACS-204	**Module: Artificial Intelligence**				**m**	**4**	**7.5**
ACS-204-A	Artificial Intelligence	Lecture (online)	Written examination	Examination period			5
ACS-204-B	Artificial Intelligence Tutorial	Tutorial (online)	Project Assessment	During the semester			2.5
ACS-205	**Module: Machine Learning**				**m**	**4**	**7.5**
ACS-205-A	Machine Learning	Lecture (online)	Written examination	Examination period			5
ACS-205-B	Machine Learning Tools	Lab (online)	Practical Assignments	During the semester			2.5
	Unit: Methods						
CTMS-MAT-12	**Module: Probability and Random Processes**				**m**	**3**	**5**
CTMS-12	Probability and Random Processes	Lecture (online)	Written examination	Examination period			
	Unit: New Skills				**m**	**3/4**	**5**
CTNS-NSK	**Module: Logic[4]**				**m**	**3**	**2.5**
CTNS-01	Logic (perspective I)	Lecture (online)	Written examination	Examination period	me	3	
CTNS-02	Logic (perspective II)	Lecture (online)	Written examination	Examination period	me	3	
CTNS-NSK-03	**Module: Causation and Correlation[4] (perspective I)**				**m**	**4**	**2.5**
CTNS-03	Causation and Correlation (perspective I)	Lecture (online)	Written examination	Examination period	me	4	
CTNS-04	Causation and Correlation (perspective II)	Lecture (online)	Written examination	Examination period	me	4	
	Management Electives[3] *Take a total of 5 CP Management Electives*				**me**	**4**	**5**
CTMS-MET-20	**Module: Marketing & Methods**				**me**	**4**	**5**
CTMS-20	Marketing & Methods	Lecture (online)	Presentation	during the semester			5
MDSSB-DSOC-02	**Module: Digital Business Models and Functions**				**me**	**4**	**5**
MDSSB-DTRANS-02	Digital Business Models and Functions	Lecture (online)	Term Paper	during the semester			5

Year 3							**60**
Take all mandatory Year 3 modules listed below (30 CP). Further, select 15 CP of ACS Specialization Modules, 10 CP of Management Electives Modules and 10 CP mandatory New Skills modules							
CA-INT-900	**Module: Summer Internship**				**m**	**4/5**	**15**
CA-INT-900-0	Summer Internship		Report/Business plan	During the 5th semester			
ACS-400	**Module: Bachelor Thesis ACS**				**m**	**6**	**10**
ACS-400-T	Bachelor Thesis ACS	Thesis (online)	Thesis&Presentation	15th of May			
	ACS Specialization Modules[2] *Take a total of 15 CP ACS Specialization Modules*				**m**	**5**	**15**
ACS-303	**Module: Computer Graphics**				**me**	**5**	**5**
ACS-303-A	Computer Graphics	Lecture (online)	Written examination	Examination period			
ACS-304	**Module: Computer Networks**				**me**	**5**	**5**
ACS-304-A	Computer Networks	Lecture (online)	Written examination	Examination period			
ACS-305	**Module: Web Application Development**				**me**	**5**	**5**
ACS-305-A	Web Application Development	Lecture (online)	Written examination	Examination period			2.5
ACS-305-B	Web Application Development - Project	Project (online)	Project Assignment	during the sememster			2.5
ACS-306	**Module: Human Computer Interaction**				**me**	**5**	**5**
ACS-306-A	Human Computer Interaction	Lecture (online)	Written examination	Examination period			
	Unit: Collaborative Software Project				**m**	**5**	**5**
ACS-301	**Module: Collaborative Software Project**				**m**	**5**	**5**
ACS-301-A	Collaborative Software Project	Project (online)	Project report	During the semester			
	Management Electives[3] *Take a total of 5 CP Management elective modules.*				**me**	**5/6**	**5**
	Unit: New Skills				**m**	**5/6**	**5**
CTNS-NSK	**Module: Argumentation, Data Visualization and Communication**[4]				**m**	**5/6**	**5**
CTNS-07	Argumentation, Data Visualization and Communication (perspective I)	Lecture (online)	Written examination	Examination period	me	**5**	
CTNS-08	Argumentation, Data Visualization and Communication (perspective II)	Lecture (online)	Presentation	During the semester	me	**6**	
CTNS-NSK-09	**Module: Agency, Leadership and Accountability**				**m**	**6**	**5**
CTNS-09	Agency, Leadership and Accountability	Lecture (online)	Written examination	Examination period			
Total CP							**180**

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

[2] For a full listing of all ACS Specialization modules please consult the current online course catalogue and /or the study program handbooks.

[3] For a full listing of all Management modules please consult the current online course catalogue and /or the study program handbooks.

[4] Choose one of the perspectives

Figure 2: Study and Examination Plan

6 Applied Computer Science Modules

6.1 Introduction to Computer Science

Module Name			**Module Code**	**Level (type)**	**CP**
Introduction to Computer Science			ACS-101	Year 1 (CHOICE)	7.5
Module Components					
Number	Name			Type	CP
ACS-101-A	Introduction to Computer Science			Lecture (online)	7.5
Module Coordinator Prof. Dr. Jürgen Schönwälder	**Program Affiliation** • Computer Science (CS)			**Mandatory Status** Mandatory for ACS	
Entry Requirements Pre-requisites ☒ None	 Co-requisites ☒ None	 Knowledge, Abilities, or Skills	**Frequency** Every semester (Fall)	**Duration** 1 semester	

Student Workload				
Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
52.5 h	10 h	115 h	10 h	187.5 h

Recommendations for Preparation

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.

Content and Educational Aims

The module introduces fundamental concepts and techniques of computer science in a bottom-up manner. Based on clear mathematical foundations (which are developed as needed), the course discusses abstract and concrete notions of computing machines, information, and algorithms, focusing on the question of representation versus meaning in Computer Science.

The module introduces basic concepts of discrete mathematics with a focus on inductively defined structures, to develop a theoretical notion of computation. Students will learn the basics of the functional programming language Haskell because it treats computation as the evaluation of pure and typically inductively defined functions. The module covers a basic subset of Haskell that includes types, recursion, tuples, lists, strings, higher-order functions, and finally monads. Back on the theoretical side, the module covers the syntax and semantics of Boolean expressions and it explains how Boolean algebra relates to logic gates and digital circuits. On the technical side, the course introduces the representation of basic data types such as numbers, characters, and strings as well as the von Neuman computer architecture. On the algorithmic side, the course introduces the notion of correctness and elementary concepts of complexity theory (big O notation).

Intended Learning Outcomes

By the end of this module, students will be able to

1. explain basic concepts such as the correctness and complexity of algorithms (including the big O notation);
2. illustrate basic concepts of discrete math (sets, relations, functions);
3. recall basic proof techniques and use them to prove properties of algorithms;
4. explain the representation of numbers (integers, floats), characters and strings, and date and time;
5. summarize basic principles of Boolean algebra and Boolean logic;
6. describe how Boolean logic relates to logic gates and digital circuits;
7. outline the basic structure of a von Neumann computer;
8. explain the execution of machine instructions on a von Neumann computer;
9. describe the difference between assembler languages and higher-level programming languages;
10. define the differences between interpretation and compilation;
11. illustrate how an operating system kernel supports the execution of programs;
12. determine the correctness of simple programs;
13. write simple programs in a pure functional programming language.

Indicative Literature

Eric Lehmann, F. Thomson Leighton, Albert R. Meyer: Mathematics for Computer Science, online 2018.

David A. Patterson, John L Hennessy: Computer Organization and Design: The Hardware/Software Interface, 6th edition, Morgan Kaufmann, 2020.

Miran Lipovaca: Learn You a Haskell for Great Good!: A Beginner's Guide, 1st edition, No Starch Press, 2011.

Usability and Relationship to other Modules

- This module introduces key mathematical concepts and various notions of computing machines and computing abstractions and is in particularly important for subsequent courses covering theoretical aspects of computer science. This module is also important for courses that require a basic understanding of computer architecture and program execution at the hardware level.

Examination Type: Module Examination

Assessment Type: Written examination

Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of the module

Module achievement: 50% of the assignments correctly solved

This module introduces the functional programming language Haskell. Students develop their functional programming skills by solving programming problems. The module achievement ensures that a sufficient level of practical programming and problem-solving skills has been obtained.

Completion: To pass this module, the examination has to be passed with at least 45%.

6.2 Programming in C and C++

Module Name	**Module Code**	**Level (type)**	**CP**
Programming in C and C++	ACS-102	Year 1 (CHOICE)	7.5

Module Components

Number	Name	Type	CP
ACS-102-A	Programming in C and C++	Lecture (online)	2.5
ACS-102-B	Programming in C and C++ - Tutorial	Tutorial (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Dr. Kinga Lipskoch	• Computer Science (CS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Fall)	1 semester
☒ None	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
17.5 h	92.5 h	20 h	57.5 h	187.5 h

Recommendations for Preparation

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.

Content and Educational Aims

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.

