Module Handbook for the Bachelor's Degree in Computer Science

For the 2021 examination regulations

Institute of Computer Science
of the Faculty of Mathematics and Natural Sciences
at the Heinrich Heine University

Published by the
Committee for the Bachelor Examination
in Computer Science

Updated on July 22, 2026

Foreword

The module handbook is intended to provide orientation for the Bachelor's degree in Computer Science. In particular, it should make it easier to choose courses and support the organization of the course.

The module handbook lists the usual courses. However, it is not a complete, exhaustive or definitive listing. Rather, the module handbook is continuously updated and thus reflects the development in research and teaching at the Institute of Computer Science.

In the section on participation requirements, we distinguish between formal and content-related requirements. If you do not meet the formal requirements, you may not register for this module. Modules with content requirements require certain knowledge. If you do not have this knowledge, you will have to acquire it yourself. In general, we recommend completing the relevant modules.

A language of instruction (German or English) is specified for all modules. This is the language in which communication typically takes place in teaching situations. It is possible for a module to be taught in the other of the two languages in individual years. This will be announced via HIS/LSF before the start of the course. Furthermore, the language of the specialist literature used does not have to correspond to the language of instruction. The referenced literature is often in English.

Please note, however, that the examination regulations of the BSc Computer Science programme are decisive for all questions relating to studies and examinations.

Düsseldorf, July 22, 2026

The Committee for the Bachelor Examination in Computer Science

Table of Contents

Foreword . . . 2
Qualification goals of the B.Sc. in Computer Science . . . 5
Scientific qualifications and qualifications for employment . . . 5
Personal development . . . 5
Course Plans . . . 6
Compulsory Modules in Computer Science . . . 7
Algorithms and Data Structures . . . 8
C Programming for Algorithms and Data Structures . . . 10
Data Science . . . 12
Databases: An Introduction . . . 14
Introduction to Computer Networks . . . 16
Programming Project 1 . . . 18
Programming Project 2 . . . 20
Programming . . . 22
Computer Architecture . . . 24
Theoretical Computer Science . . . 26
Scientific Methods . . . 28
Compulsory Modules in Mathematics . . . 30
Mathematics for Computer Science 1 . . . 31
Mathematics for Computer Science 2 . . . 33
Mathematics for Computer Science 3 . . . 35
Courses for elective areas . . . 37
Courses from other programs or institutes . . . 37
Algorithms in Bioinformatics . . . 38
Computational Geometry . . . 40
Computational Complexity Theory . . . 42
Operating Systems and System Programming . . . 44
Competitive Programming A . . . 46
Competitive Programming B . . . 48
Compiler Construction . . . 50
Computer-aided reasoning . . . 52
Data Science 2 . . . 54
Databases: Further Concepts . . . 56
Data Visualization . . . 58
Digital Innovation: From Idea to Impact . . . 60
Introduction to Algorithmic Game Theory . . . 62
Introduction to Deep Learning . . . 64
Introduction to Functional Programming . . . 66
Introduction to Scientific Computer Science . . . 68
Introduction to Python . . . 70
Introduction to Randomized Algorithms . . . 72
Embedded Systems . . . 74
Case study seminar: Digital Future Challenge . . . 76

Algorithms for Graphs 1 . . . 78
Foundations of Distributed Systems . . . 80
Introduction to Logic Programming . . . 82
Introduction to Modelling metabolic networks . . . 84
Introduction to NLP . . . 86
Combinatorial Algorithms for Clustering Problems . . . 88
Cryptocomplexity 1 . . . 90
Logic for Computer Science . . . 92
Patterns in nature: theoretical background and algorithms . . . 94
Preference Aggregation by Voting: Algorithmics and Complexity . . . 96
Python for NLP . . . 98
From Circuits to Software . . . 100
Seminar . . . 102
Compulsory Seminar . . . 103
Bachelor Thesis . . . 105
Thesis . . . 106
Final Seminar . . . 107
Modules no longer offered . . . 108
Applied Algorithmics . . . 109
Bachelor's Seminar: Introduction to blockchain technology . . . 111
Bachelor's-Seminar: Programming Languages . . . 113
Bachelor's-Seminar: Introduction to Artificial Intelligence . . . 115
Data Visualization . . . 117
Collective Decisions . . . 119
Matching . . . 121
Patterns in nature: theoretical background and algorithms . . . 123
Randomized Algorithms und Analysis . . . 125
Statistical Data Analysis . . . 127

Qualification goals of the B.Sc. in Computer Science

Upon completion of their studies, students will have achieved the following qualification goals.

Scientific qualifications and qualifications for employment

Graduates

- are able to compare and apply methods and procedures of computer science, such as systematic problem definitions, algorithmic problem solving, logical proof procedures or evaluation procedures.
- can develop applications using state-of-the art software technologies. This includes conception, implementation and testing.
- are familiar with a logical, analytical and systemic approach to thinking that enables them to analyse and solve problems in computer science, for example in fields of data science and machine learning.
- are able to apply learned knowledge and research methods for solving research problems and assessing the usefulness of the results, taking into account principles of good scientific practice.

Personal development

Graduates

- can assess their own skills with regard to aspects such as data analytics, software development or communication and already have ideas for their further development. They can independently acquire new specialised knowledge.
- can take responsibility for themselves and their tasks within a group.
- know a range of professional images (e.g. a data scientist or a software engineer) and have adopted their own.
- are able to communicate and discuss with others computer science problems and appropriate solutions, which they practiced in exercises and seminars
- are aware of basic ethical questions and challenges from the perspective of computer science as well as the social, cultural and political significance of their discipline.

Course Plans

You can find course plans on the German website of the Institute of Computer Science. These are called **Studienverlaufspläne** and serve as orientation and help you to plan your studies. There are different versions of the study plan:

- Full-time, with mathematics for computer science
- Full-time, with mathematics of mathematics
- Part-time, with mathematics for computer science

If you need help planning your studies, please contact the student advisory service at studieninteressierte@cs.uni-duesseldorf.de.

Compulsory Modules in Computer Science

Algorithms and Data Structures

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

The module conveys fundamental analysis techniques for algorithms. Important data structures and algorithms are explained and analyzed with respect to runtime and correctness. In addition, initial examples and concepts regarding the limits of algorithmic efficiency are introduced. In particular, the following topics are discussed.

- Analysis techniques: pseudocode, proving correctness of iterative algorithms, formulating and proving loop invariants, analysing of best-case and worst-case running time, asymptotic running time analysis, Landau notation
- Devolping new algorithms and design patterns: recursive running time analysis, correctness of recursive algorithms, divide and conquer, greedy algorithms and exchange arguments, dynamic programming
- Analysis of data structures: red-black trees, priority queues, hashing, union-find, heaps, heapsort
- Analysis of graph algorithms: depth-first search, topological sorting, breadth-first search, algorithms for shortest s–t paths as well as for computing all shortest s–t paths, union-find
- Limits of efficiency: lower bound for comparison-based sorting algorithms, introduction to complexity theory

Learning Outcomes

After successfully completing the courses of this module, students will be able to:

- reproduce the contents and interrelations of the lecture in their own words,
- adapt the content of the lecture to new situations,
- answer related deeper questions regarding the presented use cases of datastructures and algorithms, proofs, proof techniques and implementation decisions,
- formalize problem settings and algorithms given in natural language,
- design correct and efficient algorithms for new problems,
- purposefully apply, combine or adapt known algorithms and data structures, and
- apply fundamental analysis techniques of algorithmics to analyze the correctness and running time of known and new algorithms.

In order to reach the aforementioned topic specific outcomes, the students will develop their ability to think analytically and independently, to argue logically, and to write structured texts following scientific standards.

Bibliography

- eigenes Skript in der im WS2024/25 überarbeiteten Version
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein: Algorithmen – Eine Einführung. De Gruyter Oldenbourg. 2017. 4. Auflage.
- Robert Sedgewick, Kevin Wayne: Algorithmen. Pearson Studium. 2014. 4. Auflage.
- Thomas Ottmann, Peter Widmayer: Algorithmen und Datenstrukturen. Spektrum. 2012. 5. Auflage.
- Jon Kleinberg, Eva Tardos: Algorithm Design. Addison Wesley. 2006.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- Formal: Successfull completion of the module *Programming* or of the modul *Mathematics for Computer Science 1* or of the modul *Calculus I*.
- Contentual: Contents of modules *Programming* and *Mathematics for Computer Science 1*

Conditions for awarding credit points

- passing a practice test and achieving the required points in the exercises (admissions acquired before the winter term 2024/25 are no longer valid),
- written exam (regularly 90 minutes) or oral exam at the end of the term

Responsible persons

Dr. Daniel Schmidt, Prof. Dr. Gunnar W. Klau, Prof. Dr. Melanie Schmidt

Last updated 2025-08-28

C Programming for Algorithms and Data Structures

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This module builds on the module *Algorithms and Data Structures* and is dedicated to the implementation of this module's concepts. More precisely, it treats the following topics:

- Introduction to C
- Fundamental data structures (stacks, doubly linked lists, trees)
- Search trees
- Heaps
- Quicksort
- Hashing
- Dynamic graph data structures and Dijkstra's algorithm

Learning Outcomes

After completing the course, students are able to

- implement central algorithms and data structures in the programming language C with dynamic memory management and
- use typical tools for programming in C (memory management, debugging, build processes)

Bibliography

- R. Sedgewick: Algorithmen in C. Pearson. 2005.
- R. Seacord: Effective C. No Starch Press. 2020.
- G. Gonzales-Morris und I. Horton: Beginning C: From Beginner to Pro. Apress. 2020.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Hardwarenahe Programmierung PO 2013, 2016

Prerequisites

- Formal: Successfull completion of the module *Programming*
- Contentual: Contents of modules *Algorithms and Data Structures* and *Programming*

Conditions for awarding credit points

- active and successful participation in the tutorials (weekly programming tasks and semester-accompanying oral tests, registration modalities, passing criteria and dates will be announced at the beginning of the course)
- written exam (regularly 90 minutes)

Responsible persons

Dr. Daniel Schmidt, Prof. Dr. Gunnar W. Klau, Prof. Dr. Melanie Schmidt

Last updated 2025-06-13

Data Science

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

Data Science is the application of statistical methods and methods of machine learning to data of any kind, using a computer to model systems and predict behaviour.

- probability theory (discrete and continuous distributions, Bayes' theorem, independence, normal distribution, multivariate distributions, transformations)
- machine learning (data and models, estimator theory, classification, regression, dimensionality reduction, cluster analysis)
- python packages for data science (numpy, matplotlib, pandas, scikit-learn)

Learning Outcomes

After completing the course, students are able to

- perform exploratory data analysis using the programming language Python,
- transform data for further processing,
- ask research questions about a dataset,
- select parts of data for further processing,
- merge suitable datasets,
- describe insights from data using statistical terminology and visualize insights from data using the packages pandas and matplotlib,
- consider data protection problems about processing data,
- consider ethical questions about processing data,
- construct statistical models from data using the package scikit-learn,
- evaluate the performance of models and
- use statistical models for classification and prediction of future data.

Bibliography

- Georgii: Stochastik: Einführung In Die Wahrscheinlichkeitstheorie Und Statistik. De Gruyter. Berlin, 2015. 5. Auflage
- VanderPlas: Python Data Science Handbook. O'Reilly Media, Inc. Sebastopol, 2016. 1st edition
- Grus: Data Science from Scratch: First Principles with Python. O'Reilly UK Ltd. UK, 2019. 2nd edition
- Deisenroth et. al.: Mathematics for Machine Learning. CUP. Cambridge, 2020. 1st edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules *Programming* and *Mathematics for Computer Science 1–3*
- Formal: Passed the module *Programming* and passed 2 out of 3 modules from *Mathematics for Computer Science 1–3*. Instead of 2 out of 3 modules from *Mathematics for Computer Science 1–3*, it is also sufficient to have passed *Linear Algebra I* and *Analysis I*.

Conditions for awarding credit points

- active and successful participation in the theoretical and practical exercises
- passing the written exam (usually 90 minutes)

Responsible persons

Dr. Konrad Völkel, Prof. Dr. Milica Gašic

Last updated 2024-10-10

Databases: An Introduction

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

In this module basic theoretic and practical knowledge about databases is presented. The different phases in database design as well as SQL queries and the required formal background are to the fore.

- architektures of database systems; 3-level schema architecture
- tasks of a database system
- data(base) models; in particular the relational model
- conceptual and logical database design (phases model, ER model, transformation into the relational model); normalisation (functional dependencies, normal forms, synthesis algorithm)
- query languages for relational databases and their foundations; relational algebra, relational calculus, SQL
- concurrency control: problems of multiuser operation, transactions and schedules, serializability

Learning Outcomes

After completing the course, students are able to

- name and explain the tasks of database systems,
- design databases on their own,
- formulate database queries using different query languages (in particular, the relational algebra and SQL) and
- test whether schedules for sets of transactions are serializible.

