Computer Science

Find another programme

First year courses

BSc CS year 1 core courses

Dept. of Advanced Computing Sciences

Introduction to Computer Science

Full course description

The primary goal of Introduction to Computer Science is to introduce fundamental concepts and foster critical skills found throughout the field of computer science. Fundamental concepts include algorithms, computer architecture and hardware, models of computation, computer networks, and operating systems. Critical skills include abstraction, decomposition, pattern recognition, and algorithmic thinking. All concepts and skills are introduced in a lecture setting and explored further in the lab through the development of a wirelessly controlled microcontroller device. At the end of this course, students will appreciate the depth of the field and be prepared for subsequent research and educational activities.

Prerequisites

None.

Recommended reading

- "Computational Thinking for the Modern Problem Solver" by David Riley, Kenny A. Hunt
- "Computer Science Illuminated" by Nell B. Dale

BCS1110
Period 1
2 Sep 2024
25 Oct 2024
Print course description
ECTS credits:
4.0
Coordinators:

- A.R.K. Sai
- T.A. Bitterman

Teaching methods:
Lecture(s)
Assessment methods:
Written exam, Assignment
Dept. of Advanced Computing Sciences

Procedural Programming

Full course description

The course provides the basics of computer science and computer programming. After a short introduction to computer organization, the principles of programming are presented. The main topics of the course are: data types, variables, methods, parameters, decision structures, iteration, arrays, recursion and a branching application (related to the semester project). Programming skills will be acquired during practical sessions using the object-oriented programming language Java.

Prerequisites

None. It appears as part of the pre-requisites of the second semester project in year 1, both projects of year 2, the year 2 course Databases and the year 3 courses, Parallel Programming and Robotics and Embedded Systems.

The course appears as desired prior knowledge for the courses Introduction to Objects in Programming, Data Structures and Algorithms, Software Engineering, Databases and Machine Learning.

Recommended reading

H. Schildt, Java: A Beginner's Guide, Eighth Edition, ISBN: 1260440214, McGraw-Hill Education

BCS1120
Period 1
2 Sep 2024
25 Oct 2024
Print course description
ECTS credits:
4.0
Coordinators:

- E. Hortal Quesada
- C. Kouzinopoulos

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam, Assignment
Dept. of Advanced Computing Sciences

Discrete Mathematics

Full course description

In this course, we build a mathematical framework that is based on logic and reason. The main objective of the course is to make students familiar with the language of mathematics. Students will learn how to make sound arguments and to detect where and why certain arguments go wrong. For

this purpose, we will discuss the basic principles of logic and, closely related, the basic types of mathematical proofs. In doing so, we will encounter numbers such as integers, natural numbers and real numbers and we shall examine what makes these numbers special. After that, we will use basic logic to discuss, among other things, the following mathematical concepts: infinity, sets, relations, functions, permutations and combinations. Our fundamental tool in all of this is plain common sense. You really do not need your toolbox of mathematical formulas learned in previous studies and neither do you need a calculator. Pen and paper are the basic instruments needed. After completing each topic, exercises will be provided to be completed in class or at home, since mathematics is mainly learned by practising repeatedly.

Prerequisites

None.

Recommended reading

None.

BCS1130
Period 1
2 Sep 2024
25 Oct 2024
Print course description
ECTS credits:
4.0
Coordinators:

- M. Musegaas
- O. D'Huys

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam
Dept. of Advanced Computing Sciences

Objects in Programming

Full course description

This course is a follow-up of the course Procedural Programming. It teaches object-oriented programming in Java. The main topics covered in the course are objects and classes, interfaces and polymorphism, event handling, inheritance, graphic user interfaces, exception handling, and streams.