Intended Learning Outcomes

By the end of this module, students will be able to

1. explain basic concepts of imperative programming languages such as variables, assignments, loops, and function calls;
2. write, test, and debug programs in the procedural programming language C using basic C library functions;
3. demonstrate how to use pointers to create dynamically allocated data structures such as linked lists;
4. explain the relationship between pointers and arrays;
5. illustrate basic object-oriented programming concepts such as objects, classes, information hiding, and inheritance;
6. give original examples of function and operator overloading and polymorphism;
7. write, test, and debug programs in the object-oriented programming language C++.

Indicative Literature

Brian Kernighan, Dennis Ritchie: The C Programming Language, 2nd edition, Prentice Hall 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.

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.

Examination Type: Module Component Examinations

Component 1: Lecture

Assessment types: Written examination — Duration: 120 min, Weight: 33%

Scope: All theoretical intended learning outcomes of the module

Component 2: Tutorial

Assessment: Practical assessment — Weight: 67%

Scope: All practical intended learning outcomes of the module

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.3 Introduction to Data Science

Module Name	**Module Code**	**Level (type)**	**CP**
Introduction to Data Science	CH-700	Year 1	7.5

Module Components

Number	Name	Type	CP
CH-700-A	Introduction to Data Science	Lecture (online)	7.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Hilke Brockmann, Dr. Georgi Dragolov	• Minor in Data Science	Mandatory for ACS, MDDA and Minor in Data Science

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Fall)	1 semester
☒ None	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
52.5 h	57.5 h	20 h	57.5 h	187.5 h

Recommendations for Preparation

None.

Content and Educational Aims

The module introduces data science with an integrated presentation of three essential components, namely, (1) societal/legal implications and business opportunities, (2) technical/theoretical background and case studies, (3) an introduction to the Python coding environment. The first component entails a conceptual introduction to the opportunities and the challenges of a digitally transformed and data-driven society, presentations on industry standards and legal frameworks, and discussions of critical issues such as cybersecurity and surveillance. The second component includes topics such as data science terminology, digital data and their representations, and introductions to exploratory data analysis and prominent supervised and unsupervised learning tasks. The third component offers an introduction to the Python ecosystem of data representation, processing, analysis, and visualization, starting with Jupyter notebooks, installing suitable environments, and introductions to data science related packages such as NumPy, SciPy, Matplotlib, Seaborn, and Pandas. Fundamental data science concepts are summarized and illustrated using real-world data from various disciplines. Flexible educational formats (mostly online and hybrid) allow for asynchronous learning. Lectures are combined with an exposure to Python programming and data processing and visualization environments, including hands-on practicals, examples, and exercises.

Intended Learning Outcomes

By the end of this module, students will be able to

- explain societal implications of the digital transformation,
- understand the legal data protection framework,
- carry out basic data processing and visualization tasks,
- apply fundamental data science methods to structured data,
- understand the logic of Python scripts and functions,
- compose Python code using templates

Indicative Literature

Ani Adhikari, John DeNero, David Wagner. Computational and Inferential Thinking: The Foundations of Data Science. Originally developed for the UC Berkeley course Data 8: Foundations of Data Science. An online version of the textbook is available at https://inferentialthinking.com/.

The Alan Turing Institute, Data Science for the Social Good.

Philip D. Brooker. Programing with Python for Social Scientists. Sage 2020.

Shin Takahasi, Iroha Inoue. The Manga Guide to Linear Algebra. Trend-Pro 2012.

Steven S. Skiena. The Data Science Design Manual. Springer 2017.

Jake Vanderplas. Python Data Science Handbook. O'Reilly 2016. An online version is available at https://jakevdp.github.io/PythonDataScienceHandbook/.

Shoshana Zuboff. The Age of Surveillance Capitalism. London: Profile 2019.

Usability and Relationship to other Modules

Examination Type: Module Examination

Type: Written Examination — Duration/Length: 180 min
Scope: All intended learning outcomes of the module. — Weight: 100 %

Module achievement: 50% of the assignments need to be correctly solved.

Completion: To pass this module, the examination has to be passed with at least 45%

6.4 Algorithms and Data Structures

Module Name	**Module Code**	**Level (type)**	**CP**
Algorithms and Data Structures	ACS-103	Year 1 (CHOICE)	7.5

Module Components

Number	Name	Type	CP
ACS-103-A	Algorithms and Data Structures	Lecture (online)	7.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Dr. Kinga Lipskoch	• Computer Science (CS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring)	1 semester
☒ Programming in C and C++	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
52.5 h	57.5 h	20 h	57.5 h	187.5 h

Recommendations for Preparation

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.

Content and Educational Aims

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.

Intended Learning Outcomes

By the end of this module, students will be able to

1. explain asymptotic (time and memory) complexities and respective notations;
2. able to prove asymptotic complexities of algorithms;
3. illustrate basic data structures such as arrays, lists, queues, stacks, trees, and hash tables;
4. describe algorithmic design concepts and apply them to new problems;
5. explain basic algorithms (sorting, searching, graph algorithms, computational geometry) and their complexities;
6. 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.

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.

Examination Type: Module Examination

Assessment Type: Written examination

Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination has to be passed with at least 45%

6.5 Introduction to Cyber Physical Systems

Module Name	**Module Code**	**Level (type)**	**CP**
Introduction to Cyber Physical Systems (CPS)	ACS-104	Year 1	7.5

Module Components

Number	Name	Type	CP
ACS-104-A	Introduction to Cyber Physical Systems (CPS) Lecture	Lecture (online)	5
ACS-104-B	Introduction to Cyber Physical Systems (CPS) Tutorial	Tutorial (online)	2.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Andreas Birk	• Applied Computer Science (ACS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring)	1 semester
☒ Calculus and Elements of Linear Algebra I	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	75 h	20 h	57.55 h	187.5 h

Recommendations for Preparation

Students are expected to be familiar with the core elements of calculus and linear algebra.

Content and Educational Aims

The area of Cyber Physical Systems (CPS) deals with the interface between the digital and the physical world, i.e., the relations and interfaces of software to computer hardware, embedded systems, sensors and actuators, and networking. Application examples range from large entities like power-grids, factories, or warehouses, down to smaller systems like automobiles, home automation, or machinery in production or warehouses. CPS builds on interconnected smart devices and intelligent autonomous systems, which may range from small simple sensor-nodes to more capable systems that may also feature mobility and manipulation. It hence relates software development to aspects of computer architecture, communications, system integration, modelling, control, and artificial intelligence.

Intended Learning Outcomes

Upon completion of this module, students will be able to

1. Describe the different use-cases and application areas of CPS
2. Explain the components of CPS and their interplay
3. Understand computer architecture and be able to apply core concepts within embedded computing
4. Generate software interfaces to sensors and actuators
5. Understand the networking aspects related to CPS and apply them within the context of embedded computing
6. Explain real-time requirements and understand the related core software concepts and algorithms
7. Be able to model systems
8. Understand and apply the basics of control of physical systems in form of software

9. Explain core concepts and methods of software for intelligent autonomous systems
10. Understand and use software methods for remote access for monitoring, operation, and maintenance of physical systems and processes

Indicative Literature

Usability and Relationship to other Modules

Examination Type: Module Component Examination

Module Component 1: Lecture
Assessment Type: Written examination — Duration/length: 120 min
Weight: 67%

Scope: All intended learning outcomes of the module (with focus on theory).

Module Component 2: Tutorial
Assessment Type: Practical assignments
Weight: 33%

Scope: All intended learning outcomes of the module (with focus on practical content).

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.6 Software Design and Prototyping

Module Name	**Module Code**	**Level (type)**	**CP**
Software Design and Prototyping	ACS-105	Year 1	7.5

Module Components

Number	Name	Type	CP
ACS-105-A	Software Design and Prototyping Lecture	Lecture (online)	5
ACS-105-B	Software Design and Prototyping Tutorial	Tutorial (online)	2.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Andreas Birk	• Applied Computer Science (ACS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring)	1 semester
☒ Programming in C/C++	☒ Distributed Development II			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	75 h	20 h	57.5 h	187.5 h

Recommendations for Preparation

Students are expected to be familiar with programming in C/C++ and the basics of collaborative, remote software development.

Content and Educational Aims

During the early phases of software projects, it is often unclear what the exact requirements are and how a suitable software design could look like. Since wrong decisions taken during the early phases of a software project frequently have significant impact on the completion time and the overall costs of a software project, it is often
desirable to quickly construct prototype systems. Prototype systems can not only be used to collect early feedback in order to clarify requirements. They can also be used to acquire additional customers. This module introduces software design pattern with a specific focus on the construction of early prototypes, sometimes also called mockup systems.

Intended Learning Outcomes

Upon completion of this module, students will be able to

1. select software architectures supporting fast prototyping
2. implement interaction prototypes using suitable mockup tools
3. implement backend and server prototypes using suitable mockup tools
4. derive designs of interaction prototypes from incomplete user input
5. conduct an evaluation of mockup prototypes with target users
6. be able to revise prototypes efficiently in an agile manner
7. effectively work in a team prototyping different software components
8. create mock objects that can be used effectively for unit tests

Indicative Literature

Usability and Relationship to other Modules

Examination Type: Module Component Examination

Module Component 1: Lecture
Assessment: Written examination — Duration: 60 min
Weight: 67 %

Scope: Intended Learning outcomes 1,4 and 6.