Bibliography

- G. Saake, K.-U. Sattler, A. Heuer: Datenbanken – Konzepte und Sprachen. mitp Verlag. 2018. 6. Auflage
- Kemper, A. Eickler: Datenbanksysteme – Eine Einführung. Oldenbourg Verlag. 2015. 10. Auflage
- G. Vossen: Datenmodelle, Datenbanksprachen und Datenbankmanagementsysteme. Oldenbourg Verlag. 2008.
- R. Elmasri, S.B. Navath: Fundamentals of Database Systems. Pearson. 2016. 7th edition
- H. Garcia-Molina, J.D. Ullman, J. Widom: Database Systems: The Complete Book. Pearson. 2009. 2nd edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- None

Conditions for awarding credit points

- active and successful participation in the exercises (usually with weekly homeworks)
- written exam (usually 60 minutes)

Responsible persons

Prof. Dr. Stefan Conrad, Dr. Leonie Selbach

Last updated 2023-03-23

Introduction to Computer Networks

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

The module *Introduction to Computer Networks* targets students who want to understand the technical design principles of computer networks. Fundamental problems arising in the design of computer networks and how they have been addressed in the context of the internet are being discussed. The module aims to teach a solid foundation of computer network technology as well as practical capabilities.

- Introduction and overview
- Application Layer (World Wide Web/HTTP, File Transfer/FTP, E-Mail/SMTP, Domain Name System/DNS, socket programming with UDP and TCP)
- Transport Layer (addressing, reliable data transfer/RDT, congestion control, UDP, TCP)
- Network Layer (virtual connections and datagram networks, design principles of routers, adressing/DHCP and NAT, internet protocol/IP, ICMP, link state routing and distance vector routing, RIP, OSPF, BGP)
- Link Layer (frames, error detection and correction, media access in local networks, addressing/ARP, Ethernet, hubs, switches, PPP, IP over ATM und MPLS, applications)

Learning Outcomes

After completing the course, students are able to

- explain the layered structure of the internet and apply the important protocols in all layers,
- compute the performance of the protocols in simple networks,
- explain address mechanisms in the different layers and discuss their interactions,
- analyse simple local networks and design and implement suitable network architectures and protocols for given applications,
- apply fundamental methods to optimise the performance of complex networks and
- distinguish various types of transmission errors as well as suggest ways to discover and fix transmission errors in the various network layers.

Bibliography

- James F. Kurose and Keith W. Ross: Computer Networking – A Top-Down Approach Featuring the Internet. Pearson, 2020. 8th Edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- Homework exercises (50% needed for exam admission)
- Written final exam (typically 90 minutes)

Responsible persons

apl. Prof. Dr. Rudolf Fleischer, Prof. Dr. Martin Mauve

Last updated 2025-12-18

Programming Project 1

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	66 hours	234 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Lab (1 HPW)	every summer semester	Bachelor Computer Science	German

Remarks

- The specification of the HPW for the contact time is not exact but rounded.
- The lecture usese the flipped classroom method.

Content

The goal of the programming projects is to enable teams of students to develop a larger, webbased information system of high quality in Java. The first module contains:

- Development tools (IDE, build tools, version control systems)
- Automatic testing and test driven development
- Code quality and code smells
- Principles and practices to modularize larger systems using object-oriented programming (SOLID principles, Information Hiding, Coupling and Cohesion, Law of Demeter, Polymorphism, Dependency Injection)
- Basics of domain-driven design (ubiquotous language and tactical design)
- Systematic debugging
- Advanced Java concepts (Streams, Records, Optional, DateTime, …)

Learning Outcomes

After completing the course, students are able to:

- develop larger, maintainable systems based on a problem statement
- analyze the quality of given code and discuss the effect on its maintainability
- improve the quality and maintainabiliy of given code
- develop code driven by tests
- use Java language features idiomatically
- systematically find and fix bugs
- use common development tools (IDE, build tools, version control systems)

Bibliography

- Lecture Notes

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics

- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- Formal: Successful completion of the module *Programming*

Conditions for awarding credit points

- Active and successful participation in exercises
- Written exam (90 min)

Responsible persons

Dr. Jens Bendisposto, Dr. Markus Brenneis

Last updated 2024-07-24

Programming Project 2

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	124 hours	176 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Lab (1 HPW) Block Lab (4 HPW)	every winter semester	Bachelor Computer Science	German

Remarks

- The specification of the HPW for the contact time is not exact but rounded.
- The lecture usese the flipped classroom method.

Content

The goal of the programming projects is to enable teams of students to develop a larger, web-based information system of high quality in Java. The second module contains:

- Webdevelopment (HTTP, HTML, Accessibility, Authentification and Authorization, Security, REST, Servlets, Spring Web)
- Accessing Databases (JDBC, Spring Data)
- Software Architecture and Architectural pattern
- Integration testing and Archictecture testing (ArchUnit)
- Software development processes
- Documentation of Architecture (arc42, UML)

Learning Outcomes

After completing the course, students are able to:

- implement and secure web applications
- ensure accessibility of web applications
- develop informations systems that use databases
- describe and apply architectural patterns
- write integration test for web-based information systems
- write tests to ensure that architectural decision are respected
- describe and apply common software development processes
- write documentation of a system’s architecture

Bibliography

- Lecture Notes

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013

Prerequisites

- Formal: Successful completion of the module *Programming Project 1*

Conditions for awarding credit points

- Active and successful participation in exercises
- Written exam (90 min)

Responsible persons

Dr. Jens Bendisposto, Dr. Leonie Selbach

Last updated 2024-07-24

Programming

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	120 hours	180 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW) Exercise Course (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

This module teaches the basics of imperative, object-oriented programming using the Java programming language as an example. Aspects of algorithms and data structures are dealt with in an introductory manner. o previous programming experience is required.

- primitive data types and variables
- control structures
- arrays
- standard input and output, files
- program structures in memory (heap, stack)
- recursion
- concepts of object-oriented programming (classes, objects, polymorphism, interfaces, inheritance)
- error handling
- searching and sorting (linear search, binary search, insertion sort, mergesort)
- dynamic data structures (lists, binary search trees, hashing)
- additional practical classes from the Java Standard Library (Collection, Stream, functional interfaces) and development tools (unit tests, version control, build systems, IDEs)

Learning Outcomes

After completing the course, students are able to

- explain given Java program code,
- translate descriptions of iterative/recursive algorithms into structured code and design their own simple algorithms,
- use standard input/output and files to read and output textual data,
- write object-oriented programs that model real-world objects,
- explain and apply advantages of polymorphism,
- expound on the advantages and disadvantages of different data structures (arrays, singly linked lists, binary search trees, hash sets),
- explain algorithms for different data structures (arrays, singly linked lists, binary search trees, hash sets) and implement them,
- fix compile-time errors and handle run-time errors appropriately,
- use known methods from the Collection Framework and new methods with the help of the documentation, and
- explain the purpose of tests, version control systems, build systems, and IDEs in software development.

Bibliography

- R. Schiedermeier: Programmieren mit Java. Pearson Studium. München, 2010. 2. aktualisierte Ausgabe
- C. Ullenboom: Java ist auch eine Insel. Rheinwerk Computing. Bonn, 2021. 16. Auflage
- R. Sedgewick & K. Wayne: Introduction to Programming in Java. Addison-Wesley, United States, 2007. 1st Edition

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module I: Computer Science Bachelor study programme Computer Linguistics

Prerequisites

- None

Conditions for awarding credit points

- Passing the admission test (dates, pass mark and contents will be announced at the beginning of the course)
- Written exam (120 min)

Responsible persons

Dr. Markus Brenneis, Prof. Dr. Michael Schöttner, Prof. Dr. Johannes Stegmaier

Last updated 2026-04-09

Computer Architecture

ECTS credits	Total hours	Contact hours	Self-study hours
7 CP	210 hours	90 hours	120 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (3 HPW) Tutorial (2 HPW) Exercise Course (1 HPW)	every winter semester	Bachelor Computer Science	German

Content

This module provides a basic understanding of the structure and functioning of modern computers. The following topics will be addressed:

- Data representation (ASCII, Unicode, b-adic numbers, two's complement, non integer binary numbers, floating-point numbers according to IEEE 754),
- Two-element boolean algebra (definition according to Huntington, further rules for calculation, functionally complete sets , truth tables, disjunctive and conjunctive normal form, KV diagrams, disjunctive and conjunctive minimal form, don't care assignments),
- Digital logic (gates, decoders and encoders, multiplexers and demultiplexers, shifters, half and full adders, clock signals, flip-flops and latches (SR-, D- each), circuits made from these elements, hazard errors),
- Error detection and correction (duplication, parity bit, two-dimensional parity, Hamming distance, Hamming code),
- Introduction of Java bytecode as an example of assembly language (translation of simple Java program snippets into Java bytecode, design of simple program snippets directly in Java bytecode),
- Microarchitecture (implementation of a processor for the execution of a subset of the Java byte code, digital logic of the processor, microprogram for the processor, interpretation of Java byte code by the processor), and
- Methods for accelerating a processor (reduction of work cycles, instruction prefetching, pipelining, jump prediction, caching).

Learning Outcomes

After completing the course, students are able to

- describe the different layers of a computer architecture, taking into account the connections between them,
- represent numbers in the above representations and convert between them,
- specify how characters and character strings are represented in the computer,
- specify and evaluate Boolean functions in various forms (formula, truth table, KV diagram), minimize them using the KV diagram and show the equivalence of functions,
- evaluate whether a given circuit fits a Boolean function,
- analyze and create circuits from known components and describe their function,
- draw waveform diagrams and use these or KV diagrams to find and rectify possible hazard errors in digital circuits,
- explain how a CPU/ALU is constructed from elementary digital circuits and how it works,
- calculate the Hamming distance of words and (finite) codes and name the Hamming distance of known codes,
- apply known procedures for error detection and correction and name what kind of errors can be detected or corrected by them,

- analyze programs for the example processor discussed at the binary code, microcode and assembly code levels and describe their function,
- write simple programs, similar to the programs discussed, for the example processor discussed at the levels mentioned,
- name the methods discussed for accelerating a processor and present examples of them and
- discuss the advantages and disadvantages of the cache types discussed and carry out cache procedures on paper, comparing different caches and replacement strategies.

Bibliography

- A. S. Tanenbaum, T. Austin: Structured Computer Organization. Pearson. Boston, 2013. 6th Edition
- Further literature is provided during the course.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Module CL6 Bachelor study programme Computer Linguistics

Prerequisites

- Contentually: It is assumed that the students attend the module *Programming* at the same time or have basic programming knowledge.

Conditions for awarding credit points

- acquiring the admission to the exam by one of two variants (the chosen variant and the number of points to be achieved will be announced at the beginning of the course):
  - Variant 1: sufficient completion of the exercise sheets over the course of the semester
  - Variant 2: (mostly) disjoint partition of topics on two admission parts:
    - Part 1: sufficient completion of the exercise sheets
    - Part 2: pass the admission test
- written exam (exam, usually 90 minutes)

Responsible persons

Prof. Dr. Stefan Conrad, Janine Golov, Prof. Dr. Martin Mauve

Last updated 2026-05-11

Theoretical Computer Science

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	120 hours	180 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW) Exercise Course (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This module provides a basic understanding of the foundations, models, methods, and results of theoretical computer science, in particular introducing into the theory of formal languages and automata, the theory of computability, and the theory of NP-completeness

Formal Languages and Automata

- Foundations (Strings, Languages, and Grammars; the Chomsky Hierarchy)
- Regular Languages (Finite Automata; Regular Expressions; Equation Systems; the Pumping Lemma; Theorem of Myhill and Nerode and Minimal Automata; Closure Properties and Charakterizations of Regular Languages)
- Contextfree Languages (Normal Forms; the Pumping Lemma; Closure Properties of Contextfree Languages; the Algorithm of Cocke, Younger, and Kasami; Push-Down Automata)
- Deterministic Contextfree Languages (Deterministic Push-Down Automata; Application: Syntax Analysis by LL(k)-Parsers)
- Contextsensitive and L_0 Languages (Turingmaschinen; Linear Bounded Automata; Overview)

Computability

- Intuitive Notion of Computability and Church's Thesis
- Turing Computability
- LOOP, WHILE, and GOTO Computability
- Primitive Recursive and Partial Recursive Functions (Primitive Recursive Functions; the Ackermann Function; Total and Partial Recursive Functions; the Main Theorem of Computability Theory)
- Decidability and Enumerability
- Undecidability (Rice's Theorem; Reducibility; Post's Correspondence Problem; Undecidability in the Chomsky Hierarchy; Overview)

NP-Completeness

- Problems in P and NP (Deterministic Polynomial Time; the Satisfiability Problem of Propositional Logic; Nondeterministic Polynomial Time)
- NP-Completeness and Cook's Theorem

Learning Outcomes

After completing the course, students are able to

- classify formal languages in the Chomsky hierarchy,
- transform the considered equivalent automata models into each other into the considered grammars of the respective type,
- provide for a given language a grammar generating it or an automaton of a suitable type (e.g., a finite automaton or a push-down automaton or a linear bounded automaton or a Turing machine) accepting it,

- conversely, determine for a given grammar or automaton the corresponding language,
- give arguments for the inequivalence of considered automata models or grammar types,
- describe how a compiler is built,
- describe the tasks and methods of lexical and syntax analysis,
- discuss the algorithmic decidability of problems,
- give arguments for the undecidability of problems,
- argue that there exist functions that are not computable,
- apply the acquired skills dealing with formal concepts and models and with formal argumentations and proof techniques (such as diagonalization) and
- provide reductions between problems to show their undecidability or NP-completeness.