Computer Science

Prerequisites

Desired prior knowledge: Basic Java Programming

Recommended reading

C. Horstmann (2016). Java Concepts (8th Edition). John Wiley & Sons, New York, ISBN: 978-1-1190-5645-4 C. Horstmann (2012). Big Java Late Objects. John Wiley & Sons, New York, ISBN 978-1-1180-8788-6

BCS1220
Period 2
28 Oct 2024
13 Dec 2024
Print course description
ECTS credits:
4.0
Coordinators:

- T.A. Bitterman
- E.N. Smirnov
- F. Barile

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam, Assignment
Dept. of Advanced Computing Sciences

Calculus

Full course description

The following subjects will be discussed in Calculus: limits and continuity, differential calculus, inverse and transcendental functions, mean value theorem, integral calculus, sequences and series, introduction to differential equations, introduction to multivariable calculus. In addition to the main facts and concepts, problem-solving strategies will be discussed. Both the intuition behind the concepts and their rigorous definitions will be presented along with simple examples of formal mathematical proofs.

Prerequisites

None

Recommended reading

None

BCS1440

Computer Science

Period 2
28 Oct 2024
13 Dec 2024
Print course description
ECTS credits:
4.0
Coordinators:

- O. D'Huys
- G.M. Schoenmakers

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam
Dept. of Advanced Computing Sciences

Logic

Full course description

This course deals with three logical systems, namely propositional logic, first-order predicate logic and epistemic logic. The course covers notation systems, syntax and semantics, valid consequences, deduction, semantic tableaux, and proof systems.

Prerequisites

None

Recommended reading

None

BCS1530
Period 2
28 Oct 2024
13 Dec 2024
Print course description
ECTS credits:
4.0
Coordinators:

- T.D. Rienstra
- S.J. Maubach
- N. Roos

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam, Assignment

Computer Science

Dept. of Advanced Computing Sciences

Project 1-1

Full course description

Students work on a project assignment in small groups of about six students. The group composition stays the same for the whole project and is announced shortly before the project opening in period 1.1. The students are guided through the project by a fixed tutor. The project assignment is divided into three subtasks (one per period) and is strongly related to the content of the courses from period 1.1 and 1.2. In period 1.1, after receiving the assignment for the whole project at the end of week 5, the students work full-time on the project in week 6. In this week, each group meets the tutor twice. In period 1.2, the students continue working on the project, while also having to attend the courses of that period. They meet their tutor approximately once a week. In period 1.3, the students work three weeks full-time on the project and meet their tutor about twice a week.

At the beginning of period 1.2 and 1.3, the students have to hand in a planning for the current phase. At the end of each period, the students have to give a presentation and the source code, presentation and an overview of who did what need to be uploaded to Canvas. While the presentations at the end of period 1.1 and 1.2 are in front of the examiners and the tutors, the presentations at the end of period 1.3 will additionally be in front of the fellow students. In period 1.3, they furthermore have to hand in a report and attend a product and report examination.

Project 1-1 will start in period 1.1 and period 1.2. The credits for the projects will become available at the end of period 1.3.
For each period, we will give a short explanation of the various parts. Before the start of each period, the students will receive detailed information about the content, the study material, the teaching form, the schedule, and the examination method.

Prerequisites

This project has no prerequisites. This project occurs as part of the prerequisites of project 2-1.

Recommended reading

None

BCS1300
Semester 1
2 Sep 2024
24 Jan 2025
Print course description
ECTS credits:
6.0
Coordinator:

- M. Boussé

Teaching methods:
Project-Centered Learning, Work in subgroups, Presentation(s), Skills

Computer Science

Assessment methods:
Assignment, Presentation and paper, Participation
Dept. of Advanced Computing Sciences

Linear Algebra

Full course description

This course introduces the fundamental concepts of linear algebra, and examines them from both an algebraic and a geometric point of view. First, we address what can be recognized without doubt as the most frequently occurring mathematical problem in practical applications: how to solve a system of linear equations. Then we discuss linear functions and mappings, which can be studied naturally from a geometric point of view. Vectors spaces are then introduced as a common framework that brings all themes together. Next, we shift from the geometric point of view to the dynamic perspective, where the focus is on the effects of iterations (i.e., the repeated application of a linear mapping). This involves a basic theory of eigenvalues and eigenvectors, which have many applications in various branches of science as for instance in problems involving dynamics and stability, in control theory, and in optimization problems found in data science. Key concepts in the course are vectors, matrices, systems of linear equations, eigenvalues, eigenvectors, linear transformations, and orthogonality. The software package Matlab is introduced in the accompanying computer classes, where emphasis is put on the application of linear algebra to solve real world problems.