Module Component 2: Tutorial
Assessment: Project Assessment — Weight: 33%
Scope: Intended Learning outcomes 2,3,5,7 and 8.

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.7 Distributed Development

Module Name Distributed Development		**Module Code** ACS-106	**Level (type)** Year 1	**CP** 5
Module Components				
Number	Name		Type	CP
ACS-106-A	Distributed Development I		Lecture & Lab (online)	2.5
ACS-106-B	Distributed Development II		Lecture & Lab (online)	2.5
Module Coordinator Prof. Dr. Andreas Birk	**Program Affiliation** • Applied Computer Science (ACS)		**Mandatory Status** Mandatory for ACS	
Entry Requirements Pre-requisites ☒ None	Co-requisites ☒ Programming in C/C++	Knowledge, Abilities, or Skills	**Frequency** Annually Fall (I) Spring (II)	**Duration** 2 semester

Student Workload				
Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
17.5 h	52.5 h	20 h	35 h	125 h

Recommendations for Preparation

Previous experience with programming is a plus but not required.

Content and Educational Aims

Software development is increasingly done in collaborative teams who work in a remote fashion, i.e., with team members who are spatially distributed at different locations, sometimes even across different time-zones. This can be very convenient for employers, who can recruit from around the globe without the need for expecting the employees to relocate, as well as for the employees, who gain some freedom in where and when they execute their tasks. But it includes also quite some challenges, e.g., for the development of a joined approach, the coordination of tasks, or the meeting of deadlines. This module provides a hands-on introduction into the methods and tools for handling these opportunities and challenges.

Intended Learning Outcomes

Upon completion of this module, students will be able to

1. Understand the opportunities and challenges that are involved in collaborative, remote software development
2. Comprehend the needs for and limitations of synchronous online-meeting tools
3. Use the different standard features of tools for synchronous online-meetings
4. Comprehend the concepts of versioning software and be able to apply them
5. Understand the pro's and con's of asynchronous online communication
6. Use standard features of online communication teams for brain-storming and the development of a joined approach to solve problems and the distribution of tasks
7. Understand the needs for calendars and to-do lists and how to handle them
8. Comprehend bug-trackers and be able to use them

9. Understand the possibilities and limitations, e.g., legal restrictions, of monitoring tools that, e.g., keep track of the time spend on tasks per individual team member

Indicative Literature

Usability and Relationship to other Modules

Examination Type: Module Component Examination

Module Component 1: Lecture & lab
Assessment: Practical assessment

Weight: 50%

Module Component 2: Lecture & lab
Assessment: Practical assessment

Weight: 50%

Scope: All intended learning outcomes of the module.

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.8 Databases and Web Services

Module Name	**Module Code**	**Level (type)**	**CP**
Databases and Web Services	ACS-201	Year 2 (CORE)	7.5

Module Components

Number	Name	Type	CP
ACS-201-A	Databases and Web Services	Lecture (online)	5
ACS-201-B	Databases and Web Services - Project	Project (online)	2.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Peter Baumann	• Computer Science (CS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Fall)	1 semester
☒ Algorithms and Data Structures	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	115 h	20 h	17.5 h	187.5 h

Recommendations for Preparation

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 on https://docs.python.org/).

Content and Educational Aims

This module offers a combined introduction to databases and web services. The database part starts with database design using the Entity Relationship (ER) and Unified Modeling Language (UML) models, followed by relational databases and querying them through SQL, relational design theory, indexing, query processing, transaction management, and NoSQL/Big Data databases. In the web services part, the topics addressed include markup languages, three-tier application architectures, and web services. Security aspects are addressed from both perspectives.

A hands-on group project complements the theoretical aspects: on a self-chosen topic, students implement the core of a web-accessible information system using Python (or a similar language), MySQL, and Linux, guided through homework assignments.

Intended Learning Outcomes

By the end of this module, students will be able to

1. read and write ER and UML diagrams;
2. design and normalize data models for relational databases;
3. write SQL queries and understand their evaluation by a database server;
4. explain the concept of transactions and how to use transactions in application design;
5. use web application frameworks to create dynamic websites;
6. describe the differences of selected NoSQL data models and make a requirement-driven choice;
7. restate three-tier architectures and their components;
8. discuss the principles and basic mechanisms of reactive website design;
9. summarize the security and privacy issues in the context of databases and web services.

Indicative Literature

Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer D. Widom: Database Systems: The Complete Book. 2nd edition, Pearson, 2008.

Ragu Ramakrishnan: Database Management Systems. 3rd edition, McGraw Hill, 2003.

James Lee: Open Source Web Development with LAMP. Pearson, 2003.

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.

Examination Type: Module Component Examinations

Module Component 1: Lecture

Assessment Type: Written examination — Duration: 120 min, Weight: 67%

Scope: All intended learning outcomes of the excluding the practical aspects

Module Component 2: Project

Assessment Type: Project Assessment — Weight: 33%

Scope: All practical aspects of the intended learning outcomes

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.9 Operating Systems

Module Name	**Module Code**	**Level (type)**	**CP**
Operating Systems	ACS-202	Year 2 (CORE)	7.5

Module Components

Number	Name	Type	CP
ACS-202-A	Operating Systems	Lecture (online)	7.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Jürgen Schönwälder	• Computer Science (CS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Fall)	1 semester
☒ Introduction to Computer Science ☒ Algorithms and Data Structures	☒ None			

Student Workload

Asynchronous Lecture / Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
52.5 h	57.5h	20 h	57.5 h	187.5 h

Recommendations for Preparation

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.

Content and Educational Aims

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.

Intended Learning Outcomes

By the end of this module, students will be able to

1. explain the differences between processes, threads, application programs, libraries, and operating system kernels;
2. describe well-known mutual exclusion and coordination problems;
3. use semaphores to achieve mutual exclusion and solve coordination problems;
4. use mutual exclusion locks and condition variables to solve synchronization and coordination problems;
5. illustrate how deadlocks can be avoided, detected, and resolved;
6. summarize the different mechanisms to realize virtual memory and their trade-offs;
7. solve basic inter-process communication problems using signals and pipes;
8. use socket inter-process communication primitives;
9. multiplex I/O activities using suitable system calls and libraries;
10. describe file system programming interfaces and the design of file systems at the operating system kernel level;
11. explain how memory mapping can improve I/O performance;
12. restate the functionality of a linker and the difference between static linking and dynamic linking;
13. outline how different device types are supported by Unix-like kernels;
14. 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.

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.

Examination Type: Module Examination

Assessment Type: Written examination — Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of the module
Module achievement: 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.

Completion: To pass this module, the examination has to be passed with at least 45%

6.10 Data Analytics and Modeling

Module Name	**Module Code**	**Level (type)**	**CP**
Data Analytics and Modeling	CO-710	Year 2	7.5

Module Components

Number	Name	Type	CP
CO-710-A	Data Analytics and Modeling	Lecture (online)	7.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
	• Minor in Data Science	Mandatory for ACS, MDDA and Minor in Data Science

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Fall)	1 semester
☒ None	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
52.5 h	57.5h	20h	57.5 h	187.5 h

Recommendations for Preparation

Required for solving the coding assignments are Python skills at the level achieved after successful completion of the module Introduction to Data Science. Furthermore, students are encouraged to review first-year level statistics and linear algebra.

Content and Educational Aims

The module offers an introduction to the principles of data analytics and predictive data modeling and is structured into four parts. First, essential concepts from statistics are reviewed in the data modeling context, illustrating key ideas including randomness, distributions, and confidence regions. Examples and case studies are discussed to distinguish between proper and improper uses of statistics. Basic linear algebra is reviewed in the second part of the module, emphasizing vectors, distances, linear equations, matrices, and inversion. Key ideas such as the least squares approach are motivated with geometrical principles. The third part of the module is concerned with matrix decompositions such as the Singular Value Decomposition (SVD) and its close relatives Principal Component Analysis (PCA) and Empirical Orthogonal Function (EOF) analysis. The fourth part clarifies the distinction between linear and nonlinear modeling, and introduces key nonlinear techniques. Flexible educational formats (mostly online and hybrid) allow for asynchronous learning. Lectures are combined with Python exercises. Disciplinary applications and case studies are immersed as bridging elements.

Intended Learning Outcomes

Upon completion of this module, students will be able to

1. identify important problem types and solution approaches in data analytics,
2. understand how key concepts from statistics and linear algebra enter data science,
3. explain matrix decompositions and their usage in data science,
4. discuss regularization concepts and optimality criteria in data analytics,
5. know the basics of nonlinear modeling and related computational approaches,
6. convert data structures to Python/NumPy arrays for usage in data modeling,
7. apply Python statistics and linear algebra tools in data analytics and modeling.

Indicative Literature

Ani Adhikari, John DeNero, David Wagner. Computational and Inferential Thinking: The Foundations of Data Science 2019. Originally developed for the UC Berkeley course Data 8: Foundations of Data Science. An online version of the textbook is available at https://inferentialthinking.com/.

Steven S. Skiena. The Data Science Design Manual. Springer 2017.

Gilbert Strang: Linear Algebra and Learning from Data. Wellesley-Cambridge 2019. See https://math.mit.edu/~gs/learningfromdata/.