Bibliography

- Uwe Schöning: Theoretische Informatik kurz gefasst. Spektrum Akademischer Verlag. Heidelberg, 2008. 5. Auflage.
- John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman: Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie. Pearson Studium. München, 2002. 2. Auflage.
- Klaus W. Wagner: Theoretische Informatik. Eine kompakte Einführung. Springer-Verlag. Berlin, Heidelberg, 2003. 2. Auflage.
- Further literature is provided during the course.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Compulsory Area Bachelor study programme PO 2016, 2013
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contents of module *Mathematics for Computer Science 1* (or, alternatively, *Linear Algebra I*)

Conditions for awarding credit points

- passing a practice test resp. active and successful participation in the exercises (exact terms will be announced at the beginning of the course)
- written exam (exam, usually 90 minutes)

Responsible persons

Mareike Mutz, Prof. Dr. Michael Leuschel, Prof. Dr. Jörg Rothe

Last updated 2026-01-22

Scientific Methods

ECTS credits	Total hours	Contact hours	Self-study hours
3 CP	90 hours	30 hours	60 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

Part I: Practical methods

- Structure of scientific texts: how is science communicated?
- Efficient reading and peer review: How do you provide good feedback?
- Delivering effective presentations
- High-quality academic illustrations
- Academic research
- Plagiarism and how to appropriately cite existing knowledge
- Structural formatting of documents: LaTeX and other word processing programmes
- Tools for academic creativity

Part II: Artificial Intelligence in Higher Education and Academic Work

- Introduction to the technology behind AI chatbots (Large Language Models)
- AI as a tool for researching and understanding academic texts
- AI as a tool for writing texts
- AI as a tool for programming

Part III: Ethical and Societal Aspects

- ethical issues and challenges in computer science
- the societal, cultural and political significance of computer science
- risks associated with artificial intelligence

Learning Outcomes

After completing the course, students are able to

- describe the basic methods of science, both on a practical and on an abstract level,
- describe the process of scientific publishing and the central role of citations in scientific communication,
- describe the peer review process and the appropriate structure of reviews,
- describe the requirements of good presentations,
- research, identify and read scientific literature,
- design, layout, and summarize appropriately structured scientific texts and graphics,
- assess whether given texts and graphics meet academic standards,
- identify ethical questions and challenges in computer science and describe how these can be approached,
- assess the societal, cultural, and political significance of their academic discipline,
- identify and categorise risks associated with AI,
- identify the AI tools presented, use them appropriately and recognise the contexts in which they are best applied,
- format academic texts structurally using the programs introduced in the lectures,
- identify the fundamentals of the technology underpinning large language models,
- justify the position of formal proofs in computer science,

- explain the central role of hypotheses and their testing in science and
- appreciate the role of creativity in science and distinguish the process of scientific work from scientific communication.

Bibliography

- Alan Chalmers: What is this thing called science? University of Queensland Press, Open University Press. 2013. 4th edition
- Michael Strevens: The Knowledge Machine: How Irrationality Created Modern Science. Liveright Publishing Corporation. 2021
- Itai Yanai & Martin Lercher: Night Science Collection. https://www.biomedcentral.com/collections/night-science
- Further literature is provided during the course.

Module compatibility

- Compulsory Area Bachelor study programme PO 2021
- Praxis- und Berufsorientierung für die Bachelor PO 2013 und PO 2016

Prerequisites

- None

Conditions for awarding credit points

- active and successfull participation in the exercises
- passing the written exam (exam, usually 60 minutes)

Responsible persons

Mareike Mutz, Prof. Dr. Martin Lercher

Last updated 2026-07-21

Compulsory Modules in Mathematics

In the Bachelor's program in Computer Science at HHU, you have two options to cover the field of Mathematics. You can either choose the path *Mathematics for Computer Science* or the path *Mathematics of Mathematics*. The modules of the *Mathematics for Computer Science* path can be found in this module handbook on the following pages. The modules of the path *Mathematics of Mathematics* can be found in the Module Handbook of Mathematics. Please refer to this in particular for the content, learning objectives and prerequisites of the modules.

The following modules make up this area:

- *Analysis I* (mandatory)
- *Lineare Algebra I* (mandatory)
- either *Stochastik* or *Numerik I*

It is your decision wether to cover the Angewandte Mathematik module via *Numerik I* or *Stochastik*. Please note that for *Numerik I*, other modules in mathematics are recommended as prerequisites.

Mathematics for Computer Science 1

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	120 hours	180 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW) Exercise Course (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

- basic mathematics: sets, relations, mappings, countability and equinumerosity, propositional calculus
- real numbers: field axioms and ordering of real numbers, absolute value, induction, real functions
- euklidian space: lineare equations, Sscalar product, lineare mappings, determinant
- sequences, completeness of real numbers, series, exponential series and function
- b-adic rationals, asymptotic notation
- algebraic structures: groups, fields

Learning Outcomes

After completing the course, students are able to

- apply the basic mathematical concepts,
- write short proofs using the learned methods,
- solve systems of linear equations,
- argue with the central analysis notion of limits,
- handle selected techniques applied to sequences and series in computer science and
- solve exercises and present solutions applying the necessary mathematical precision and language.

Bibliography

- B. Kreußler, G. Pfister: Mathematik für Informatiker. Springer. Berlin, Heidelberg, 2009.
- G. Teschl, S. Teschl: Mathematik für Informatiker I. Springer. Berlin, Heidelberg, 2013. 4. Auflage
- G. Teschl, S. Teschl: Mathematik für Informatiker II. Springer. Berlin, Heidelberg, 2014. 3. Auflage
- P. Hartmann: Mathematik für Informatiker. Springer, Wiesbaden, 2020. 7. Auflage
- W. Struckmann, D. Wätjen: Mathematik für Informatiker, Springer, Berlin, Heidelberg, 2016. 2. Auflage

Module compatibility

- Compulsory Area Bachelor study programme PO 2021

Prerequisites

- None

Conditions for awarding credit points

- active and successful participation in exercise class
- written exam (120 minutes)

Responsible persons

Dr. Nadja Hempel (Valentin), Dr. Andreas Rätz

Last updated 2024-07-24

Mathematics for Computer Science 2

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	120 hours	180 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW) Exercise Course (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

- differential calculus: limits of functions and Continuity, elementary funktions, maxima und suprema, differentiable functions, local extrema and mean value theorems, de l'Hospital's rules
- linear algebra: abstract vector spaces, subspaces, linear mappings, basis, dimension, endomorphisms, eigenvalues and eigenvectors
- coding theory: Hamming-metric, linear codes, perfect codes

Learning Outcomes

After completing the course, students are able to

- apply further motions of limits for functions,
- handle concepts and rules of differential calculus,
- be able to investigate the asymptotic behaviour of functions applying various techniques,
- apply the central concepts of linear algebra,
- solve linear algebra problems by solving systems of linear equations,
- determine the representing matrix of a linear mapping and to compute the transition matrix associated with a change of basis for a vector space,
- calculate Eigenvalues and Eigenvectors of matrices and endomorphisms as well as the diagonal form of diagonalizable endomorphisms/matrices and
- apply basic concepts of linear algebra to questions arising in computer science.

Bibliography

- B. Kreußler, G. Pfister: Mathematik für Informatiker. Springer. Berlin, Heidelberg, 2009.
- G. Teschl, S. Teschl: Mathematik für Informatiker I. Springer. Berlin, Heidelberg, 2013. 4. Auflage
- G. Teschl, S. Teschl: Mathematik für Informatiker II. Springer. Berlin, Heidelberg, 2014. 3. Auflage
- P. Hartmann: Mathematik für Informatiker. Springer, Wiesbaden, 2020. 7. Auflage
- W. Struckmann, D. Wätjen: Mathematik für Informatiker, Springer, Berlin, Heidelberg, 2016. 2. Auflage

Module compatibility

- Compulsory Area Bachelor study programme PO 2021

Prerequisites

- Contentual: Contents of module *Mathematics for Computer Science 1*

Conditions for awarding credit points

- active and successful participation in exercise class
- written exam (120 minutes)

Responsible persons

Dr. Nadja Hempel (Valentin), Dr. Andreas Rätz

Last updated 2024-07-24

Mathematics for Computer Science 3

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	120 hours	180 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW) Exercise Course (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

- integral calculus: Riemann‘s integral, fundamental theorem of calculus, improper integrals
- Taylor series, power series
- analysis of several variables: differential calculus of several variables, extreme values for functions with several variables
- basic numerics: numerical treatment of systems of linear equations, interpolation, treatment of nonlinear equations (Newton‘s method)
- basic probability theory: sample space, probability, conditional probability, random variable (discrete and continuous), distribution function, expected value, standard deviation, variance, binomial distribution, Poisson distribution, exponential distribution, uniform distribution, Gaussian distribution

Learning Outcomes

After completing the course, students are able to

- handle concepts and rules of integral calculus,
- apply the differential calculus for functions with several variables to precise problems,
- evaluate the discussed numerical techniques with respect to efficiency and feasibility and
- describe the discussed terms of stochastics as well as know and apply the discussed distributions.

Bibliography

- B. Kreußler, G. Pfister: Mathematik für Informatiker. Springer. Berlin, Heidelberg, 2009.
- G. Teschl, S. Teschl: Mathematik für Informatiker I. Springer. Berlin, Heidelberg, 2013. 4. Auflage
- G. Teschl, S. Teschl: Mathematik für Informatiker II. Springer. Berlin, Heidelberg, 2014. 3. Auflage
- P. Hartmann: Mathematik für Informatiker. Springer, Wiesbaden, 2020. 7. Auflage
- W. Struckmann, D. Wätjen: Mathematik für Informatiker, Springer, Berlin, Heidelberg, 2016. 2. Auflage

Module compatibility

- Compulsory Area Bachelor study programme PO 2021

Prerequisites

- Contentual: Contents of modules *Mathematics for Computer Science 1 and 2*

Conditions for awarding credit points

- active and successful participation in exercise class
- written exam (120 minutes)

Responsible persons

Dr. Nadja Hempel (Valentin), Dr. Andreas Rätz

Last updated 2024-07-24

Courses for elective areas

The elective area comprises 45 credit points. The following rules apply to the composition of the modules:

- In order to improve the feasibility of studying the course and to increase the options for students to choose from, modules of different sizes are offered in the bachelor's course in computer science. Modules must be combined in such a way that the total number of credits corresponds to the requirements of the examination regulations. Modules can be freely combined for the elective area.
- 20 CP may come from courses offered in other subjects. A list from which you can see which subjects are recognised is maintained on the website of the degree program.
- 5 CP may be covered by courses offered outside the faculties. These are not included in the overall grade of the bachelor's degree. The corresponding offers include in particular the offers of the student academy (career, soft & study skills, languages).

Up to 30 CP of additional courses can be recognized as part of the Beachlor s program in computer science. These can also come from the courses offered in the master's program in computer science. For the modules offered, please refer to the module handbook for the Master's course, which you can find on the computer science website. Participation in these modules is only permitted if the courses Programming, Computer Architecture, Algorithms and Data Structures and Theoretical Computer Science have been passed.

A module for the bachelor's degree can only be used for the individual supplement in the master's degree in computer science if it has not already been used for a previous bachelor's degree and is approved for this (this is usually not the case after September 1st, 2022).

Courses from other programs or institutes

Some courses from other degree programs and/or institutes can also be taken by students on the Bachelor's degree program in Computer Science.

Please note that courses from the Master's degree prograann of Computer Science (which is not the Master's degree programm AI and Data Science) cannot be used in the Bachelor's degree program in Computer Science.

Master AI and Data Science

The modules listed here can be found in the Module Handbook of the *Master AI and Data Science* program. Please be aware of the contentual prerequisites of the respective modules given in the module handbook. If a module is listed here, this does *not* automatically mean that the module can be completed with the previous knowledge from the compulsory modules of the Bachelor of Computer Science alone.

Title	Responsible Persons	Cycle	ECTS	Lang.
Machine Learning	Prof. Dr. Paul Swoboda	every winter semester	10 CP	EN

Elective subjects outside of computer science

As part of your studies you may, to a limited extent (see examination regulations), also take subjects outside of Computer Science. You can find further information on this on our webpages.

Algorithms in Bioinformatics

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

The module teaches introductory concepts of algorithmic bioinformatics. The students will deepen and apply them in theoretical and practical exercises.

- Biological background
- Exhaustive search: DNA motifs
- Greedy algorithms: genome rearrangements
- Dynamic programming: sequence alignments
- Graph algorithms: assembly
- Combinatorial pattern matching and suffix trees
- Clustering
- Phylogenetic trees and molecular evolution
- Hidden Markov models: CpG islands

Learning Outcomes

After completing the course, students are able to:

- apply the discussed algorithmic design principles, prove correctness and analyze running times,
- differentiate between tractable and intractable algorithmic problems and explain the consequences,
- distinguish different classes of algorithms,
- explain and apply classic bioinformatics algorithms,
- implement many of these algorithms in the programming language Python, and
- select an appropriate algorithm to solve a given task.