Prerequisites

None.

Recommended reading

None.

BCS1410
Period 4
27 Jan 2025
28 Mar 2025
Print course description
ECTS credits:
4.0
Coordinators:

- M. Musegaas
- P.W.L. Dreesen

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam
Dept. of Advanced Computing Sciences

Computer Science

Data Structures and Algorithms

Full course description

As a continuation of the courses Procedural Programming and Objects in Programming, this course will treat the systematic design and application of data structures and algorithms.
Data structures such as lists, trees, graphs, and dictionaries, the associated algorithms and their complexity are explored in this course. Algorithms for applications such as sorting, pattern matching and graph
traversal are also part of the course. Furthermore, design principles for algorithms such as recursion, divide-and-conquer and dynamic programming will be treated as well. Furthermore, students will develop skills to analyse the run-time and space complexity of data structures and algorithms.

Prerequisites

None.

Desired Prior Knowledge: Discrete Mathematics, Procedural Programming and Objects in Programming.

Recommended reading

Required Reading: Sedgewick and Wayne (2011) Algorithms Fourth Edition. Addison Wesley. ISBN: 978-0321573513

Recommended Reading: A Y Bhargava (2016). Grokking Algorithms: An Illustrated Guide for Programmers and Other Curious People. Manning. ISBN: 978-1617292231

BCS1420
Period 4
27 Jan 2025
28 Mar 2025
Print course description
ECTS credits:
4.0
Coordinators:

- T.H.J. Pepels
- F. Barile
- S.S. Kotala

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam, Assignment
Dept. of Advanced Computing Sciences

Computer Science

Object-Oriented Modelling

Full course description

This course introduces students to the design and analysis aspects of object-oriented programming. Software construction for real world applications has inherent complexities both in terms of designing and maintaining it. In this course, the students will learn how to model a real-world problems in an object-oriented programming context using tools like Unified Modelling Language (UML). Students will also learn techniques such as structural, behavioral and creational design patterns, GRASP principles to create modular, flexible and reusable software. After completing the course, the students would have gained practical experience in problem formulation, decomposition (analysis) and solution building (design) using object-oriented modelling techniques.

Prerequisites

None.

Recommended reading

- "Clean Code: A Handbook of Agile Software Craftsmanship" by Robert Cecil Martin
- "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma et al.

BCS1430
Period 4
27 Jan 2025
28 Mar 2025
Print course description
ECTS credits:
4.0
Coordinators:

- A.R.K. Sai
- A.J. Garnock-Jones

Teaching methods:
Lecture(s)
Assessment methods:
Written exam, Assignment
Dept. of Advanced Computing Sciences

Databases

Full course description

This course introduces students to fundamental concepts of databases with a focus on relational database systems. Students will learn how to design, implement and optimize a relational database. The course provides a comprehensive study of data modeling, data description languages, and query facilities such as relational algebra and SQL. Students will also be exposed to the internal workings of database management systems with an overview of topics such as concurrency, recovery,

indexing, and triggers. The course will cover current topics related to managing Big Data using alternate data models such as NoSQL. The students will apply their knowledge on these topics in designing and implementing a database system as a part of a group project.

Prerequisites

None.

Recommended reading

"Readings in Database Systems" by Peter Bailis, Joseph M. Hellerstein, Michael Stonebraker, 5th Edition.