Joe Suzuki: Statistical Learning with Math and Python. Springer 2021.

Jake Vanderplas. Python Data Science Handbook. O'Reilly 2016. An online version is available at https://jakevdp.github.io/PythonDataScienceHandbook/.

Usability and Relationship to other Modules

Examination Type: Module Examination

Type: Written Examination — Duration/Length: 180 min
Scope: All intended learning outcomes of the module. — Weight: 100 %

Module achievement: 50% of the assignments need to be correctly solved.

Completion: To pass this module, the examination has to be passed with at least 45%

6.11 Software Engineering

Module Name	**Module Code**	**Level (type)**	**CP**
Software Engineering	ACS-203	Year 2 (CORE)	7.5

Module Components

Number	Name	Type	CP
ACS-203-A	Software Engineering	Lecture (online)	2.5
ACS-203-B	Software Engineering Project	Project (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Peter Baumann	• Computer Science (CS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring)	1 semester
☒ Databases and Web Services	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	132.5 h	10 h	10 h	187.5 h

Recommendations for Preparation

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.

Content and Educational Aims

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.

Intended Learning Outcomes

By the end of this module, students will be able to

1. understand and apply object-oriented design patterns;
2. read and write UML diagrams;
3. contrast the benefits and drawbacks of different software development models;
4. design and plan a larger software project involving a team development effort;
5. translate requirements formulated by a customer into computer science terminology;
6. evaluate the applicability of different software engineering models for a given software development project;
7. assess the quality of a software design and its implementation;
8. apply tools that assist in the various stages of a software development process;
9. 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.

Usability and Relationship to other Modules

-

Examination Type: Module Component Examinations

Module Component 1: Lecture

Assessment Type: Written examination — Duration: 60 min, Weight: 33%

Scope: The first three intended learning outcomes of the module (the lecture module component)

Module Component 2: Project

Assessment Type: Project Assessment — Weight: 66%

Scope: The remaining intended learning outcomes of the module (the project module component)

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.12 Artificial Intelligence

Module Name Artificial Intelligence		**Module Code** ACS-204	**Level (type)** Year 2 (CORE)	**CP** 7.5
Module Components				
Number	Name		Type	CP
ACS-204-A	Artificial Intelligence		Lecture	5
ACS-204-B	Artificial Intelligence Tutorial		Tutorial	2.5
Module Coordinator Prof. Dr. Andreas Birk	**Program Affiliation** • Applied Computer Science (ACS)		**Mandatory Status** Mandatory for ACS	
Entry Requirements Pre-requisites ☒ Programming in C/C++ ☒ Introduction to RIS OR ☒ Introduction to CPS	Co-requisites ☒ None	Knowledge, Abilities, or Skills	**Frequency** Annually (Spring)	**Duration** 1 semester

Student Workload				
Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	17.5 h	20 h	115 h	187.5 h

Recommendations for Preparation

Revise content of the pre-requisite modules.

Content and Educational Aims

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. 39

Intended Learning Outcomes

By the end of this module, students should be able to

1. outline and explain the history, general developments, and application areas of AI;
2. apply the basic concepts and methods of behavior-oriented AI;
3. use concepts and methods of search algorithms for problem-solving;
4. explain the basic concepts of path-planning as an application example for domain-specific search;
5. apply basic path-planning algorithms and to compare their relations to general search algorithms;
6. write and explain concepts of propositional and first-order logic;
7. use logic representations and inference for basic examples of artificial planning systems;
8. apply AI concepts and methods to develop software.

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.

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.
-

Examination Type: Module Component Examinations

Module Component 1: Lecture

Assessment Type: Written examination — Duration: 120 min, Weight: 67%

Scope: All intended learning outcomes of the excluding the practical aspects

Module Component 2: Project

Assessment Type: Project Assessment — Weight: 33%

Scope: All practical aspects of the intended learning outcomes

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.13 Machine Learning

<table>
<tr><td colspan="3">Module Name
Machine Learning</td><td>Module Code
ACS-205</td><td>Level (type)
Year 2 (CORE)</td><td>CP
7.5</td></tr>
<tr><td colspan="6">Module Components</td></tr>
<tr><td>Number</td><td colspan="3">Name</td><td>Type</td><td>CP</td></tr>
<tr><td>ACS-205-A</td><td colspan="3">Machine Learning</td><td>Lecture (online)</td><td>5</td></tr>
<tr><td>ACS-205-B</td><td colspan="3">Machine Learning Tools</td><td>Lab (online)</td><td>2.5</td></tr>
<tr><td>Module Coordinator
Prof. Dr. Andreas Birk</td><td colspan="3">Program Affiliation
• Applied Computer Science (ACS)</td><td colspan="2">Mandatory Status
Mandatory for ACS</td></tr>
<tr><td colspan="3">Entry Requirements
Pre-requisites: ☒ None
Co-requisites: ☒ None
Knowledge, Abilities, or Skills:
• Knowledge and command of probability theory and methods, as in the module “Probability and Random Process (CTMS-MAT-12)</td><td>Frequency
Annually (Spring)</td><td colspan="2">Duration
1 semester</td></tr>
<tr><td colspan="6">Student Workload</td></tr>
<tr><td>Asynchronous Self Study</td><td>Interactive Learning</td><td>Exam Preparation/ Lab assignments</td><td colspan="2">Independent Study</td><td>Hours Total</td></tr>
<tr><td>35 h</td><td>75 h</td><td>20 h</td><td colspan="2">57.5 h</td><td>187.5 h</td></tr>
<tr><td colspan="6">Recommendations for Preparation
None</td></tr>
<tr><td colspan="6">Content and Educational Aims
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. The ML lecture is complemented in this module by an online tutorial where the application-oriented side of software development in the context of ML is considered.
Modern machine learning in industry and research requires the knowledge of a comprehensive stack of tools and systems that allow to store and administrate data (e.g. Amazon S3, Kaggle, Dataverse, GIT LFS), extract features for various applications (e.g. Word2Vec, TSFEL), build up machine learning pipelines of training, testing, and hyperparameter optimization (e.g. skit-learn, Keras, TensorFlow, PyToarch) and ultimately deploy finalized models (e.g. TensorFlow Serving, MLFlow). This module gives exposure to a regularly updated latest state of the art set of tools that are relevant</td></tr>
</table>

for the practical use of Machine Learning. It thereby complements the more theoretical and methods-driven module "Machine Learning" with market-oriented skills.

Intended Learning Outcomes

By the end of this module, students should be able to

1. understand the notion of probability spaces and random variables;
2. understand basic linear modeling and estimation techniques;
3. understand the fundamental nature of the "curse of dimensionality;"
4. understand the fundamental nature of the bias-variance problem and standard coping strategies;
5. use elementary classification learning methods (linear discrimination, radial basis function networks, multilayer perceptions);
6. implement an end-to-end learning suite, including feature extraction and objective function optimization with regularization based on cross-validation.
7. deploy ML tools in an application context.

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.

Usability and Relationship to other Modules

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

Examination Type: Module Component Examinations

Module Component 1: Lecture & Project

Assessment Type: Written examination — Duration: 120 min, Weight: 67%

Scope: All intended learning outcomes of the excluding the practical aspects

Module Component 2: Lab

Assessment Type: Practical Assignments — Weight: 33%

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.14 Computer Graphics

Module Name	**Module Code**	**Level (type)**	**CP**
Computer Graphics	ACS-303	Year 3 (Specialization)	5

Module Components

Number	Name	Type	CP
ACS-303-A	Computer Graphics	Lecture (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Alexander Omelchenko	• Computer Science (CS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Fall)	1 semester
☒ Algorithms and Data Structures	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	20 h	20 h	50 h	125 h

Recommendations for Preparation

None

Content and Educational Aims

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.

Intended Learning Outcomes

By the end of this module, students will be able to

1. construct 3D geometry representations;
2. apply 3D transformations;
3. understand the algorithms and optimizations applied by graphics rendering systems;
4. explain the stages of modern computer graphics programmable pipelines
5. implement simple computer graphics applications using graphics frameworks such as OpenGL;
6. 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.

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.

Examination Type: Module Examination

Assessment Type: Written examination

Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination has to be passed with at least 45%.

6.15 Computer Networks

Module Name Computer Networks		**Module Code** ACS-304	**Level (type)** Year 2 (CORE)	**CP** 5
Module Components				
Number	Name		Type	CP
ACS-304-A	Computer Networks		Lecture (online)	5
Module Coordinator Prof. Dr. Jürgen Schönwälder	**Program Affiliation** • Computer Science (CS)		**Mandatory Status** Mandatory for ACS	
Entry Requirements Pre-requisites ☒ Algorithms and Data Structures	Co-requisites ☒ Operating Systems	Knowledge, Abilities, or Skills	**Frequency** Annually (Fall)	**Duration** 1 semester

Student Workload				
Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	20h	20 h	50 h	125 h

Recommendations for Preparation

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/).

Content and Educational Aims

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 course should finally understand the technical complexity behind every day 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 requires to take engineering decisions that reflect non-technical requirements.