Bibliography

- Neil C. Jones, Pavel A. Pevzner: An Introduction to Bioinformatics Algorithms. The MIT Press, 2004
- Phillip Compeau, Pavel A. Pevzner : Bioinformatics Algorithms, An Active Learning Approach, Vol. I and II, Active Learning Publishers, 2015

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Wahlpflicht- und Major Subject Bachelor-Studiengang PO 2016

Prerequisites

- Contentual: Contents of modules *Programmierung* and *Algorithmen und Datenstrukturen* and *Mathematik für Informatik 1* (or *Lineare Algebra I* or *Analysis I*)

Conditions for awarding credit points

- successful completion of the exercises
- final exam (written, usually 90 min)

Responsible persons

Nguyen Khoa Tran

Last updated 2026-01-20

Computational Geometry

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module imparts basic knowledge from the following areas:

- convex hull
- plane sweep algorithms / segment intersection problems
- distance problems
- geometric divide and conquer / closest point pair
- Voronoi diagrams / Delaunay triangulations / nearest neighbor queries
- triangulation of polygons / monotonic polygons
- area queries / ham-sandwich theorem
- rectangle queries / area trees
- point/line duality/line arrangement
- smallest enclosing circles / randomized algorithms

Learning Outcomes

After completing the course, students are able to

- deal with data structures for computing the convex hull, a voronoi diagram, a Delauney triangulation or a line arrangement
- apply plane-sweep techniques, divide-and-conquer methods and randomized approaches to solve geometric problems
- determine lower bounds for the complexity of geometric problems on sets of points in the Euclidean plane

Bibliography

- de Berg et al: Computational Geometry, Algorithms and Applications. Springer-Verlag. Berlin, 2000. 2. rev. ed.
- Preparata, Shamos: Computational Geometry, an Introduction. Springer-Verlag. New York, 1985.
- Edelsbrunner: Algorithms in Combinatorial Geometry, EATCS Monographs in Computer Science 10. Springer-Verlag, 1987.
- Matousek: Lectures on Discrete Geometry, Graduate Texts in Mathematics, 212. Springer-Verlag, New York, 2002.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules *Algorithms and Data Structures* and *Mathematics for Computer Science 1* (or *Linear Algebra I* and *Calculus I*)

Conditions for awarding credit points

- active participation in the exercises,
- submission of selected homework,
- written exam (usually 90 minutes) or oral exam at the end of the semester

Responsible persons

Prof Dr. Egon Wanke

Last updated 2023-03-23

Computational Complexity Theory

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module imparts basic knowledge from the following areas:

- introduction to computability (Goedelization, diagonalization and subroutine-technique)
- introduction to computational complexity (non-determinism, NP-completeness, …)
- SAT, 3-SAT, Clique, IS, VC, 3-DM, Dominating Set, 3-Partition and other NP-complete problems
- pseudopolynomial algorithms e.g. for knapsack, partition problems
- approximation algorithms
- parameterized algorithms, FPT
- space complexity (Savich's theorem, Immerman's and Szelepcsényi's theorem)
- randomized algorithms

Learning Outcomes

After completing the course, students are able to

- apply non-deterministic calculation models,
- estimate the complexity of algorithmic problems,
- apply basic techniques for approximating solutions,
- develop pseudo-polynomial solution and
- analyze parameterized questions.

Bibliography

- Michael R. Garey and David S. Johnson: Computers & Intractability: A Guide to the Theory of NP-Completeness. Freeman. 1979
- Christos H. Papadimitriou: Computational Complexity. Addison-Wesley. 2008

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Contentual: Contents of modules *Algorithms and Data Structures* and *Mathematics for Computer Science 1* (or *Linear Algebra I* and *Calculus I*)

Conditions for awarding credit points

- active participation in the exercises,
- written exam (usually 90 minutes) or oral exam at the end of the semester

Responsible persons

Dr. Andreas Abels

Last updated 2024-08-23

Operating Systems and System Programming

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

- C programming
- Libraries, binder, and loader
- Processes and threads
- Scheduling: different algorithms (multi level, feedback, realtime); case studies: Linux, Windows, …
- Synchronisation: mutex, sempaphores, deadlocks, lock-free synchronisation
- Main memory: heap, stack, memory management, garbage collection
- Virtual memory: one and several levels, inverted page tables, page reclamation strategies, case study: Linux memory management
- Secondary storage: HDD/SDD characteristics, partitions, memory management
- File systems: FAT, UNIX, ext4, NTFS (including journaling)
- Inter process communication: signal, message queue, pipes, shared memory, sockets
- Input/output: interrupts, I/O software, Linux kernel modules and drivers
- Security: access control, hardware protection mechanisms, buffer exploits, shellcode, meltdown, address space layout randomization, kernel page table isolation
- Architectures: monolith, micro kernel, virtual machines, client/server

Learning Outcomes

After completing the course, students are able to

- describe the interaction of operating system kernel, drivers and hardware
- compare operating system concepts
- develop system programs in C based on the UNIX system interface
- design and implement basic parallel programs using threads and synchronzation primitives
- explain operating system security issues and possible solutions based on hardware protection of the 86 architecture and operating system concepts

Bibliography

- A. Tanenbaum: „Modern Operating Systems“, 4. Aufl., Prentice Hall, 2014.
- W. Stallings, „Operating Systems: Internals and Design Principles”, Prentice Hall, 9. Aufl., 2017.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Formal: Successful completion of the modules *Programming* and *Computer Architecture*

Conditions for awarding credit points

- Active and successful participation in the exercises
- Written exam (90 min, 50% programming, 50% paper work)

Responsible persons

Prof. Dr. Michael Schöttner

Last updated 2025-03-12

Competitive Programming A

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Lab (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

In Competitive Programming the goal is to solve programming problems quickly under stringent runtime and space constraints for the test data. This module teaches basic techniques to recognise common types of contest problems and solve them in C++. The goal is to prepare students for participation at national and international programming contests (e.g., ICPC). There will be weekly homework programming tasks as well as several live contests during lecture time.

Topics of the module:

- Basic algorithms (binary search, sorting, dynamic programming, backtracking, prefix sums)
- Greedy algorithms
- Graph problems (DFS, shortest paths, spanning trees, union-find)
- segment trees

Additional topics are discussed in the module Competitive Programming B in summer semester. Both modules complement each other and can be taken independently.

Learning Outcomes

After completing the course, students are able to

- use data structures and algorithms of the C++ Standard Library,
- estimate the running time of computer programs on given input data sets,
- identify common patterns in easy program contest problems covered by the topics of this module, and
- design and implement efficient programs for easy programming contest problems covered by the topics of this module.

Bibliography

- Ahmed Shamsul Arefin: Art of Programming Contest. Second Edition, Special Online Edition for UVa Online Judge Users
- Steven Halim, Felix Halim, and Suhendry Effendy: Competitive Programming 4: The Lower Bound of Programming Contest in the 2020s, Books 1+2. Lulu Press. 2020
- Nicolai Josuttis: The C++ Standard Library - A Tutorial and Reference. Addison Wesley Longman. 1999. 2nd edition
- Steven Skiena: The Algorithm Design Manual. Springer. 2008. 2nd edition.
- Steven Skiena and Miguel Revilla: Programming Challenges - The Programming Contest Training Manual. Springer Verlag. 2003.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields

- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Good programming skills in C++, Java, or Python and basic knowledge of efficient data structures and algorithms.

Conditions for awarding credit points

The final grade is computed from

- weekly homework problems (50%), a minimum of 4 individual results
- performance at live contests (50%), a minimum of 4 individual results

Responsible persons

apl. Prof. Dr. Rudolf Fleischer

Last updated 2024-07-24

Competitive Programming B

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Lab (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

In Competitive Programming the goal is to solve programming problems quickly under stringent runtime and space constraints for the test data. This module teaches basic techniques to recognise common types of contest problems and solve them in C++. The goal is to prepare students for participation at national and international programming contests (e.g., ICPC). There will be weekly homework programming tasks as well as several live contests during lecture time.

Topics of the module:

- Geometric problems (convex hull, intersections)
- Number theoretical problems (prime number decomposition, divisibility problems)
- Word problems (subwords)

Additional topics are discussed in the module Competitive Programming A in winter semester. Both modules complement each other and can be taken independently.

Learning Outcomes

After completing the course, students are able to

- use data structures and algorithms of the C++ Standard Library,
- estimate the running time of computer programs on given input data sets,
- identify common patterns in easy program contest problems covered by the topics of this module, and
- design and implement efficient programs for easy programming contest problems covered by the topics of this module.

Bibliography

- Ahmed Shamsul Arefin: Art of Programming Contest. Second Edition, Special Online Edition for UVa Online Judge Users
- Steven Halim, Felix Halim, and Suhendry Effendy: Competitive Programming 4: The Lower Bound of Programming Contest in the 2020s, Books 1+2. Lulu Press. 2020
- Nicolai Josuttis: The C++ Standard Library - A Tutorial and Reference. Addison Wesley Longman. 1999. 2nd edition
- Steven Skiena: The Algorithm Design Manual. Springer. 2008. 2nd edition.
- Steven Skiena and Miguel Revilla: Programming Challenges - The Programming Contest Training Manual. Springer Verlag. 2003.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics

- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Good programming skills in C++, Java, or Python and basic knowledge of efficient data structures and algorithms.

Conditions for awarding credit points

The final grade is computed from

- weekly homework problems (50%), a minimum of 4 individual results
- performance at live contests (50%), a minimum of 4 individual results

Responsible persons

apl. Prof. Dr. Rudolf Fleischer

Last updated 2024-07-24

Compiler Construction

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	90 hours	60 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Praktische Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

In this lecutre we discuss the basics of compiler building.

- Lexical analysis (regular expressions and finite automata)
- Syntax analysis (context-free grammars and deterministic parsing)
- Semantic Analysis
- Code Generation
- Using tools to automatically generate compilers

Learning Outcomes

After completing the course, students are able to:

- explain how programming languages are translated and implemented
- explain and customize syntax descriptions of a programming language (In particular, the students should be able to determine whether the description is suitable for automated processing in a compiler), and
- develop a parser or compiler for a new programming language theirself.

Bibliography

- Andrew W. Appel: Modern Compiler Implementation in Java. Cambridge University Press. 2nd Edition
- Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffrey D. Ullman: Compilers: Principles, Techniques, and Tools. Addison Wesley. 2nd Edition

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Formal: Successful completion of the module *Programming Project 1*

Conditions for awarding credit points

- Successful processing of compulsory exercises
- Successful development of your own compiler
- Passing the exam

Responsible persons

Prof. Dr. Michael Leuschel, Dr. John Witulski

Last updated 2026-06-17

Computer-aided reasoning

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	45 hours	105 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (1 HPW)	irregular	Bachelor Computer Science	German

Content

This interdisciplinary course is about proof assistants: Computer programs that can be used to formally construct and verify mathematical proofs. We talk about the theoretical and technical foundations of proof assistants using the example of the proof assistant Lean: functional programming, type theory, and proof tactics. Finally, we discuss how proof assistants can be used for applications in mathematics, computer science, and linguistics.

Learning Outcomes

After completing the course, students are able to

- explain and apply typing rules of type theories,
- formalize data types, mathematical statements and their proofs in the Calculus of Inductive Constructions, and
- apply this theoretical knowledge practically in the proof assistant Lean.

Bibliography

- Smolka: Modeling and Proving in Computational Type Theory Using the Coq Proof Assistant. https://www.ps.uni-saarland.de/~smolka/drafts/icl_book.pdf
- Avigad, de Moura, Kong, Ullrich: Theorem Proving in Lean 4. https://leanprover.github.io/theorem_proving_in_lean4/
- Lecture Notes

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Wahlpflichtbereich Bachelor-Studiengang PO 2013 und PO 2016

Prerequisites

- Contentual: Contents of modules *Mathematics for Computer Science 1*, *Programming* and *Theoretical Computer Science*

Conditions for awarding credit points

- Active and successful participation in the exercise groups
- Passing the written exam

Responsible persons

Dr. Alexander Bentkamp

Last updated 2023-08-01

Data Science 2

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every second winter semester	Bachelor Computer Science	German

Content

Data Science is the application of statistical methods and methods of machine learning to data of any kind, using a computer to model systems and predict behaviour. In the module Data Science 2 the basics from Data Science are deepened close to applications, in particular with respect to the topics:

- big data
- data parallel processing (algorithms and software packages)

Learning Outcomes

After completing the course, students are able to

- analyze algorithms for big data and/or data streams with respect to runtime and communication complexity,
- in particular in the topic areas Near Neighbor Search in High Dimensional Data, Locality Sensitive Hashing (LSH), Dimensionality reduction, Recommendation Systems, Clustering (variants of k-means), Link Analysis (PageRank), Web Advertising,
- decide which kinds and which size of data needs parallelization,
- decide which techniques of parallel processing are feasible for given problems,
- in particular the map-reduce technique,
- and to implement these together with using common software packages.

Bibliography

- Leskovec, Rajaraman and Ullman: Mining of Massive Datasets. Cambridge University Press. 2020. 3rd Edition (mmds.org)

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Contentual: Contents of modules *Data Science* or *Machine Learning*

Conditions for awarding credit points

- active and successful participation in the theoretical and practical exercises
- passing the written exam (usually 90 minutes)

Responsible persons

Dr. Konrad Völkel

Last updated 2023-03-23

Databases: Further Concepts

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Lab (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

Building upon the basics of database design and query languages for relational databases, this module covers further concepts of databases and database systems. Beside the fundamentals the practical realization (database design and database application programming) is the main focus of this module. For this, the following topics are covered in the lecture:

- database definition (using SQL)
- database application programming and using databases in the Web
- data protection and data security (SQL injection; views and user privileges in databases)
- trigger
- aspects of im plementiong database systems (physical storage; index structures)
- algorithms for query operators and query optimization

Learning Outcomes

After completing the course, students are able to

- develop database applications (including database design, database definition and application programming),
- respect basic aspects of data protection and data security during the development of database applications, and
- explain and assess essential implementation concepts for storing data as well as elementary data structures and algorithms for query processing.