BCS1510
Period 5
31 Mar 2025
23 May 2025
Print course description
ECTS credits:
4.0
Coordinators:

- K. Schneider
- A.J. Garnock-Jones

Teaching methods:
Lecture(s)
Assessment methods:
Written exam, Assignment
Dept. of Advanced Computing Sciences

Statistics

Full course description

: Statistics introduces the student to the main concepts of both probability theory and statistics. With respect to probability theory, students learn how to make use of random variables to extract the probability distribution of an experiment. Additionally, topics such as expectation, standard deviation, and independence will be discussed. The statistics part of the course discusses basic statistical topics such as the central limit theorem, verification of hypotheses, and confidence intervals. After completing this course students will have obtained an overview of commonly seen probability distributions, as well as several statistical procedures. Additionally, the student will be able to deal with problems that involve probabilities and determine an outcome for such problems (e.g., the expected outcome).

Prerequisites

None.

Computer Science

BCS1520
Period 5
31 Mar 2025
23 May 2025
Print course description
ECTS credits:
4.0
Coordinator:

- A. Wodeyar

Teaching methods:
Lecture(s)
Assessment methods:
Written exam, Take home exam
Dept. of Advanced Computing Sciences

Algorithmic Design

Full course description

Algorithmic Design formalizes the main algorithmic paradigms and techniques including greedy and divide-and-conquer strategies, dynamic programming, multi-dimensional searching, computational geometry, linear programming, randomization, and approximation algorithms. It familiarizes students with amortization and NP-completeness. After completing the course, students will be expected to show good design principles and adequate skills at reasoning about the correctness and complexity of algorithms.

Prerequisites

None.

Recommended reading

Goodrich and Tamassia (2015) Algorithm Design and Applications. Wiley. ISBN: 978—1-118-33591-8

BCS1540
Period 5
31 Mar 2025
23 May 2025
Print course description
ECTS credits:
4.0
Coordinators:

- T.A. Bitterman
- Y. Wang

Teaching methods:
Lecture(s)

Computer Science

Assessment methods:
Written exam, Assignment
Dept. of Advanced Computing Sciences

Project 1-2

Full course description

Students work on a project assignment in small groups of about six students. The group composition stays the same for the whole project and is announced before the project opening in period 1.4. The students are guided through the project by a fixed tutor. The project assignment is divided into three subtasks (one per period) and is strongly related to the content of the courses from period 1.4 and 1.5. In period 1.4, after receiving the assignment for the whole project at the end of week 5, the students work full-time on the project in week 6. In this week, each group meets the tutor twice. In period 1.5, the students continue working on the project, while also having to attend the courses of that period. They meet their tutor approximately once a week. In period 1.6, the students work three weeks full-time on the project and meet their tutor twice a week.

At the beginning of period 1.5 and 1.6, the students have to hand in a planning for the current phase. At the end of each period, the students have to give a presentation and the source code, presentation and an overview of who did what need to be uploaded to Canvas. While the presentations at the end of period 1.4 and 1.5 are in front of the examiners and the tutors, the presentations at the end of period 1.6 will additionally be in front of the fellow students. In period 1.6, they furthermore have to hand in a report and attend a product and report examination.

Project 1-2 will start in period 1.4 and period 1.5. The credits for the projects will become available at the end of period 1.6.
For each period, we will give a short explanation of the various parts. Before the start of each period, the students will receive detailed information about the content, the study material, the teaching form, the schedule, and the examination method.

Prerequisites

In order to participate in this project the student has to have passed two out of four courses from the set: Discrete Mathematics, Calculus, Procedural Programming and Objects in Programming.

This project occurs as part of the prerequisites of project 2-2.

Recommended reading

None.

BCS1600
Semester 2
27 Jan 2025
20 Jun 2025
Print course description
ECTS credits:
6.0

Computer Science

Coordinator:

- O. D'Huys

Teaching methods:
Project-Centered Learning, Work in subgroups, Presentation(s), Skills
Assessment methods:
Assignment, Presentation and paper, Participation
Second year courses

If the student has obtained at least 45 credits in the first year courses, they may register for the second year courses.