Intended Learning Outcomes

By the end of this module, students will be able to

1. recall layering principles and the OSI reference model;
2. articulate the organization of the Internet and the organization involved in providing Internet services;
3. describe media access control, flow control, and congestion control mechanisms;
4. explain how local area networks differ from global networks;
5. illustrate how frames are forwarded in local area networks;
6. contrast addressing mechanisms and translations between addresses used at different layers;
7. demonstrate how the Internet network layer forwards packets;
8. present how routing algorithms and protocols are used to determine and select routes;
9. describe how the Internet transport layer provides different end-to-end services;
10. demonstrate how names are resolved to addresses and vice versa;
11. summarize how application layer protocols send and access electronic mail or access resources on the world-wide web;
12. design and implement simple application layer protocols;
13. recognize to which extent computer networks are fragile and evaluate strategies to cope with the fragility;
14. 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, Nick Feamster, David Wetherall: Computer Networks, 6th Edition, Pearson Education Limited, 2021.

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.

Examination Type: Module Examination

Assessment Type: Written examination — Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination has to be passed with at least 45%.

6.16 Web Application Development

Module Name			**Module Code**	**Level (type)**	**CP**
Web Application Development			ACS-305	Year 3 (Specialization)	5
Module Components					
Number	Name			Type	CP
ACS-305-A	Web Application Development			Lecture (online)	2.5
ACS-305-B	Web Application Development - Project			Project (online)	2.5
Module Coordinator Prof. Dr. Alexander Omelchenko	**Program Affiliation** • Computer Science (CS)			**Mandatory Status** Mandatory for ACS	
Entry Requirements Pre-requisites ☒ None	Co-requisites ☒ None	Knowledge, Abilities, or Skills	**Frequency** Every semester (Fall/Spring)	**Duration** 1 semester	
Student Workload					
Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total	
17.5 h	50 h	17.5 h	40h	125 h	
Recommendations for Preparation None					

Content and Educational Aims

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

By the end of this module, students will be able to

1. explain the document object model behind HTML and its relation to CSS;
2. discuss the principles and basic mechanisms of reactive website design;
3. analyze the interactions between web applications and web services.
4. use languages such as Java, Kotlin, or Swift to implement mobile web applications;
5. 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.

Usability and Relationship to other Modules

Examination Type: Module Component Examinations

Module Component 1: Lecture

Assessment Type: Written examination — Duration: 120 min, Weight: 50%

Scope: First group of intended learning outcomes of the module

Module Component 2: Project

Assessment Type: Project assignment — Weight: 50%

Scope: Second group of intended learning outcomes of the module

Completion: To pass this module, the examination of each module component has to be passed with at least 45%.

6.17 Human Computer Interaction

Module Name Human Computer Interaction			**Module Code** ACS-306	**Level (type)** Year 3 (Specialization)	**CP** 5
Module Components					
Number	Name			Type	CP
ACS-306-A	Human Computer Interaction			Lecture (online)	5
Module Coordinator Prof. Dr. Francesco Maurelli	**Program Affiliation** • Robotics and Intelligent Systems (RIS)			**Mandatory Status** Mandatory elective for ACS	
Entry Requirements Pre-requisites ☒ None	Co-requisites ☒ None	Knowledge, Abilities, or Skills	**Frequency** Annually (Fall)	**Duration** 1 semester	

Student Workload				
Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	20 h	20 h	50 h	125 h

Recommendations for Preparation

None

Content and Educational Aims

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.

Intended Learning Outcomes

By the end of this module, students should be able to

- explain the evolution of human–computer interaction models;
- design and implement simple graphical user interfaces;
- explain ergonomic principles guiding the design of user interfaces;
- illustrate different types of interaction (e.g., visual, voice, gestures) and their usability aspects;
- evaluate aspects of and tradeoffs between usability, learnability, efficiency, and safety;
- apply scientific methods to evaluate interfaces with respect to their usability and other desirable properties;
- use prototyping tools that can be employed to create mockups of user interfaces during the early stages of a software project.

Indicative Literature

Not specified

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.
-

Examination Type: Module Examination

Assessment Type: Written examination

Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination has to be passed with at least 45%.

6.18 Collaborative Software Project

Module Name	**Module Code**	**Level (type)**	**CP**
Collaborative Software Project	ACS-301	Year 3	5

Module Components

Number	Name	Type	CP
ACS-301-A	Collaborative Software Project	Project (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Andreas Birk	• Applied Computer Science (ACS)	Mandatory for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Every semester (Fall/Spring)	1 semester
☒ Students must have successfully passed 90 CP.	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
5 h	60 h	h	60 h	125 h

Recommendations for Preparation

Content and Educational Aims

The project enables the students to deepen their knowledge and skills in one or multiple areas of the 1st and especially 2nd year. They are exposed to state-of-the-art research with the goal to derive ideas and strategies to address application-oriented problems and to develop software for them. Students learn how to organize and execute an application-oriented research and development (R&D) project and how to present the results in the format of a white-paper. Students are expected to organize themselves in group work under the guidance of the instructor.

Intended Learning Outcomes

Upon completion of this module, students will be able to

1. Understand state-of-the-art research papers in a chosen field of specialization
2. Plan a research project to reproduce research results or to extend ideas of recent research results
3. Explain research questions and choose suitable methodologies to address them
4. Use methods and tools for remote collaborative software development
5. Document a research project in the style of a typical white-paper

Indicative Literature

State-of-the-art literature provided by the instructor

Usability and Relationship to other Modules

•

Examination Type: Module Examination

Assessment: Project report (4,000 words) Weight: 100%

Scope: All intended learning outcomes of the module.

Completion: To pass this module, the examination has to be passed with at least 45%.

6.19 Internship / Startup and Career Skills

Module Name Internship / Startup and Career Skills			**Module Code** CA-INT-900	**Level (type)** Year 3 (CAREER)	**CP** 15
Module Components					
Number	Name			Type	CP
CA-INT-900-0	Internship			Internship	15
Module Coordinator Clémentine Senicourt & Dr. Tanja Woebs (CSC Organization); SPC / Faculty Startup Coordinator (Academic responsibility)	**Program Affiliation** • CAREER module for undergraduate study programs			**Mandatory Status** Mandatory for all undergraduate study programs except IEM	
Entry Requirements Pre-requisites ☒ at least 15 CP from CORE modules in the major	Co-requisites ☒ None	Knowledge, Abilities, or Skills • Information provided on CSC pages (see below) • Major specific knowledge and skills	**Frequency** Annually (Spring/Fall)	**Duration** 1 semester	

Student Workload				
Internship	Interactive Learning	Internship Event	Independent Study	Hours Total
308 h	33 h	2 h	32 h	375 h

Forms of Learning and Teaching

- Internship/Start-up
- Internship event
- Seminars, info-sessions, workshops and career events
- Self-study, readings, online tutorials

Recommendations for Preparation

- 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

Content and Educational Aims

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 Student Career Support 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.

Intended Learning Outcomes

By the end of this module, students should be able to

1. describe the scope and the functions of the employment market and personal career development;
2. 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 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 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 professional decisions based on theoretical knowledge and academic methods;
6. reflect on their professional conduct in the context of the expectations of and consequences for employers and their society;
7. reflect on and set their own targets for the further development of their knowledge, skills, interests, and values;
8. 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 observations and reflections in a professional network.

Indicative Literature

Not specified

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.

Examination Type: Module Examination

Assessment Type: Internship Report or Business Plan and Reflection	Length: approx. 3.500 words
Scope: All intended learning outcomes	Weight: 100%

6.20 Bachelor Thesis

Module Name Bachelor Thesis ASC			**Module Code** ACS-400	**Level (type)** Year 3 (CAREER)	**CP** 10
Module Components					
Number	Name			Type	CP
ACS-400-T	Thesis ACS			Thesis	10
Module Coordinator Study Program Chair	**Program Affiliation** • all Bachelor Programs			**Mandatory Status** Mandatory for all Bachelor Programs	
Entry Requirements			**Frequency**	**Duration**	
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Every semester (Fall/Spring)	14 week lecture period	
☒ Students must be in their third year and have taken at least 30 CP from Year 2 modules.	☒ None	Comprehensive knowledge of the subject and deeper insight into the chosen topic; ability to plan and undertake work independently; skills to identify and critically review literature.			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study & Lab work	Hours Total
5 h	20 h	0 h	225 h	250 h

Recommendations for Preparation

- Identify an area or a topic of interest and discuss this with your prospective supervisor in good time.
- 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 again the University's Code of Academic Integrity and Guidelines to Ensure Good Academic Practice.

Content and Educational Aims

This module is a mandatory graduation requirement for all undergraduate students to demonstrate their ability to deal with a problem from their respective major subject independently by means of academic/scientific methods within a set period. Although supervised, the module requires the student to be able to work independently and regularly and set their own goals in exchange for the opportunity to explore a topic that excites and interests them personally and which a faculty member is interested to supervise. Within this module, students apply their acquired knowledge about the major discipline, skills, and methods to conduct 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 the results.

This module consists of an independent thesis. The thesis 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 conclusions.