Bibliography

- G. Saake, K.-U. Sattler, A. Heuer: Datenbanken – Konzepte und Sprachen. mitp Verlag. 2018. TODO Auflage
- G. Saake, K.-U. Sattler, A. Heuer: Datenbanken – Implementierungstechniken. mitp Verlag. 2011. 3. Auflage
- Kemper, A. Eickler: Datenbanksysteme – Eine Einführung. Oldenbourg Verlag. 2015. 10. Auflage
- R. Elmasri, S.B. Navathe: Fundamentals of Database Systems. Pearson. 2016. 7th edition
- H. Garcia-Molina, J.D. Ullman, J. Widom: Database Systems: The Complete Book. Pearson. 2009. 2nd edition

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields

Prerequisites

- Contentual: Contents of module *Databases: An Introduction*

Conditions for awarding credit points

- active and successful participation in the exercises (mix of theoretical and practical tasks)
- either final exam (60 minutes) or portfolio (will be announced in the first lecture)

Responsible persons

Prof. Dr. Stefan Conrad

Last updated 2026-06-30

Data Visualization

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

Data visualization skills will be taught.

- Human perception, aesthetics, coordinate systems
- Color scales, visualizations of selected data types (quantitative values, distributions, proportions,trends, geodata, inaccuracies), labeling, tables
- Visualization of multidimensional data, storytelling, image formats and visualization tools

Learning Outcomes

After completing the course, students are able to

- select and create appropriate visualization for the corresponding data types
- take into account the acquired knowledge about human perception when creating visualizations, and
- select appropriate image formats and tools for visualization.

Bibliography

- Further literature is provided during the course.
- Wilke, Claus O.: Datenvisualisierung – Grundlagen und Praxis: Wie Sie aussagekräftige Diagramme und Grafiken gestalten. Sebastopol: O´Reilly, 2020.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- Group work with presentation; 20% weighted
- Paper on the group work at the end of the semester; 80% weighted

Responsible persons

Prof. Dr. Dominik Heider, Dr. Hannah Franziska Löchel

Digital Innovation: From Idea to Impact

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	30 hours	120 hours
Components	**Häufigkeit des Angebots**	**Course of Study**	**Language of instruction**
Lecture/Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This course is designed for future founders, CTOs (Chief Technology Officers), and anyone who wants to know how to develop technology-based innovations and bring them to market in startups and large companies.

Through practice-oriented theoretical input and accompanying hands-on applications, relevant methods, tools, and processes are taught that are necessary to develop an initial idea into a concrete, market-ready product that is attractive to its target group and has a sustainable impact on both the economy and society.

Through group projects, students apply the tools and methods to create and evaluate their own digital ideas, translate them into a digital product, and design a prototype.

Guest speakers from the regional and/or national innovation ecosystem share their experiences.

Learning Outcomes

After completing the course, students are able to

- Create, develop, analyze, and evaluate digital innovations,
- Model, analyze, and discuss digital business models and their components,
- Implement and evaluate digital innovation in prototypes,
- Present and evaluate group results in front of peers and experts, and
- Provide feedback.

Bibliography

- Allmendinger, M.; Horstmann, M.; Horstmann, O.: Digitale Innovationen entwickeln. Die besten Ansätze und Methoden. Haufe Lexware GmbH, 2020.
- Osterwalder, A; Pigneur, Y.: Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. John Wiley & Sons, 2010. 1st edition
- Tidd, J; Bessant, J. R.: Managing Innovation: Integrating Technological, Market and Organizational Change. Wiley, 2020. 7th edition
- Further literature is provided during the course.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

None

Conditions for awarding credit points

- presentation
- Written elaboration (in groups)

Responsible persons

Prof. Dr. Steffi Haag

Last updated 2026-01-19

Introduction to Algorithmic Game Theory

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This module teaches foundations in the field of algorithmic game theory. Different areas within algorithmic game theory will be introduced and important results will be taught.

- Strategic games (normal form, dominance, pure and mixed equilibria and their existance)
- Potential games (atomic- and non-atomic selfish routing games, price of stability and anarchy, potential functions)
- Mechanism design (simple auctions, Myerson's lemma, VCG prices, revenue maximization)
- Mechanism design without money (kidney exchange, stable matching)

Learning Outcomes

After completing the course, students are able to

- recognize and explain the game theoretic models discussed in the course,
- reproduce and apply the results discussed in the course,
- explain the discussed properties and metrics, and apply these in simple analyses, and
- develop new game theoretic models and mechanisms.

Bibliography

- Noam Nisan, Tim Roughgarden, Éva Tardos, Vijay V. Vazirani: Algorithmic Game Theory. Cambridge University Press. 2007.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of the modules *Algorithms and Data Structures* and *Mathematics for Computer Science 1* (or, alternatively, *Linear Algebra I* and *Calculus I*)

Conditions for awarding credit points

- successfull participation in the exercises
- writen exam (usually 90 minutes) or oral examination at the end of the semester

Responsible persons

Dr. Andreas Abels

Last updated 2024-02-16

Introduction to Deep Learning

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every second winter semester	Bachelor Computer Science	German

Content

Building on the competences in machine learning from the *Data Science* module, we treat a selection of core topics in theory and application of neural networks, with a view towards Deep Learning, such as:

- single neurons: logistic regression and activation functions
- automatic differentiation and backpropagation
- latent variables, autoencoder
- implementation of training and inference with Pytorch
- using pretrained models

Learning Outcomes

After completing the course, students are able to

- name and explain basic notions and concepts of neural networks,
- apply mathematical prerequisites for neural networks,
- implement simple models,
- integrate pretrained models in other systems, and
- judge which of the models discussed are sensible to use in applications.

Bibliography

- Nielsen, Michael A.: Neural Networks and Deep Learning. Determination Press. San Francisco, 2015. (online)
- Goodfellow, Ian and Bengio, Yoshua and Courville, Aaron: Deep Learning. MIT. Cambridge, 2017. (online)
- Stevens, Eli and Antiga, Luca and Viehmann, Thomas: Deep Learning with PyTorch. Manning. New York, 2020 (online)
- Zhang, Aston and Lipton, Zachary C. and Li, Mu and Smola, Alexander J.: Dive into Deep Learning. Online. Everywhere, 2021. (online)

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module *Data Science*

Conditions for awarding credit points

- active participation in the tutorials
- handing in the homework
- written exam (regularly 90 minutes) or oral exam at the end of the term

Responsible persons

Dr. Konrad Völkel

Last updated 2023-08-01

Introduction to Functional Programming

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

This module consists of concepts widespread in the functional programming paradigm. We use Clojure - a modern Lisp running on the JVM - as a programming langauge. The following topics are covered:

- Clojure's syntax and Clojure programming
- immutable data structures and laziness
- the epochal time model
- simplicity and Clojure philosophy
- polymorphismus a la carte
- homoiconicity and macros.

Learning Outcomes

After completing the course, students are able to

- name characteristics of functional programming and to compare them with traditional, imperative programming,
- evaluate for which applications functional programming is beneficial,
- create and test functional programs,
- explain and apply the concepts listed above (Content).

Bibliography

- Moseley, Marks: Out of the tarpit. Online
- Fogus, Houser: The Joy of Clojure. Manning. 2014. 2nd edition.
- Emerick, Carper, Grand: Programming Clojure. O'Reilly. 2012. 1st edition.
- Rathore, Avila: Clojure in Action. Manning. 2016. Second 2nd.
- Higginbotham: Clojure for the Brave and True. No Starch Press. 2015. 1st edition.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Contentual: Contents of modules *Programming* and *Programming Project 1*

Conditions for awarding credit points

- Understanding of the material
- Depending on the number of participants:
- preferably written exam (exam, usually 90 minutes)
- oral exam (usually 30-45 minutes)

Responsible persons

Philipp Körner, Dr. Jens Bendisposto, Prof. Dr. Michael Leuschel

Last updated 2024-08-20

Introduction to Scientific Computer Science

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

With the help of selected examples, the module describes the application of computer science and statistics to solve various problems in biology, physics and chemistry. Essential parts of the underlying algorithms are implemented in the Python programming language:

- Fast Fourier-Transform to reduce runtime in multiple Alignment
- Pebble-Game-Algorithm for rigidity analysis of biomolecules
- Motiv-search in DNA sequences using Gibbs-Sampling
- Dynamic programming in pairwise sequence comparison
- Clustering algorithms for sequence and expression data: Neighbor-joining, Markov-clustering-algorithm, k-means, expectation maximization
- Lateral gene transfer or phylogenetic artifact? Statistical test to test congruence of trees with non-identical leaf sets without a reliable reference tree.
- Rooting phylogenetic trees using mean-ancestor-deviation
- Recursion and the problem of independent-phylogenetic-contrasts

Learning Outcomes

After completing the course, students are able to

- describe the underlying scientific background and the associated problems of the presented methods,
- apply the algorithms presented for problem solving to example data,
- critically compare different possible solutions to a problem, and
- independently implement the learned methods in the programming language Python.

Bibliography

- Relevant literature is provided during the course.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields

Prerequisites

- None

Conditions for awarding credit points

- Minimum of 50 percent points from exercises

- Passing the written examination (usually 90 minutes)

Responsible persons

Dr. Mayo Röttger, Prof. Dr. Martin Lercher

Last updated 2023-08-01

Introduction to Python

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Remarks

- Places are limited due to the room situation. Information on the allocation of places can be found on the LSF.

Content

Python is a popular programming language that allows imperative, functional, object-oriented and other programming paradigms, in particular to process data in systems for machine learning. In this introduction, concepts about programming that are known in Java are applied with Python.

- data types, control structures, exception handling and input/output with Python
- tools for software development with Python (linter, profiler, test-driven development)
- application development with Python (user interfaces and data processing)

Learning Outcomes

After completing the course, students are able to

- describe the differences between Java and Python,
- comprehend foreign Python source code and their functionality,
- evaluate and improve given Python source code,
- develop relevant test cases for given Python source code,
- systematically find bugs in Python source code,
- develop Python programs using generative AI,
- program simple applications for data processing in Python and
- retain typical conventions to manage Python software packages.

Bibliography

- Klein: Einführung in Python 3: Für Ein-und Umsteiger. Carl Hanser Verlag GmbH Co KG. München, 2021. 4. Auflage
- Sweigart: Automate the Boring Stuff with Python: Practical Programming for Total Beginners. No Starch Press. San Francisco, 2019. 2nd. Edition

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Wahlpflicht- und Major Subject Bachelor-Studiengang PO 2016

Prerequisites

- Formal: Passed the module *Programming*

Conditions for awarding credit points

- Portfolio

Responsible persons

Dr. Konrad Völkel

Last updated 2026-01-20

Introduction to Randomized Algorithms

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Remarks

The exam for this module cannot be taken if you have already passed the module ”Randomized Algorithms und Analysis”.

Content

This course is a lecture on advanced algorithms and deals with randomization as a method of algorithm design and analysis. The goal is to obtain efficient algorithms via randomized decisions that run faster than their deterministic variants and at the same time provide precise results with high probability.

- models of randomized algorithms (Las-Vegas and Monte-Carlo algorithms)
- running time and accuracy analysis
- randomized approximation algorithms (e.g., for SAT and graph problems)
- methods for probability amplification
- randomized design paradigms (e.g., probabilistic method, fingerprints, hashing)
- randomization in data analysis

Learning Outcomes

After completing the course, students are able to

- describe technical terms and basic methods of randomization in algorithms,
- match algorithms to different design paradigms and apply them examplarily,
- analyse, evaluate and compare properties such as running time and accuracy and
- design first randomized algorithms according to adequate design methods.

Bibliography

- Juraj Hromkovič: Randomisierte Algorithmen: Methoden zum Entwurf von zufallsgesteuerten Systemen für Einsteiger. Teubner-Verlag. Wiesbaden, 2004. 1. Ausgabe
- Michael Mitzenmacher and Eli Upfal: Probability and Computing. Cambridge University Press. Cambridge, 2017. 2nd Edition.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Contentual: Contents of modules *Algorithms and Data Structures*, *Theoretical Computer Science* and *Mathematics for Computer Science 3*

Conditions for awarding credit points

- actively participate in tutorials
- successfully work on exercises
- final exam (written or oral exam)

Responsible persons

Prof. Dr. Melanie Schmidt, Dr. Anja Rey, Dr. Andreas Abels

Last updated 2024-10-01

Embedded Systems

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	45 hours	105 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lab (4 HPW)	every winter semester	Bachelor Computer Science	German

Remarks

- For didactic reasons, only a limited number of students per semester can take part in this course.

Content

This module is an introduction to microcontrollers, CAD and FPGAs.

The concepts of microcontrollers are explored using Arduinos. The fundamentals of sensors and actuators are taught. An example microcontroller and CAD project is the Cody Computer, an 8-bit homebrew computer.

FPGA programming is taught using the Verilog and VHDL languages and is practiced in practical exercises on FPGAs.

The module is offered according to the inverted classroom principle: The lecture content is taught asynchronously through instructional videos and applied live in-person in practical exercises.