Second year students **choose one** out **of two** electives modules in semester 1 and **choose one** out **of three** electives modules in semester 2.

BSc CS year 2 core courses

Dept. of Advanced Computing Sciences

Computer Networks

Full course description

This course introduces the fundamental concepts in computer networking. It covers the principles and structures of network architectures, protocols, and interfaces. Topics include the OSI and TCP/IP models, network devices, routing algorithms, wireless communication, network security, and emerging technologies.

Prerequisites

None.

Recommended reading

None.

BCS2110
Period 1
2 Sep 2024
25 Oct 2024
Print course description
ECTS credits:
4.0
Coordinator:

- A.I. Iamnitchi

Teaching methods:
Project-Centered Learning, Work in subgroups

Computer Science

Assessment methods:
Written exam
Dept. of Advanced Computing Sciences

Introduction to Artificial Intelligence

Full course description

The course starts with an analysis of the question “Can machines think?”, and the preconceptions usually encountered in discussions about that idea. Next, the metaphor of an “intelligent agent” is introduced, that is, of an entity that pursues goals by perceiving and acting flexibly and autonomously in a possibly very complex environment. Several state-of-the-art concepts, algorithms, and methods that enable computers (i.e., software and robots) to solve problems in a way that deserves to be called intelligent are discussed. Besides the technical background, societal and ethical issues around artificial intelligence such as the current quest for human-centered and explainable AI and how computational techniques might lead to biases and misconceptions through incautious data collection are discussed

Prerequisites

None.

BCS2120
Period 1
2 Sep 2024
25 Oct 2024
Print course description
ECTS credits:
4.0
Coordinators:

- A.S. Härmä
- D.J.N.J. Soemers

Teaching methods:
Project-Centered Learning
Dept. of Advanced Computing Sciences

Intelligent User Interfaces

Full course description

Intelligent user interaction is a relatively new field in Computer Science, involving the areas of Human-Computer Interaction and Artificial Intelligence. It can often be seen as the intersection of computer science, behavioural sciences, and other field of study. This course will start with an overview of how to develop user-centric interfaces, what is the role of data analysis and research design methods in prototyping, and how different phases, from ideation to high fidelity prototyping relate to each other. Moreover, intelligent user interactions, involving technologies able to automatically recognize human intentions and behaviours, will be discussed. These can come, for

example, from the fields of computer vision, speech processing, text mining. A thriving area in the field is that of interface personalization and affective computing, that is, computing driven by human emotions and behaviours.

Prerequisites

None.

BCS2130
Period 1
2 Sep 2024
25 Oct 2024
Print course description
ECTS credits:
4.0
Coordinators:

- K. Zarkogianni
- Y.C. Semerci

Teaching methods:
Project-Centered Learning
Dept. of Advanced Computing Sciences

Software Engineering and Architectures

Full course description

In this course, the student is introduced to the software engineering process. The course addresses the way in which large and complex software projects are conceived and managed. Topics in this course include, among others, requirement analysis, design methodologies, implementation strategies and test and maintenance procedures. In addition, the course discusses the software architectural design process. Several guidelines and several popular example software architectures are presented, as are different software delivery platforms and the current state of the art app development. After completing this course, the student will be able to judge the viability of a selected software development methodology and architectures.

Prerequisites

None.

BCS2210
Period 2
28 Oct 2024
13 Dec 2024
Print course description
ECTS credits:
4.0
Coordinator:

Computer Science

- T.A. Bitterman

Teaching methods:
Project-Centered Learning
Dept. of Advanced Computing Sciences

Principles of Programming Languages

Full course description

Programming Languages are ubiquitous. Every program with an “if” statement is, in an important sense, an interpreter for a programming language, however simple! This course will equip students with an analytical toolkit for understanding contemporary programming languages, relating them to a “standard model” of programming languages. The flip-side of analysis is construction: students will develop the basic understanding and knowledge necessary to begin creating programming languages of their own and writing tools which interpret, compile, or otherwise process other programs as their input data. On the way, students will examine the algebraic, functional “heart” of many programming languages, using equational reasoning to investigate time, change, concurrency and communication in programs.