Intended Learning Outcomes

On completion of this module, students will be able to

1. independently plan and organize advanced learning processes;
2. design and implement appropriate research methods taking full account of the range of alternative techniques and approaches;
3. collect, assess and interpret relevant information;
4. draw scientifically founded conclusions that consider social, scientific and ethical insights;
5. apply their knowledge and understanding to a context of their choice;
6. develop, formulate and advance solutions to problems and arguments in their subject area, and defend these through argument;
7. discuss information, ideas, problems and solutions with specialists and non-specialists;

Usability and Relationship to other Modules

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

Examination Type: Module Examination

Assessment component 1
Assessment Type: Thesis — Weight: 80%

Scope: All intended learning outcomes, mainly 1-6.

Assessment component 2
Assessment Type: Presentation — Duration: approx. 15 to 30 minutes
Weight: 20%

Scope: The presentation focuses mainly on ILOs 6 and 7, but by nature of these ILOs also touches on the others.

Completion: To pass this module, both module component examinations have to be passed with at least 45%.

7 Management modules

7.1 Digital Business Models and Functions

Module Name	**Module Code**	**Level (type)**	**CP**
Digital Business Models and Functions	MDSSB-DSOC-02	Year 1 / 2 (CORE)	5

Module Components

Number	Name	Type	CP
MDSSB-DTRANS-02	Digital Business Models and Functions	Lecture (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
NN	• Data Science for Society and Business (DSSB)	Mandatory for DSSB Mandatory elective for ACS

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring)	1 semester
☒ None	☒ None	Academic writing skills Good understanding of the principles of business functions		

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	10 h	20 h	60 h	125 h

Recommendations for Preparation

None.

Content and Educational Aims

Businesses today have just begun to understand the potential of data abundance. Companies such as Amazon and Google were among the pioneers of data-driven business models. Many technology-based start-ups are eager to follow their lead. The data-driven revolution in the business world is nothing less than what Schumpeter termed a process of creative destruction. In this case, the destruction is of the long-established ways of doing business. The representatives of this new-age alternative business models range from shared economies and platform businesses to subscription models, even in the most traditional industries.

In this module, we will uncover the antecedents, drivers, and potentials of a data-driven economy by focusing on entrepreneurs and how their experiments creatively destruct the way we used to do business. We will explain why e-commerce is the fastest growing segment in retail today. We will examine e-commerce business models, technology infrastructure, e-commerce marketing and advertising concepts, social networks, auctions, and portals, as well as ethical, social, and political issues with the help of prominent case studies. At the end of the module, students will be able to build their own e-commerce (small-scale) companies.

Intended Learning Outcomes

By the end of this module, students should be able to

1. know about the development of business models on the Internet
2. conceptually understand how to build an e-commerce presence
3. comprehensively understand e-commerce security and payment systems
4. critically understand e-commerce marketing and advertising
5. discuss and reflect on major obstacles and possible solutions in e-commerce ethics
6. critically evaluate and design business case studies

Indicative Literature

Zott, Amit (2017) Business Model Innovation: How to Create Value in a Digital World. Marketing Intelligence Review 9 (1) DOI: https://doi.org/10.1515/gfkmir-2017-0003

Wirtz (2019) Digital Business Models: Concepts, Models, and the Alphabet Case Study. Cham: Springer Nature.

Usability and Relationship to other Modules

This module focuses on digital business concepts and digital business models. It connects to all business modules in the "Society and Business" track to the core "Digital Transformation and Innovation" and "Artificial Intelligence in Business and Society" modules. However, it also forms the base for students who want to develop their own business ideas in the discovery section of the program and outside academia.

Examination Type: Module Examination

Assessment Type: Term Paper

Length: 5000 words
Weight: 100%

Scope: All intended learning outcomes of the module.

Completion: To pass this module, the examination has to be passed with at least 45%.

7.2 Marketing & Methods

Module Name			**Module Code**	**Level (type)**	**CP**
Marketing & Methods			CTMS-MET-20	Year 2 (Methods)	5
Module Components					
Number	Name			Type	CP
CTMS-20	Marketing & Methods			Lecture (online)	5
Module Coordinator Dr. Mathias Meckel	**Program Affiliation** • CONSTRUCTOR Track area			**Mandatory Status** Mandatory MDDA Mandatory elective for ACS	
Entry Requirements Pre-requisites ☒ None	Co-requisites ☒ None	Knowledge, Abilities, or Skills	**Frequency** Annually (Fall)	**Duration** 1 semester	

Student Workload				
Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
80 h	60 h	h	45 h	125 h

Recommendations for Preparation

N.A.

Content and Educational Aims

This module is focused on key aspects of marketing and its methodologies used in today's marketing practice. State-of-the-art methods including the usage of data and approaches will be at the core of the module.

The overall goal of this module is to help students without prior marketing knowledge to learn, understand and practice the fundamentals of applied marketing methodology. This module helps students to understand today's marketing challenges in a complex world, where unpredictable is common, and where managers need to focus on achieving goals rather than repetitive tasks.
Students learn to develop and present consumer-centered and theory-based solutions for real-world marketing challenges.

Major challenges and concerns will be reflected:

- the role of the customer and data in a transformed business world
- state-of-the-art methods and marketing techniques
- ethics and security issues.

Intended Learning Outcomes

Upon completion of this module, students will be able to:

1. Develop practical knowledge and marketing skills, and mind sets to master the challenges of today's markets
2. Understand (routine) marketing processes in various context and how to state-of-the art methodology to inform marketing decisions
3. Summarize and classify the new data- and customer-driven methodologies in a marketing context
4. Understand the idea and potential for value-creation of consumer-centricity
5. Apply innovative creativity methods and processes for marketing

Indicative Literature

Kotler, Keller, Chernev (2021): Marketing Management, Global Edition, 16th edition

Usability and Relationship to other Modules

Examination Type: Module Examination

Assessment Type: Presentation

Duration: 30 min
Weight: 100%

Scope: All intended learning outcomes.

Completion: To pass this module, the examination has to be passed with at least 45%.

8 Constructor Track Modules

8.1 Methods

8.1.1 Calculus and Elements of Linear Algebra I

<table>
<tr><td colspan="3">Module Name

Calculus and Elements of Linear Algebra I</td><td>Module Code

CTMS-MAT-09</td><td>Level (type)

Year 1 (Methods)</td><td>CP

5</td></tr>
<tr><td colspan="6">Module Components

Number | Name | Type | CP
CTMS-09 | Calculus and Elements of Linear Algebra I | Lecture | 5</td></tr>
<tr><td>Module Coordinator

Dr. Keivan Mallahi Karai</td><td colspan="3">Program Affiliation

• CONSTRUCTOR Track Area</td><td colspan="2">Mandatory Status

Mandatory elective for ACS</td></tr>
<tr><td colspan="2">Entry Requirements

Pre-requisites

☒ None</td><td>Co-requisites

☒ None</td><td>Knowledge, Abilities, or Skills

• 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, 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)
• Some familiarity with elementary Calculus (limits, derivative) is helpful, but not strictly required.</td><td>Frequency

Annually (Fall)

Duration

1 semester</td><td>Forms of Learning and Teaching

• Lectures (35 hours)
• Private study (90 hours)

Workload

125 hours</td></tr>
</table>

Recommendations for Preparation

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

Content and Educational Aims

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, and Mathematics. 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 I” and “Linear Algebra”.

The lecture comprises the following topics

- Brief review of number systems, elementary functions, and their graphs
- Brief introduction to complex numbers
- Limits for sequences and functions
- Continuity
- Derivatives
- Curve sketching and applications (isoperimetric problems, optimization, error propagation)
- Introduction to Integration and the Fundamental Theorem of Calculus
- 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
- Matrix inverse

Intended Learning Outcomes

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

1. apply 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 the mathematical structures in an unfamiliar context and translate them into a mathematical problem statement;
3. 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

S.I. Grossman (2014). Calculus of one variable, 2nd edition. Cambridge: Academic Press.

S.A. Leduc (2003). Linear Algebra. Hoboken: Wiley.

K. Riley, M. Hobson, S. Bence (2006). Mathematical Methods for Physics and Engineering, third edition. Cambridge: Cambridge University Press.

Usability and Relationship to other Modules

- The module is followed by “Calculus and Elements of Linear Algebra II”. All students taking this module are expected to register for the follow-up module.
- A rigorous treatment of Calculus is provided in the module “Analysis I”. All students taking “Analysis I” are expected to either take this module or exceptionally satisfy the conditions for advanced placement as laid out in the Constructor University's Academic Policies for Undergraduate Study.
- The second-semester module “Linear Algebra” will provide a complete proof-driven development of the theory of Linear Algebra. Students enrolling in “Linear Algebra” are expected to have taken this module; in particular, the module “Linear Algebra” will assume that students are proficient in the operational aspects of Gauss elimination, matrix inversion, and their elementary applications.
-

Examination Type: Module Examination

Assessment type: Written examination

Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of this module

Completion: To pass this module, the examination has to be passed with at least 45%.