Specifically, the following topics will be covered:

- Microcontrollers / Arduino Uno R3: Arduino IDE, I/O, analog vs. digital, sensors and actuators
- Microcontrollers / Parallax Propeller: The Cody Computer, Assembler
- Serial Communication: UART/COM, SPI, I2C
- CAD: Circuit diagrams with KiCAD
- FPGA basics: Structure and functional principles
- AMD/Xilinx Vivado, Intel/Altera Quartus Prime, Command Line Tools
- Various FPGAs and FPGA boards in practice: Basys 3 (AMD/Xilinx Artix 7), MAX1000 (Intel MAX10), Lattice iCE40, Tang Nano (Gowin LittleBee)
- Hardware description languages: Verilog and VHDL

Learning Outcomes

After completing the course, students are able to

- Independently build and explain circuits with microcontrollers and FPGAs
- Read, explain, and create program code in Verilog and VHDL
- Read circuit diagrams with the known building blocks
- Independently develop and implement projects with microcontrollers and FPGAs
- name concepts of microcontrollers and FPGAs and explain how they work

Bibliography

- Uwe Altenburg: Embedded Systems. Rheinwerk Verlag, Bonn. 2025. Erstausgabe
- Danny Schreiter: Arduino Kompendium. BMU Verlag, 2019. Erstausgabe
- Claus Kühnel: Arduino: Das umfassende Handbuch. Rheinwerk Verlag, Bonn. 2024. Erstausgabe
- Erik Bartmann: FPGA für alle. Bombini Verlag. 2023. Erstausgabe

- Jörg Rippel: FPGAs: Einstieg, Schaltungen, Projekte. Rheinwerk Verlag, Bonn. 2024. Erstausgabe
- Steven Hugg: Designing Video Game Hardware in Verilog. Independently published. 2018. 1st Edition
- Frederick John Milens III: The Cody Computer Book. Open Book. 2025. Public draft

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Wahlpflicht- und Major Subject Bachelor-Studiengang PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Contentual: Contents of modules *Programming* and *Computer Architecture*

Conditions for awarding credit points

- Successful completion of the exercises
- Successful completion of the oral exam (usually 30 minutes)

Responsible persons

Dr. John Witulski

Last updated 2026-07-15

Case study seminar: Digital Future Challenge

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Seminar (2 HPW)	every winter semester	Bachelor Computer Science	German

Content

This course is aimed at students who want to develop innovative digital solutions for the upcoming challenges of our time and would like to take the opportunity to take part in the Digital Future Challenge (DFC) university competition with this idea. For the Digital Future Challenge (DFC), we are working together with the Deloitte Foundation and the Initiative D21 with the aim of examining corporate responsibility in the course of digitalisation (Corporate Digital Responsibility) from various perspectives and developing solutions for digital-ethical issues.

As part of the seminar, groups of students will develop innovative ideas and approaches on the topic of corporate digital responsibility (e.g. in the areas of Europe, climate & environment and inclusion), which will be submitted to the DFC university competition in November. Methods and techniques in the following subject areas are taught in workshops:

- Creative idea generation and problem solving
- Design thinking
- Teambuilding
- Development of a business model (Business Model Canvas)
- Project management
- Pitch training

All participants who submit their idea to the DFC also have the chance to be invited to the semi-finals in Berlin and pitch (i.e. present) their innovations in front of a jury from science and business.

Learning Outcomes

After completing the course, students are able to

- Develop innovative ideas and approaches to solve future challenges using digital technologies
- Explain and apply methods and techniques for generating ideas and solving problems
- Explain and practically implement design thinking
- Develop and discuss digital business models
- Design, present and evaluate convincing idea pitches

Bibliography

- Relevant literature is provided during the course.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Individual Supplementary Course im Master-Studiengang Informatik

Prerequisites

- Content: Participation in the course *Digital Innovation: From Idea to Impact* is recommended

Conditions for awarding credit points

- Written elaboration of the idea
- Presentation of the idea
- Active participation in discussions

Responsible persons

Prof. Dr. Steffi Haag, Nina Grigoriou

Last updated 2024-08-16

Algorithms for Graphs 1

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This module covers basic knowledge from the following areas.

- searching in graphs
- topological sorting
- connectivity problems
- shortest path problems
- minimal spanning trees
- network flow problems
- matching problems

Learning Outcomes

After completing the course, students are able to

- describe and explain the discussed graph algorithms,
- allocate the discussed algorithms to different problem settings and apply them adequately,
- analyze the discussed algorithms with respect to their running time and correctness and
- design and analyze easy new graph algorithms.

Bibliography

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein: Algorithmen – Eine Einführung. De Gruyter Oldenbourg. 2017. 4. Auflage.
- Ravindra Ahuja, Thomas Magnanti, James B. Orlin: Network Flows: Theory, Algorithms and Applications. Prentice Hall. Englewood Cliffs, 1993. 1. Ausgabe

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules *Programming*, *Algorithms and Data Structures* and *Mathematics for Computer Science 1*

Conditions for awarding credit points

- actively participate in tutorials
- hand in exercises
- final written exam (usually 90 min.) or oral exam at the end of the semester)

Responsible persons

Prof. Dr. Melanie Schmidt, Dr. Daniel Schmidt

Last updated 2024-10-02

Foundations of Distributed Systems

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (3 HPW) Tutorial (1 HPW)	every summer semester	Bachelor Computer Science	German

Content

- Architectures (client/server, cloud, fog, edge)
- Sockets, multithreading and scalability
- Remote procedure call (gRPC)
- Time (clock synchronization, logical time, causality)
- Group communication and pub/sub
- Replication and consisteny (basics)
- Global states (asynchronous snapshots, applications)
- Fault tolerance (fault detection and recovery)
- Weak consistency and scalability (gnutella, chord, dynamo)
- Strong consistency and scalability (transactions, Paxos)
- Security (basics)

Learning Outcomes

After completing the course, students are able to

- explain and compare different distributed systems architectures,
- compare and apply different communication concepts,
- apply the discussed distributed algorithms,
- describe and compare replication and consistency strategies, also regarding their scalability,
- explain fault tolerance models and recovery strategies, and
- describe basic security aspects.

Bibliography

- G. Coulouris et.al., „Distributed Systems: Concepts and Design“, Addison-Wesley, 5. Aufl. 2011
- A. Tanenbaum and M. van Steen: „Distributed Systems: Principles and Paradigms”, 3. Auflage, Prentice Hall, 2013.
- Further literature is provided during the course.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contents of the module *Programming*

Conditions for awarding credit points

- Passing the exam

Responsible persons

Prof. Dr. Michael Schöttner

Last updated 2024-07-24

Introduction to Logic Programming

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	90 hours	60 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW) Lab (2 HPW)	every winter semester	Bachelor Computer Science	English German

Remarks

The slides are available in English and German.

Content

Logic programming is quite different from the classical imperative approach to programming. Logic programming is a declarative programming language where one declares the properties of a solution rather than providing an algorithm to solve a problem step-by-step. The lecture will give students a new perspective on programming which will also prove useful even if software is still developed in classical programming languages like C or Java.

The lecture covers the following topics:

- Propositional logic, predicate logic
- Resolution
- Programming with Horn clauses
- Practical foundations of Prolog
- Search algorithms and AI with Prolog
- Basics of constraint programming

Learning Outcomes

After completing the course, students are able to

- apply the logical foundations of Prolog to perform derivations in propositional and predicate logic
- use Prolog data structures to encode data
- develop smaller Prolog programs independently, as seen in the exercises
- compare advantages and drawbacks of various search algorithms and implement them in Prolog
- solve smaller symbolic AI tasks in Prolog

Bibliography

- Nilsson, Maluszynski: Logic, Programming and Prolog (2nd edition). (eBook)
- Blackburn, Bos, Striegnitz: Learn Prolog Now!. College Publications.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

- Elective Area Master study programme Artificial Intelligence and Data Science
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Contentual: Programming skills.

Conditions for awarding credit points

- Active and successful participation in the exercises
- Successful participation at the final exam

Responsible persons

Prof. Dr. Michael Leuschel

Last updated 2024-08-20

Introduction to Modelling metabolic networks

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	English

Content

- Introduction to the statistical programming language R
- Repetition of basic linear algebra
- Basic properties and reconstruction of stoichiometric matrices
- Topology and fundamental subspaces of the stoichiometric matrix
- Elementary flux modes
- Properties of the solution space
- Flux balance analysis
- Flux variability, flux coupling
- Modeling of gene knockouts
- Flux balance analysis with molecular crowding
- Resource balance analysis

Learning Outcomes

After completing the course, students are able to

- summarize important constraint-based modeling techniques and apply them to metabolic networks,
- describe biological systems from possible biochemical reactions,
- formulate and solve linear optimization problems using the R programming language and
- consider metabolic modules as a system and simulate their behavior under different conditions.

Bibliography

- Bernhard Ø. Palsson: Systems Biology: Properties of Reconstructed Networks. Cambridge University Press. 2015.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules *Programming* und *Mathematics for Computer Science 1* (or *Lineare Algebra I*)

Conditions for awarding credit points

- active participation in the exercises
- successful completion of the exercises (50%)
- passing final exam (usually written)

Responsible persons

Prof. Dr. Martin Lercher

Last updated 2023-03-23

Introduction to NLP

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) + Tutorial (2 HPW)	every winter semester	Bachelor Computer Science	English

Content

This course provides an introduction to foundational methods as well as state of the art techniques for natural language processing. It covers essential topics such as text processing, statistical language modeling, and text classification using logistic regression. We will have a closer look at neural modeling, word and contextual embeddings, the Transformer architecture and large language models. Core NLP tasks such as sequence labeling, parsing, semantic modeling, translation, information retrieval, question answering and dialogue modeling are addressed, providing a comprehensive overview of the topic.

- Foundations & Preprocessing: NLP tasks & paradigms, tokenization, normalization
- Language Modeling & Evaluation: N-grams, neural LMs, perplexity, BLEU
- Text Classification & Embeddings: naive Bayes, logistic regression, Word2Vec
- Sequence Modeling & Parsing: RNN limitations, self-attention, POS/NER, parsing
- Transformer: details, architectures, BERT pre-training & fine-tuning
- Applications & Advanced Topics: MT, summarization, QA, IR, dialogue systems
- LLMs & Trends: GPT evolution, prompting, alignment, hallucination, trends & outlook

Learning Outcomes

After completing the course, students are able to

- name and explain core NLP tasks and apply text processing techniques
- implement basic classical and neural methods for text classification and sequence labeling,
- apply statistical and neural language models and conduct their evaluation,
- conceptualize word and contextual embeddings,
- explain basic parsing and semantic modeling,
- clarify the basics of the Transformer architecture and how to fine-tune pretrained models,
- explain approaches to neural machine translation, information retrieval and question answering,
- explain the design and the challenges of large language models and dialogue systems.

Bibliography

- Daniel Jurafsky and James H. Martin: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models. Online (https://web.stanford.edu/~jurafsky/slp3), 2026. 3rd edition.
- Jacob Eisenstein: Introduction to Natural Language Processing. The MIT Press. United States of America, 2019.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Wahlpflicht- und Major Subject Bachelor-Studiengang PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields

- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module *Data Science*

Conditions for awarding credit points

- gathering at least 50% of the total points achievable in four mini tests (typically 15 minutes) that are conducted in presence. The dates of the mini tests will be communicated with the beginning of the course in the first lecture and on ILIAS.
- passing the written examination (typically 90 minutes)

Responsible persons

Prof. Dr. Milica Gasic, Dr. Michael Heck

Last updated 2026-06-25

Combinatorial Algorithms for Clustering Problems

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This course is a lecture on advanced algorithms and deals with clustering algorithms. In this lecture, we discuss combinatorial clustering problems and related algorithms. In particular, we discuss:

- hierarchical clustering
- algorithms for the k-center problem
- algorithms for the k-supplier problem
- similarity-based clustering procedures

Learning Outcomes

After completing the course, students are able to

- describe and develop combinatorical arguments,
- apply the discussed clustering algorithms,
- discuss and evaluate hierarchical clustering methods, and
- analyse clustering precedures.

Bibliography

- Selected publications regarding the topic of the course.
- Lecture Notes

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Contentual: Contents of module *Mathematics for Computer Science 1*

Conditions for awarding credit points

- actively participate in tutorials
- successfully work on exercises
- final exam (written or oral exam)

Responsible persons

Prof. Dr. Melanie Schmidt

Last updated 2024-08-20

Cryptocomplexity 1

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module provides a basic understanding of the foundations of cryptology, introduces some important cryptosystems, and discusses their security that often relies on the computational complexity of suitable problems. Therefore, the foundations of complexity theory are introduced as well, in particular with the goal to understand methods for proving and applying lower bounds with respect to the complexity measures time and space. A special focus is given to understanding the close connections between these two fields.

Introduction to Cryptology

- Some Classical Cryptosystems and their Cryptoanalysis (Substitution and Permutation Chiffre, Affin Linear Block Chiffre, Block and Stream Chiffres)
- Perfect Secrecy (Shannon's Theorem and Vernam's One-Time Pad, Entropy and Key Equivocation)
- RSA, Primality Tests and the Factorization Problem (the Public-Key Cryptosystem RSA, Digital Signatures with RSA, Security of RSA)

Introduction to Complexity Theory

- Foundations (Complexity Measures and Classes, Compression and Speed-Up Theorems, Hierarchy Theorems)
- Between L and PSPACE (simplee Inclusions, Complexity-Bounded Many-One-Reductions, Complete Problems in NL, NP-Complete Problems)

Learning Outcomes

After completing the course, students are able to

- classify symmetric cryptosystems with respect to their properties,h
- evaluate the security of block chiffres and other classical cryptosystems,
- reason why certain cryposystems possess which properties,
- explain the idea of public-key cryptography,
- explain the basic goals and definitions of complexity theory,
- describe and evaluate the complexity of natural problems,
- unassistedly design reductions between problems to show their lower bounds and provide a corresponding proof of correctness.