Prerequisites

Discrete Mathematics; Procedural Programming; Object-Oriented Modelling.

Recommended reading

Krishnamurthi, Shriram. Programming Languages: Application and Interpretation. Version 3.2.2., 2023. https://www.plai.org/.

Additional literature:

1. Felleisen, Matthias, Robert Bruce Findler, Matthew Flatt, and Shriram Krishnamurthi. How to Design Programs. Second edition. MIT Press, 2014. https://htdp.org/.
2. Fisler, Kathi, Shriram Krishnamurthi, Benjamin S. Lerner, and Joe Gibbs Politz. A Data-Centric Introduction to Computing. Version 2023-02-21., 2023. https://dcic-world.org/.
3. Van Roy, Peter, and Seif Haridi. Concepts, Techniques and Models of Computer Programming. Cambridge, Massachusetts: MIT Press, 2004. https://www.info.ucl.ac.be/~pvr/book.html.
4. Abelson, Harold, Gerald Jay Sussman, and Julie Sussman. Structure and Interpretation of Computer Programs. 2nd ed. Cambridge, Massachusetts: The MIT Press, 1996. https://mitpress.mit.edu/sites/default/files/sicp/index.html.
5. Felleisen, Matthias, Robert Bruce Findler, and Matthew Flatt. Semantics Engineering with PLT Redex. Cambridge, Massachusetts: MIT Press, 2009.

BCS2220
Period 2
28 Oct 2024
13 Dec 2024
Print course description
ECTS credits:
4.0

Computer Science

Coordinator:

- A.J. Garnock-Jones

Teaching methods:
Project-Centered Learning
Dept. of Advanced Computing Sciences

Embedded Programming

Full course description

This course starts by introducing the students to CPU architectures (circuits, adders, multipliers, floating-point units, RAM) and then introduces the hardware description language VHDL. Students will get to practice design, implementation and debugging while they create a circuit that, for example, can add two numbers. Near the end of the course, when the C programming language has been introduced in parallel skill classes, the step to microcontroller programming will be made, discussing the advantages and limitations of micro-controllers and micro-processors, and allowing students to get some initial experience with limited general-purpose circuits

Prerequisites

None.

BCS2410
Period 4
27 Jan 2025
28 Mar 2025
Print course description
ECTS credits:
4.0
Coordinator:

- C. Kouzinopoulos

Teaching methods:
Project-Centered Learning
Dept. of Advanced Computing Sciences

Computer Security

Full course description

Computer security is the process of detecting and preventing unauthorized and illicit access to a computer. As information systems have become mandatory in the commercial world, coupled with the increased frequency of security incidents, organizations now recognize the need for a comprehensive security strategy. This course will introduce a wide range of topics in such as security concepts and services, physical, operational, and organizational security, the role of people in systems security, an introduction to cryptography and the public key infrastructure, computing systems hardening, secure code, and secure applications development.

Computer Science

Prerequisites

None.

BCS2420
Period 4
27 Jan 2025
28 Mar 2025
Print course description
ECTS credits:
4.0
Coordinator:

- A.R.K. Sai

Teaching methods:
Project-Centered Learning
Dept. of Advanced Computing Sciences

Parallel Programming

Full course description

Parallel programming is the paradigm of doing computations on a computer in parallel. This is possible, since nowadays almost all computer systems include so-called multi-core chips. To exploit the full performance of such systems, parallel programming needs to be emploed.

This course covers shared-memory parallelization with OpenMP as well as parallelization with message passing on distributed-memory architectures with MPI. The course starts with a recap of the programming language C followed by a brief theoretical introduction to parallel computing. Next, the course treats theoretical aspects like MPI communication, race conditions, deadlocks, efficiency as well as the problem of serialization. The course is accompanied by practical labs in which the students have the opportunity to apply the newly acquired concepts.