8.1.2 Calculus and Elements of Linear Algebra II

<table>
<tr><td colspan="2">Module Name
Calculus and Elements of Linear Algebra II</td><td>Module Code
CTMS-MAT-10</td><td>Level (type)
Year 1 (Methods)</td><td>CP
5</td></tr>
<tr><td colspan="5">Module Components

Number | Name | Type | CP
CTMS-10 | Calculus and Elements of Linear Algebra II | Lecture | 5</td></tr>
<tr><td>Module Coordinator

Dr. Keivan Mallahi Karai</td><td colspan="2">Program Affiliation

• CONSTRUCTOR Track Area</td><td colspan="2">Mandatory Status
Mandatory elective for ACS</td></tr>
<tr><td colspan="3">Entry Requirements

Pre-requisites: ☒ Calculus and Elements of Linear Algebra I
Co-requisites: ☒ None
Knowledge, Abilities, or Skills: • None beyond formal pre-requisites</td><td>Frequency

Annually (Spring)

Duration

1 semester</td><td>Forms of Learning and Teaching

• Lectures (35 hours)
• Private study (90 hours)

Workload

125 hours</td></tr>
<tr><td colspan="5">Recommendations for Preparation

Review the content of Calculus and Elements of Linear Algebra I</td></tr>
<tr><td colspan="5">Content and Educational Aims

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, and Mathematics. 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 I” and “Linear Algebra”.

The lecture comprises the following topics
• Directional derivatives, partial derivatives
• Linear maps
• The total derivative as a linear map
• Gradient and curl (elementary treatment only, for more advanced topics, in particular the connection to the Gauss and Stokes’ integral theorems, see module “Applied Mathematics”
• Optimization in several variables, Lagrange multipliers
• Elementary ordinary differential equations
• Eigenvalues and eigenvectors
• Hermitian and skew-Hermitian matrices
• First important example of eigendecompositions: Linear constant-coefficient ordinary differential equations
• Second important example of eigendecompositions: Fourier series
• Fourier integral transform
• Matrix factorizations: Singular value decomposition with applications, LU decomposition, QR decomposition</td></tr>
<tr><td colspan="5">Intended Learning Outcomes

By the end of the module, students will be able to
1. apply 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 the mathematical structures in an unfamiliar context and translate them into a mathematical problem statement;
3. 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.</td></tr>
</table>

Indicative Literature

S.I. Grossman (2014). Calculus of one variable, 2nd edition. Cambridge: Academic Press.
S.A. Leduc (2003). Linear Algebra. Hoboken: Wiley.
K. Riley, M. Hobson, S. Bence (2006). Mathematical Methods for Physics and Engineering, third edition. Cambridge: Cambridge University Press.

Usability and Relationship to other Modules

- 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.

Examination Type: Module Examination

Assessment type: Written examination

Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of this module

Completion: To pass this module, the examination has to be passed with at least 45%.

8.1.3 Probability and Random Processes

<table>
<tr><td colspan="3">Module Name
Probability and Random Processes</td><td>Module Code
CTMS-MAT-12</td><td>Level (type)
Year 2 (Methods)</td><td>CP
5</td></tr>
<tr><td colspan="6">Module Components

Number | Name | Type | CP
CTMS-MAT-12 | Probability and random processes | Lecture (online) | 5</td></tr>
<tr><td>Module Coordinator

Dr. Keivan Mallahi Karai</td><td colspan="3">Program Affiliation

• CONSTRUCTOR Track Area</td><td colspan="2">Mandatory Status

Mandatory for ACS</td></tr>
<tr><td colspan="3">Entry Requirements

Pre-requisites: ☒ Calculus and Elements of Linear Algebra I & II or Matrix Algebra and Advanced Calculus I & II
Co-requisites: ☒ None
Knowledge, Abilities, or Skills: 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.</td><td>Frequency

Annually (Fall)</td><td colspan="2">Duration

1 semester</td></tr>
<tr><td colspan="6">Student Workload
Asynchronous Self Study | Interactive Learning | Exam Preparation | Independent Study | Hours Total
35 h | 35 h | 20 h | 35 h | 125 h</td></tr>
<tr><td colspan="6">Recommendations for Preparation

Review all of the first year calculus and linear algebra modules as indicated in “Entry Requirements – Knowledge, Ability, or Skills” above.</td></tr>
</table>

Content and Educational Aims

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.

Intended Learning Outcomes

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

1. 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 the probabilistic structures in an unfamiliar context and translate them into a mathematical problem statement;
3. 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. Introduction to Probability, second edition. London: Chapman & Hall 2019.

S. Ghahramani. Fundamentals of Probability with Stochastic Processes, fourth edition. Upper Saddle River: Prentice Hall 2018.

Usability and Relationship to other Modules

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

Examination Type: Module Examination

Assessment type: Written examination

Duration: 120 min
Weight: 100%

Scope: All intended learning outcomes of this module

Completion: To pass this module, the examination has to be passed with at least 45%

8.2 New Skills

8.2.1 Logic (perspective I)

<table>
<tr><td colspan="3">Module Name
Logic (perspective I)</td><td>Module Code
CTNS-NSK-01</td><td>Level (type)
Constructor Track</td><td>CP
2.5</td></tr>
<tr><td colspan="6">Module Components

Number | Name | Type | CP
CTNS-01 | Logic (perspective I) | Lecture (online) | 2.5</td></tr>
<tr><td colspan="2">Module Coordinator

Prof. Dr. Jules Coleman</td><td colspan="2">Program Affiliation

• CONSTRUCTOR Track Area</td><td colspan="2">Mandatory Status

Mandatory elective for all UG students (one perspective must be chosen)</td></tr>
<tr><td colspan="3">Entry Requirements

Pre-requisites: ☒ None
Co-requisites: ☒ None
Knowledge, Abilities, or Skills</td><td>Frequency

Annually (Fall)</td><td colspan="2">Duration

1 semester</td></tr>
<tr><td colspan="6">Student Workload
Asynchronous Self Study | Interactive Learning | Exam Preparation | Independent Study | Hours Total
17.50 h | 10 h | 10 h | 25 h | 62.75 h</td></tr>
<tr><td colspan="6">Recommendations for Preparation
None</td></tr>
<tr><td colspan="6">Content and Educational Aims

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</td></tr>
</table>

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

Students acquire transferable and key skills in this module.

By the end of this module, the students will be able to

1. apply the various principles of logic and expand them to computational thinking.
2. understand the way in which logical processes in humans and in computers are similar and different at the same time.
3. 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 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.

Usability and Relationship to other Modules
Examination Type: Module Examination Assessment Type: Written Examination — Duration/Length: 60 min Weight: 100% Scope: All intended learning outcomes of the module. **Completion:** To pass this module, the examination has to be passed with at least 45%

8.2.2 Logic (perspective II)

<table>
<tr><td colspan="2">Module Name
Logic (perspective II)</td><td>Module Code
CTNS-NSK-02</td><td>Level (type)
Constructor Track</td><td>CP
2.5</td></tr>
<tr><td colspan="5">Module Components

Number | Name | Type | CP
CTNS-02 | Logic (perspective II) | Lecture (online) | 2.5</td></tr>
<tr><td>Module Coordinator

NN</td><td colspan="2">Program Affiliation

• CONSTRUCTOR Track Area</td><td colspan="2">Mandatory Status

Mandatory elective for all UG students (one perspective must be chosen)</td></tr>
<tr><td colspan="3">Entry Requirements

Pre-requisites | Co-requisites | Knowledge, Abilities, or Skills
☒ none | ☒ none</td><td colspan="2">Frequency

Annually (Fall)
Duration

1 semester</td></tr>
<tr><td colspan="5">Student workload

Asynchronous Self Study | Interactive Learning | Exam Preparation | Independent Study | Hours Total
17.50 h | 10 h | 10 h | 25 h | 62.5 h</td></tr>
<tr><td colspan="5">Recommendations for Preparation</td></tr>
<tr><td colspan="5">Content and Educational Aims

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. Once 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.</td></tr>
</table>

Intended Learning Outcomes

Students acquire transferable and key skills in this module.
By the end of this module, the students will be able to

1. apply the various principles of logic
2. explain practical relevance of non-standard logic
3. describe how many-valued logic extends basic predicate logic
4. apply basic rules of fuzzy logic to calculate partial truth values
5. sketch basic rules of temporal logic
6. implement predicates in a logic programming language
7. 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.

Usability and Relationship to other Modules

Examination Type: Module Examination

Assessment Type: Written Examination — Duration/Length: 60 min
Weight: 100%

Scope: All intended learning outcomes of the module.

Completion: To pass this module, the examination has to be passed with at least 45%

8.2.3 Causation and Correlation (perspective I)

Module Name	**Module Code**	**Level (type)**	**CP**
Causation and Correlation (perspective I)	CTNS-NSK-03	Constructor Track	2.5

Module Components

Number	Name	Type	CP
CTNS-03	Causation and Correlation (perspective I)	Lecture (online)	2.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Jules Coleman	• CONSTRUCTOR Track Area	Mandatory elective for all UG students (one perspective must be chosen)

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring/Fall)	1 semester
☒ None	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
17.50 h	10 h	10 h	25 h	62.75 h

Recommendations for Preparation

None

Content and Educational Aims

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

Students acquire transferable and key skills in this module.

By the end of this module, the students will be able to

1. formulate testable hypotheses that are designed to reveal causal connections and those designed to reveal interesting, important and useful correlations.
2. distinguish scientifically interesting correlations from unimportant ones.
3. apply critical thinking skills to evaluate information.
4. 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.