Bibliography

- Jörg Rothe: Komplexitätstheorie und Kryptologie. Eine Einführung in Kryptokomplexität. eXamen.Press. Springer-Verlag, Heidelberg, 2008. 1. Auflage.
- Jörg Rothe: Complexity Theory and Cryptology. An Introduction to Crypto-complexity. EATCS Texts in Theoretical Computer Science, Springer-Verlag. Heidelberg, 2005. 1. Auflage.
- Further literature is provided during the course.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- None

Conditions for awarding credit points

- active and successful participation in the exercises
- written exam (exam, usually 90 minutes)

Responsible persons

Prof. Dr. Jörg Rothe

Last updated 2024-08-20

Logic for Computer Science

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module is an introduction to logic. The relationship between syntax and semantics, like completeness and expressiveness, are investigated for different logical systems and related to questions from computer science.

The following topics will be covered:

- Classical Propositional Logic: Syntax, Semantics, correctness and completeness, further properties (e.g. interpolation, maximality, functional completeness), SAT-solving
- some non-classical propositional logics, their syntax and semantics
- first order logic, correctness and completeness, compactness, Löwenheim-Skolem theorem
- Gödel's incompleteness theorem
- extensions of first order logic
- finite model theory and descriptive complexity
- higher order logic and type theory

Learning Outcomes

After completing the course, students are able to

- conduct proofs in a given formal system and check proofs for correctness
- show the (non-)derivability of formulas by semantical means
- show correctness and completeness of simple logical systems
- relate different logical systems to each other
- explain central theoretical results, like completeness theorems, the compactness theorem, and Gödel's incompleteness theorems, judge their applicability in concrete situations, and apply them

Bibliography

- Lecture Notes
- Ebbinghaus, Flum, Thomas: Einführung in die mathematische Logik. Springer, Berlin. 2018.
- Further literature is provided during the course.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Wahlpflicht- und Major Subject Bachelor-Studiengang PO 2016
- Individual Supplementary Course for the Master Computer Science study programme
- Anwendungsfach Informatik im Bachelor- oder Master-Studiengang Mathematik und Anwendungsgebiete

Prerequisites

- Contentual: The sections on logic of the modules *Computer Architecture* and *Mathematics for Computer Science 1*

Conditions for awarding credit points

- successful completion of the exercises
- written exam (usually 90 minutes) or oral exam

Responsible persons

Dr. Peter Arndt

Last updated 2026-02-05

Patterns in nature: theoretical background and algorithms

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

The course will consider patterns found in nature, in terms of their formation, the underlying chemical/physical processes, possible mathematical modeling, and the algorithms to recreate them on the computer.

- Natural patterns (symmetry, fractals, spirals, chaos, parquetry, dots and stripes (Turing Patterns).
- Underlying chemical/physical processes, mathematical modeling (e.g., reaction diffusion equation, Fibonacci numbers, and golden ratio)
- Algorithms to recreate these on the computer (such as cellular automata, L-systems, Chaos Game).

Learning Outcomes

After completing the course, students are able to

- recognize the different patterns that occur in nature and explain how they are created
- implement and use the common algorithms for generating these patterns

Bibliography

- Further literature is provided during the course.
- Lecture Notes

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Content: Contents of the modules *Programming*, *Algorithms and Data Structures* and *Mathematics for Computer Science 1* (or *Linear Algebra I* or .*Calculus I*)

Conditions for awarding credit points

- Group work with presentation; 20% weighted
- Paper on the group work at the end of the semester; 80% weighted

Responsible persons

Dr. Hannah Franziska Löchel

Last updated 2024-02-16

Preference Aggregation by Voting: Algorithmics and Complexity

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	irregular	Bachelor Computer Science	German

Content

This module provides a basic understanding of the foundations, models, methods, and results of the young and rapidly evolving interdisciplinary field *Computational Social Choice* that has applications in the areas *Artificial Intelligence* and *Multiagent Systems*. In particular, properties of voting systems are investigated and the related decision problems (winner determination, manipulation, electoral control, bribery, etc.) are studied in terms of their algorithmical solvability and complexity.

Foundations of Social Choice Theory

- Elections
- Voting Systems (Scoring Protocols like Plurality, Veto, and Borda; Condorcet; Copeland; Maximin; Dodgson; Young; Bucklin; Fallback; etc.)
- Properties of Voting Systems
- Some Voting Paradoxa and Impossibility Results

Algorithmics and Complexity of Voting Problems

- Winner Determination
- Possible and Necessary Winners
- Manipulation
- Electoral Control
- Bribery

Learning Outcomes

After completing the course, students are able to

- describe the most common voting systems (scoring protocols like Plurality, Veto, and Borda; Condorcet; Copeland; Maximin; Dodgson; Young; Bucklin; Fallback; etc.) and their properties,
- discuss the reasonability of axiomatic properties of voting systems,
- determine the winners of the most common voting systems for any given preference profile,
- give examples of successful manipulation, control, and bribery actions for the most common voting systems and describe the related decision problems,
- argue why which of these problems can either be solved by efficient algorithms or are hard to solve.

Bibliography

- Jörg Rothe (ed.): Economics and Computation: An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division. Springer Texts in Business and Economics, Springer-Verlag. Heidelberg, 2015. 1. Auflage.
- Jörg Rothe, Dorothea Baumeister, Claudia Lindner und Irene Rothe: Einführung in Computational Social Choice. Individuelle Strategien und kollektive Entscheidungen beim Spielen, Wählen und Teilen. Spektrum Akademischer Verlag (Springer). Heidelberg, 2011. 1. Auflage.

- Further literature is provided during the course.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- None

Conditions for awarding credit points

- active and successful participation in the exercises
- written exam (exam, usually 90 minutes)

Responsible persons

Prof. Dr. Jörg Rothe

Last updated 2024-08-20

Python for NLP

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lab (4 HPW)	irregular	Bachelor Computer Science	English

Content

- Python programming foundations and development workflows for natural language processing (NLP), including debugging, testing, and reproducible experimentation.
- Text representation and data preparation techniques, covering tokenisation, vectorisation, embeddings, and preprocessing pipelines.
- Design, implementation, and training of neural network models for sentiment analysis task, including feed-forward networks RNNs, and attention mechanisms.
- Practical model training, evaluation, and optimisation, including GPU-based training and comparative analysis of model configurations.

Learning Outcomes

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

- implement NLP pipelines, from raw text processing to model training and evaluation, using Python.
- explain and apply core neural network concepts for sequential data, including gradient-based optimisation, recurrent architectures, and attention mechanisms.
- analyse the strengths and limitations of different neural architectures for NLP tasks and select appropriate models for given problems.
- conduct structured experiments, utilise GPU resources effectively, and present results in a scientifically sound and reproducible manner.

Bibliography

- Daniel Jurafsky and James H. Martin: Speech and Language Processing (https://web.stanford.edu/~jurafsky/slp3/)

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Wahlpflicht- und Major Subject Bachelor-Studiengang PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields

Prerequisites

- Contentual: Basic understanding of Python programming and deep learning.
- Formal: None.

Conditions for awarding credit points

- Portfolio.
- Passing grades for presentation and report.

Responsible persons

Prof. Dr. Milica Gasic, Shutong Feng

Last updated 2026-01-19

From Circuits to Software

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Häufigkeit des Angebots**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	every summer semester	Bachelor Computer Science	German

Content

This course offers a journey through different areas of computer science, in order to convey an overall picture of the basic functioning of computers. Starting from the NAND gate, all important hardware components of a computer are reproduced in a simulation, ALU, CPU, RAM, etc.

For the CPU and the associated machine language developed during the lecture, an assembly language, a virtual machine and a programming language as well as the associated compiler will be developed step by step. With all these tools, a simple operating system and application programs are finally developed.

The lecture is accompanied by exercises. The independent practical application of what has been learned will be the focus of the event. In the exercises, the students develop the various components of modern computers presented in the lecture.

Learning Outcomes

After completing the course, students are able to

- to explain and evaluate the basic principles of all discussed levels and
- independently develop new functionalities on these levels.

Bibliography

- Noam Nisan, Shimon Schocken: The Elements of Computing Systems: Building a Modern Computer from First Principles. MIT Press. Cambridge, 2008. 1st Edition

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Formal: Successful completion of the module *Programming Project 1*

Conditions for awarding credit points

- successfully completing the exercises
- passing the final exam

Responsible persons

Dr. John Witulski

Seminar

The seminar serves as preparation for the bachelor thesis and therefore should be completed before registration.

These courses are different in terms of enrollment and attendance than most other courses at the Institute of Computer Science. Specifically, that means:

- Places are limited to 25 participants per seminar group.
- These places are usually allocated before the start of lectures and the allocation of places is completed at the start of lectures.
- The exact allocation deadlines can be found in HIS/LSF.
- In accordance with the examination regulations, attendance is compulsory in seminars.

Compulsory Seminar

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Seminar (2 HPW) Seminar / Lecture / Tutorial (2 HPW)	every semester	Bachelor Computer Science	German, English

Remarks

- The language of instruction for the course is indicated in the information linked in HIS/LSF.
- For more information on the seminar topics, visit https://www.cs.hhu.de/bachelor21/seminar.
- The weighting of the criteria for awarding credit points for the grade will be announced in the first or second session.
- Exam registration must take place within the first two weeks of lectures via ILIAS. The exact registration deadline will also be announced via HIS/LSF.
- Cancellation of the examination can be made up to one week before the first graded examination at the latest. The lecturers of the respective seminar groups will announce the exact date within the first two weeks of lectures.

Content

In a course of the module “Compulsory Seminar”, students independently acquire knowledge and methods on a topic of computer science with the help of literature and other sources. They practise presenting the previously unknown and now independently learned content in a presentation, learn how to write scientific texts, and give other students constructive feedback. Both the oral and the written presentation of the independently developed topic serve as preparation for the Bachelor’s thesis.

Seminars are offered by all computer science working groups on various topics. The concrete, thematic content of a seminar can be found in the information linked in HIS/LSF.

The quality of the presentation and the written seminar paper contribute to the seminar grade. The assessment criteria are communicated at the beginning of the seminar.

Learning Outcomes

After completing the course, students are able to

- independently find, select, evaluate, summarise and critically discuss relevant, qualitatively appropriate literature, methods and sources on a given specialist topic
- present this topic in a presentation appropriate to the target audience, adhering to a time limit
- give other people constructive feedback on a presentation
- critically discuss a topic with other computer scientists
- explain, classify and discuss a specialist topic in a coherent, structured technical text, adhering to formal criteria

Bibliography

- J. Zobel: Writing for Computer Science. Springer. 2014. 3. Auflage
- Selected publications regarding the topic of the course.

Module compatibility

- Bereich Pflichtseminar Bachelor-Studiengang PO 2021

Prerequisites

- Formal: Successful completion of the module *Scientific Methods* or 90 CP already achieved in the Bachelor's degree program in Computer Science.
- Contentual: see Compulsory Seminar information linked in HIS/LSF

Conditions for awarding credit points

- at least sufficient written elaboration
- at least sufficient seminar presentation
- active participation in the seminar sessions
- attendance at 80% of the seminar presentations

Responsible persons

Lehrende der Informatik

Last updated 2026-06-02

Bachelor Thesis

The bachelor thesis module consists of two parts, both of which are graded.

Thesis

ECTS credits	Total hours	Contact hours	Self-study hours
12 CP	360 hours	-	-
Components	**Cycle**	**Course of Study**	**Language of instruction**
-	anytime	Bachelor Computer Science	German English

Content

The content of the Bachelor's thesis is in a chosen module of Computer Science. The bachelor thesis must be written in German or English.

Learning Outcomes

With the written thesis, the students should demonstrate that they are able to:

- within a specified period (of 3 months)
- work independently on a topic and
- present this appropriately.

In the assessment interview, students should demonstrate that they are able to present the topic they have worked on independently in an appropriate oral presentation and be able to answer questions about their own work.

Bibliography

- In agreement with the supervisor.

Module compatibility

- Bachelorarbeit

Prerequisites

To register for the bachelor thesis, at least 120 of the Proof of the credit points to be acquired as part of the bachelor's degree will.

Conditions for awarding credit points

- Successful processing of the topic and presentation in one draft submitted on time (Bachelor thesis)
- Passed assessment interview on the content of the Bachelor's thesis

Responsible persons

Lehrende der Informatik

Last updated 2024-08-26

Final Seminar

ECTS credits	Total hours	Contact hours	Self-study hours
3 CP	90 hours	30	60
Components	**Cycle**	**Course of Study**	**Language of instruction**
-	anytime	Bachelor Computer Science	German English

Content

Within the framework of the graded final seminar, the subject and results of the Thesis are presented in front of the corresponding working group (or a merger of such) with subsequent scientific discussion.

Learning Outcomes

After participating in the final seminar, the students are able to

- present an independently developed topic in a talk
- to follow scientific talks of other graduates
- get involved in discussions about scientific presentations by others students

Bibliography

-

Module compatibility

- Bachelorarbeit

Prerequisites

The thesis must be registered.

Conditions for awarding credit points

Active participation in the final seminar and presentation of your own topic in an oral presentation with discussion.

Responsible persons

Lehrende der Informatik

Last updated 2022-10-04

Modules no longer offered

In this chapter you will find all modules that we no longer offer.

Applied Algorithmics

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

> “In theory, there is no difference between theory and practice. In practice, there is.”