Usability and Relationship to other Modules

Examination Type: Module Examination

Assessment Type: Written Examination

Duration/Length: 60 min
Weight: 100%

Scope: All intended learning outcomes of the module.

Completion: To pass this module, the examination has to be passed with at least 45%

8.2.4 Causation and Correlation (perspective II)

Module Name	**Module Code**	**Level (type)**	**CP**
Causation and Correlation (perspective II)	CTNS-NSK-04	Constructor Track	2.5

Module Components

Number	Name	Type	CP
CTNS-04	Causation and Correlation	Lecture (online)	2.5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Dr. Keivan Mallahi-Karai Dr. Eoin Ryan Dr. Irina Chiaburu	• CONSTRUCTOR Track Area	Mandatory elective for all UG students (one perspective must be chosen)

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring/Fall)	1 semester
☒ None	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
17.50 h	10 h	10 h	25 h	62.75 h

Recommendations for Preparation

None

Content and Educational Aims

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.

Intended Learning Outcomes

Students acquire transferable and key skills in this module.

By the end of this module, the students will be able to

1. have a clear understanding of the history of causal thinking.
2. form a critical understanding of the key debates and controversies surrounding the idea of causality.
3. recognize and apply probabilistic causal models.
4. explain how understanding of causality differs among different disciplines.
5. 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.

Usability and Relationship to other Modules

Examination Type: Module Examination

Assessment: Written examination

Duration/Length: 60 min
Weight: 100 %

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination has to be passed with at least 45%

8.2.5 Argumentation, Data Visualization and Communication (perspective I)

Module Name	**Module Code**	**Level (type)**	**CP**
Argumentation, Data Visualization and Communication (perspective I)	CTNS-NSK-07	Constructor Track	5

Module Components

Number	Name	Type	CP
CTNS-07	Argumentation, Data Visualization and Communication (perspective I)	Lecture (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Jules Coleman, Prof Dr. Arvid Kappas	• CONSTRUCTOR Track Area	Mandatory elective for all UG students (one perspective must be chosen)

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring)	1 semester
☒ None	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	20 h	20 h	70 h	125 h

Recommendations for Preparation

None

Content and Educational Aims

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

Students acquire transferable and key skills in this module.

By the end of this module, the students will be able to

1. Distinguish among different modalities of argument, e.g. legal arguments, vs. scientific ones.
2. Construct arguments using tools of data visualization.
3. 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.

Usability and Relationship to other Modules

Examination Type: Module Examination

Assessment Type: Written Examination

Duration/Length: 120 (min)
Weight: 100%

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination has to be passed with at least 45%

8.2.6 Argumentation, Data Visualization and Communication (perspective II)

Module Name	**Module Code**	**Level (type)**	**CP**
Argumentation, Data Visualization and Communication (perspective II)	CTNS-NSK-08	Constructor Track	5

Module Components

Number	Name	Type	CP
CTNS-08	Argumentation, Data Visualization and Communication	Lecture (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Jules Coleman, Prof Dr. Arvid Kappas	• CONSTRUCTOR Track Area	Mandatory elective for all UG students (one perspective must be chosen)

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Fall)	1 semester
☒ None	☒ None	• ability and openness to engage in interactions • media literacy, critical thinking and a proficient handling of data sources • own research in academic literature		

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	20h	20h	50 h	125 h

Recommendations for Preparation

None

Content and Educational Aims

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.

Intended Learning Outcomes

Upon completion of this module, students will be able to

1. Analyze communication processes in formal and informal contexts.
2. Identify challenges and failures in communication.
3. Design communications to achieve specified goals to specific target groups.
4. Understand the principles of argumentation theory.
5. 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

Usability and Relationship to other Modules

Examination Type: Module Examination

Assessment Type: Digital submission of asynchronous presentation, including reflection

Duration/Length: Asynchronous/Digital submission

Weight: 100%

Scope: All intended learning outcomes of the module

Module achievement: 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

Completion: To pass this module, the examination has to be passed with at least 45%

8.2.7 Agency, Leadership, and Accountability

Module Name	**Module Code**	**Level (type)**	**CP**
Agency, Leadership, and Accountability	CTNS-NSK-09	Constructor Track	5

Module Components

Number	Name	Type	CP
CTNS-09	Agency, Leadership, and Accountability	Lecture (online)	5

Module Coordinator	**Program Affiliation**	**Mandatory Status**
Prof. Dr. Jules Coleman	• CONSTRUCTOR Track Area	Mandatory for ACS Mandatory elective for all other UG study programs

Entry Requirements			**Frequency**	**Duration**
Pre-requisites	Co-requisites	Knowledge, Abilities, or Skills	Annually (Spring)	1 semester
☒ None	☒ None			

Student Workload

Asynchronous Self Study	Interactive Learning	Exam Preparation	Independent Study	Hours Total
35 h	20 h	20 h	50 h	125 h

Recommendations for Preparation

None

Content and Educational Aims

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

Students acquire transferable and key skills in this module.

By the end of this module, the students will be able to

1. 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 how science is an economic sector, populated by large powerful organizations that set norms, fund research agendas.
3. 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 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).

Usability and Relationship to other Modules

Examination Type: Module Examination

Assessment Type: Written examination

Duration/Length: 120 min
Weight: 100%

Scope: All intended learning outcomes of the module

Completion: To pass this module, the examination has to be passed with at least 45%

9 Appendix

9.1 Intended Learning Outcomes Assessment Matrix

Applied Computer Science (BSc.)					Introduction to Computer Science	Programming in C and C++	Introduction to Data Science	Calculus and Elements of Linear Algebra I	Algorithms and Data Structures	Introduction to Cyber Physical Systems	Software Design and Prototyping	Calculus and Elements of Linear Algebra II	Distributed Development	Databases and Web Services	Operating Systems	Data Analytics and Modeling	Probability and Random Processes	Software Engineering	Artificial Intelligence	Machine Learning	Internship/Start-Up	ACS Specialization Modules	Collaborative Software Project	Management Electives	Project	CT New Skills	Bachelor Thesis
Semester					1	1	1	1	2	2	2	2	1-2	3	3	3	3	4	4	4	5	5	5	4	5	3-6	4
Mandatory/ optional					m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	me	m	me	m	m	m
Credits					7.5	7.5	7.5	5	7.5	7.5	7.5	5	5	7.5	7.5	7.5	5	7.5	7.5	7.5	15	15	5	10	5	15	30
	Competencies*																										
Program Learning Outcomes	**A**	**E**	**P**	**S**																							
acquire Applied Computer Science knowledge in an independent, self-governed way	x	x	x		x	x		x	x	x	x			x	x			x	x	x	x	x	x		x		x
Work in teams distributed around the globe to analyze complex problems, to evaluate them, and to derive solutions	x	x			x	x	x	x	x	x	x		x	x	x	x		x	x	x		x			x	x	x
Comprehend the processes and tools of Software Engineering for collaborative, remote software and systems development	x	x				x					x		x			x		x					x		x		
Program software in C/C++ and understand algorithms;	x	x				x			x	x								x	x						x		x
Be able to use libraries and to generate software in core Computer Science areas	x	x	x						x	x				x	x	x			x	x					x		x
Apply suited mathematical methods	x	x	x				x	x		x		x				x	x			x					x		x
Understand operating systems, databases, and web services	x	x	x							x				x	x												
Comprehend methods from Artificial Intelligence and Machine Learning	x	x	x	x			x									x			x	x							
Understand the relation between software and its links to the physical world	x	x					x			x									x								
analyze data and to extract insights from it	x	x	x	x			x									x				x			x				
apply the acquired Software Engineering skills and Computer Science knowledge in collaborative, remote projects	x	x	x	x		x	x		x	x	x		x	x	x	x		x	x	x		x	x		x		
Use academic or scientific methods as appropriate in the field of Applied Computer Science such as defining research questions, justifying methods, collecting, assessing and interpreting relevant information, and drawing scientifically-founded conclusions that consider social, scientific and ethical insights;	x	x	x	x			x							x	x	x			x	x	x	x	x		x		x
Develop and advance solutions to problems and arguments in their subject area and defend these in discussions with specialists and non-specialists;		x	x	x						x			x					x			x	x	x	x	x	x	x
Engage ethically with academic, professional and wider communities and to actively contribute to a sustainable future, reflecting and respecting different views;		x	x	x			x						x			x										x	x
Take responsibility for their own learning, personal and professional development and role in society, evaluating critical feedback and self-analysis;		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 their knowledge and understanding to a professional context;	x	x	x								x		x							x		x	x	x			x
Take on responsibility in a diverse team;		x	x	x							x		x					x							x		
Adhere to and defend ethical, scientific and professional standards.		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	
Term paper																								x			
Essay																											
Project report																					x		x		x		
Poster presentation																											
Laboratory report																											
Program Code																											
Oral examination																											
Presentation																								x		x	x
Practical assignment						x				x			x							x							
Project assessment											x			x				x	x			x					
Portfolio Assessments																											
Bachelor Thesis																											x
Module Achievements							x								x	x											

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

Figure 3: ILO-Assessment Matrix