Algorithms are the foundation of every computer program. Traditionally, the focus of algorithm design is on the theory of efficient algorithms and their worst-case analysis. In this module we focus on practically efficient algorithms for provably hard optimization problems. The aim is to not (totally) give up the principles of optimality. Topics are:

- Fundamental aspects of algorithms and complexity
- Complete enumeration and dynamic Programmierung
- Branch and Bound
- Approximation algorithms
- Heuristics and metaheuristics
- (Integer) linear programming
- Fixed-parameter tractable algorithms
- Modelling with satisfiability (SAT)

Learning Outcomes

After completing the course, students are able to

- apply the presented applied algorithmic design techniques to new problems,
- solve problems practically efficient and implement, test and evaluate these solutions.

Bibliography

- Steven Skiena, The Algorithm Design Manual, Second Edition, Springer, 2008

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of modules *Programming*, *Algorithms and Data Structures*, *Theoretical Computer Science* and *Mathematics for Computer Science 1* (or *Linear Algebra I* or *Calculus I*)

Conditions for awarding credit points

- active participation in the tutorial classes
- successful completion of the exercises (50%)
- final exam (written, usually 90 minutes)

Responsible persons

Prof. Dr. Gunnar W. Klau

Last updated 2025-06-25

Bachelor’s Seminar: Introduction to blockchain technology

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Tutorial (2 HPW) Seminar (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

- Rules for constructive feedback
- Basics of blockchain technology (hash functions, signatures, proof of work, …)
- Decentralized currencies (e.g. Bitcoin)
- Further content will be determined in consultation with the students based on the seminar presentations (e.g. attacks, defense strategies, various protocols)

Learning Outcomes

After completing the course, students are able to

- Familiarize themself with a given topic area independently,
- Describe this topic in writing and present it in a lecture,
- Use feedback of fellow students about their work to revise it,
- Give constructive feedback on other drafts and presentations,
- Explain the basics of blockchain technology,
- Name and justify different areas of application of the blockchain, and
- Summarize the contents of the individual seminar presentations.

Bibliography

- Selected publications regarding the topic of the course.

Module compatibility

- Seminar Bachelor-Studiengang PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Formal: Successful completion of the scientific work module

Conditions for awarding credit points

- At least sufficient essay(this will be made available to all participants of the course)
- Participation in the peer review process of the elaborations
- At least sufficient seminar presentation
- Pass an oral exam

Responsible persons

Janine Golov

Last updated 2023-08-01

Bachelor’s-Seminar: Programming Languages

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Seminar (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Remarks

- Students who do not study according to PO 2021 will be given lower priority in the allocation of places.

Content

The seminar Programming Languages conveys general knowledge about programming languages as well as their typical properties and concepts. The seminar consists on the one hand of a series of lectures in which participants introduce a language and on the other hand an exercise in which this language is used by programming tasks. Writing an elaboration on lecture language prepares participants for the bachelor thesis.

- General knowledge of programming languages (Properties, differences, applications/areas of application, syntax and semantics)
- Properties and concepts of programming languages
- give a lecture on an independently developed topic, as well as write an elaboration on it
- Writing a written elaboration on a programming language

Learning Outcomes

After completing the course, students are able to

- describe and explain properties of the discussed programming languages,
- independently create a lecture and an elaboration on a given topic, and
- give this lecture.

Bibliography

- Bruce A. Tate. 2010: Seven Languages in Seven Weeks: A Pragmatic Guide to Learning Programming Languages. Pragmatic Bookshelf. 1st edition

Module compatibility

- Bereich Bachelor-Studiengang PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016

Prerequisites

- Contentual: Contents of module Programming

Conditions for awarding credit points

- Create an elaboration

- Giving a lecture
- Processing of programming tasks

Responsible persons

Dr. John Witulski

Last updated 2023-08-01

Bachelor's-Seminar: Introduction to Artificial Intelligence

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	90 hours	60 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture/Seminar (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science und Master DSAI	German / English (DSAI)

Remarks

- Students who do not study computer science according to PO 2021 will be given lower priority in the allocation of places.

Content

The lecture covers the following topics:

- Rules for constructive feedback
- Basics of artificial intelligence, from classical symbolic AI to modern techniques like deep learning
- Further content will be determined in consultation with the students based on the seminar presentations

Learning Outcomes

After completing the course, students are able to

- familiarize themselves with a given topic area independently,
- describe this topic in writing and present it in a lecture,
- use feedback of fellow students about their work to revise it,
- give constructive feedback on other drafts and presentations,
- asses the capabilities and problem-specific applicability of different AI approaches (e.g., expert systems, SVM, decision trees, random forests, CNNs, …)
- explain the goals and techniques of various subfields of AI,
- name and explain the limits of current AI approaches (e.g., bias, explainability, safety, …) and judge the relevance for concrete practical questions.

Bibliography

- Bratko: Prolog Programming for Artificial Intelligence. Addison Wesley
- Russel, Norvig: Artificial Intelligence: A Modern Approach. Prentice Hall
- Szeliski: Computer Vision: Algorithms and Applications. Springer
- Goodfellow, Bengio, Courville: Deep Learning. MIT Press

Module compatibility

- Bereich Seminar Bachelor-Studiengang PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Elective Area Master study programme Artificial Intelligence and Data Science

Prerequisites

- Formal: Successful completion of the scientific work module

Conditions for awarding credit points

- seminar presentation
- successful creation of a written paper
- written exam

Responsible persons

Prof. Dr. Michael Leuschel

Last updated 2023-03-23

Data Visualization

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

Data visualization skills will be taught.

- Human perception, aesthetics, coordinate systems
- Color scales, visualizations of selected data types (quantitative values, distributions, proportions,trends, geodata, inaccuracies), labeling, tables
- Visualization of multidimensional data, storytelling, image formats and visualization tools

Learning Outcomes

After completing the course, students are able to

- select and create appropriate visualization for the corresponding data types
- take into account the acquired knowledge about human perception when creating visualizations, and
- select appropriate image formats and tools for visualization.

Bibliography

- Further literature is provided during the course.
- Wilke, Claus O.: Datenvisualisierung – Grundlagen und Praxis: Wie Sie aussagekräftige Diagramme und Grafiken gestalten. Sebastopol: O´Reilly, 2020.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- Group work with presentation; 20% weighted
- Paper on the group work at the end of the semester; 80% weighted

Responsible persons

Prof. Dr. Dominik Heider, Dr. Hannah Franziska Löchel

Collective Decisions

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

This module deals with different methods of collective decision-making under preferences. Such methods are applied in several areas of artificial intelligence, for example in the interaction of autonomous agents. Central contents of this lecture are different methods with their axiomatic and algorithmic properties from the three areas: voting, participatory budgeting, and resource allocation.

- Voting (voting rules, properties, impossibility results)
- Participatory budgeting (preferences, aggregation rules, properties)
- Resource allocation (preferences, social welfare, allocations)

Learning Outcomes

After completing the course, students are able to

- describe different methods for collective decision making and apply them in concrete situations,
- define new decision-making methods and analyze them with respect to their properties, and
- give justified recommendations for the usage of particular decision-making methods under given conditions.

Bibliography

- Jörg Rothe, Dorothea Baumeister, Claudia Lindner und Irene Rothe: Einführung in Computational Social Choice. Individuelle Strategien und kollektive Entscheidungen beim Spielen, Wählen und Teilen. Spektrum, Akademischer Verlag. 2011.
- Felix Brandt, Vincent Conitzer, Ulle Endriss, Jerome Lang, and Ariel Procaccia (eds.): Handbook of Computational Social Choice. Cambridge University Press. 2015.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module *Theoretical Computer Science*

Conditions for awarding credit points

- active and successful participation in the theoretical exercise courses

- writen test (exam, usually 90 minutes)

Responsible persons

apl. Prof. Dr. Dorothea Baumeister

Last updated 2023-03-23

Matching

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

This module deals with different methods of matching under preferences. Such methods are applied in different areas, examples include school admissions, allocation of residents to hospitals, or the allocation of houses to students. Contents of this lecture are different methods with their axiomatic and algorithmic properties.

- matching in graphs
- bipartite matching with two-sided preferences (stable marriage problem)
- non-bipartite matching with preferences (stable roommates problem)
- bipartite matching with one-sided preferences (house allocation problem)

Learning Outcomes

After completing the course, students are able to

- conduct matching algorithms in different situations,
- identify challenges in practical matching problems,
- develop matching algorithms for specific areas and investigate their properties,
- transfer known matching algorithms to new areas,
- compare different matching algorithms, and
- recommend methods for specific areas of matching.

Bibliography

- Dan Gusfield and Robert W. Irving: The Stable Marriage Problem, Structure and Algorithms. MIT Press. 1989.
- David F. Manlove: Algorithmics of Matching under Preferences. World Scientific Publishing Company. 2013.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Contentual: Contents of module *Theoretical Computer Science*

Conditions for awarding credit points

- active and successful participation in the theoretical exercise courses
- writen test (exam, usually 90 minutes)

Responsible persons

apl. Prof. Dr. Dorothea Baumeister

Last updated 2023-03-23

Patterns in nature: theoretical background and algorithms

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

The course will consider patterns found in nature, in terms of their formation, the underlying chemical/physical processes, possible mathematical modeling, and the algorithms to recreate them on the computer.

- Natural patterns (symmetry, fractals, spirals, chaos, parquetry, dots and stripes (Turing Patterns).
- Underlying chemical/physical processes, mathematical modeling (e.g., reaction diffusion equation, Fibonacci numbers, and golden ratio)
- Algorithms to recreate these on the computer (such as cellular automata, L-systems, Chaos Game).

Learning Outcomes

After completing the course, students are able to

- recognize the different patterns that occur in nature and explain how they are created
- implement and use the common algorithms for generating these patterns

Bibliography

- Further literature is provided during the course.
- Lecture Notes

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- Content: Contents of the modules *Programming*, *Algorithms and Data Structures* and *Mathematics for Computer Science 1* (or *Linear Algebra I* or .*Calculus I*)

Conditions for awarding credit points

- Group work with presentation; 20% weighted
- Paper on the group work at the end of the semester; 80% weighted

Responsible persons

Dr. Hannah Franziska Löchel

Last updated 2024-08-20

Randomized Algorithms und Analysis

ECTS credits	Total hours	Contact hours	Self-study hours
10 CP	300 hours	90 hours	210 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (4 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

This course is a lecture on advanced algorithms and deals with randomization as a method of algorithm design and analysis. The goal is to obtain efficient algorithms via randomized decisions that run faster than their deterministic variants and at the same time provide precise results with high probability.

- models of randomized algorithms (Las-Vegas and Monte-Carlo algorithms)
- running time and accuracy analysis
- randomized approximation algorithms (e.g., for SAT and graph problems)
- methods for probability amplification
- randomized design paradigms (e.g., probabilistic method, fingerprints, hashing)
- randomization in data analysis

Learning Outcomes

After completing the course, students are able to

- describe technical terms and basic methods of randomization in algorithms,
- match algorithms to different design paradigms and apply them examplarily,
- analyse, evaluate and compare properties such as running time and accuracy and
- design first randomized algorithms according to adequate design methods.

Bibliography

- Juraj Hromkovič: Randomisierte Algorithmen: Methoden zum Entwurf von zufallsgesteuerten Systemen für Einsteiger. Teubner-Verlag. Wiesbaden, 2004. 1. Ausgabe
- Michael Mitzenmacher and Eli Upfal: Probability and Computing. Cambridge University Press. Cambridge, 2017. 2nd Edition.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics
- Individual Supplementary Course for the Master Computer Science study programme

Prerequisites

- Contentual: Contents of modules *Algorithms and Data Structures*, *Theoretical Computer Science* and *Mathematics for Computer Science 3*

Conditions for awarding credit points

- actively participate in tutorials
- successfully work on exercises
- final exam (written or oral exam)

Responsible persons

Prof. Dr. Melanie Schmidt, Dr. Anja Rey

Last updated 2024-10-01

Statistical Data Analysis

ECTS credits	Total hours	Contact hours	Self-study hours
5 CP	150 hours	60 hours	90 hours
Components	**Cycle**	**Course of Study**	**Language of instruction**
Lecture (2 HPW) Tutorial (2 HPW)	no longer offered	Bachelor Computer Science	German

Content

The module is based on a course and related book by Prof. Gianluca Bon-tempi at the Université Libre de Bruxelles on the statistical foundations of machine learning. Contents are in detail:

- The R programming language for statistical computing
- Descriptive statistics
- Basics of probability calculation
- Classic parametric estimation and testing
- Non-parametric estimation and testing
- Statistical Learning
- Linear approaches
- Non-linear approaches
- Dimensionality reduction

Learning Outcomes

After completing the course, students are able to

- summarize basic concepts of statistical data analysis (e.g. probability distributions, parametric and non-parametric hypothesis tests, parameter estimators, conditional probabilities, permutation tests),
- devise parametric and non-parametric tests for statistical hypotheses,
- formulate statistical models with multi-dimensional predictors,
- plan and carry out statistical data analyses with R, and
- create meaningful graphical representations of data using ggplot2.

Bibliography

- Gianluca Bontempi. Handbook statistical foundations of machine learning. Self-published, Brüssel, 2021. 2nd edition.

Module compatibility

- Elective Area Bachelor study programme PO 2021
- Elective Area and Major Subject Bachelor study programme PO 2013 and PO 2016
- Application subject Bachelor study programme Mathematics and Applied Fields
- Minor subject Bachelor study programme Physics
- Minor subject Bachelor study programme Medical Physics

Prerequisites

- None

Conditions for awarding credit points

- active participation in the exercises
- successful completion of the exercises (50%)
- final exam (usually written)

Responsible persons

Prof. Dr. Martin Lercher

Last updated 2024-07-24