Prerequisites

None.

Recommended reading

Recommended literature: Eijkhout: The Art of HPC Vol. 1 & 2 (https://theartofhpc.com/)

Additional literature: Pacheco: An Introduction to Parallel Programming

BCS2430
Period 4
27 Jan 2025
28 Mar 2025
Print course description
ECTS credits:

Computer Science

4.0
Coordinator:

- B. Küppers

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam
Dept. of Advanced Computing Sciences

IT Management & Privacy

Full course description

This course will provide the student with a general introduction to enterprise information systems and the functionality of specific enterprise IS types (e.g., Enterprise Resource Planning, Customer Relationship Management, Supply Chain Management, Executive IS, Product Lifecycle Management systems). The students will become familiar with System Development Life Cycle (SDLC), different adaptation and transition approaches, project management approaches, as well as issues related to post-implementation, such as maintenance. Additionally, the course will provide the student with an introduction to the legal and technological aspects of EU and global data protection and privacy issues, and a business understanding of data usage practices, including the GDPR. Students will learn to grasp the key actors, concepts, and obligations of the GDPR, and develop awareness and understanding of the legal requirements they will encounter in their professional careers. The general introduction of the aspects of the GDPR is concretized by several examples. Students will learn how to become ambassadors for a global digital economy and society by learning to approach the processing of personal data competently and ethically. The concepts of digital ethics and accountability along with their limitations are illustrated through different real-life cases and scenarios

Prerequisites

IT Management & Privacy

BCS2510
Period 5
31 Mar 2025
23 May 2025
Print course description
ECTS credits:
4.0
Teaching methods:
Project-Centered Learning
Dept. of Advanced Computing Sciences

Numerical Methods

Computer Science

Full course description

Numerical methods are techniques for solving problems from continuous mathematics (calculus and linear algebra) with the aid of a digital computer. In this course, we will cover the fundamental concepts of numerical mathematics, including the floating-point representation of real numbers, truncation and round-off errors, iterative methods and convergence. We will study the simplest and most important methods for core problems of continuous mathematics, namely the solution of algebraic equations and differential equations, interpolating data by polynomials, numerically estimating derivatives and integrals, approximating functions by polynomials and trigonometric series, solving systems of linear algebraic equations and computing eigenvalues. There will be a strong practical component, with students being expected to write their own numerical code and test the performance and suitability of different methods on various problems.

Prerequisites

Desired prior knowledge: Calculus, Linear Algebra

Recommended reading

Recommended literature: J.D. Faires & R. Burden, "Numerical Methods", International 4th Edition, Cengage, 2012; ISBN: 978-0-495-38569-1.

Additional literature: C.F. Gerald & P.O. Wheatley, "Applied Numerical Analysis", Seventh Edition, Pearson, 2003; ISBN: 0-321-13304-8. T. Siauw & A.M. Bayen, "An Introduction to Matlab Programming and Numerical Methods for Engineers", Academic Press, 2015; ISBN 978-0-12-520228-3.

BCS2540
Period 5
31 Mar 2025
23 May 2025
Print course description
ECTS credits:
4.0
Coordinators:

- P.J. Collins
- M. Boussé
- B. Sakçak

Teaching methods:
Project-Centered Learning
Assessment methods:
Written exam

BSc CS year 2 Elective Modules

Dept. of Advanced Computing Sciences

M2-1: Intelligent Interaction

Full course description

Elective Module Project 2-1 > **M2-1: Intelligent Interaction**

Course: Image and Video Processing (period 2) +
Project: Project 2-1 Human-Computer Interaction (semester 3, bachelor year 2)

Each elective module comprises an elective course, an elective project, and related skill classes.

Second year students **choose one** out **of the two** electives (Module Project 2-1) during semester 3.

- Course Image & Video Processing (period 2): In this course, students will have a brief introduction to basic 2D signals and systems, sampling, image filtering, and computer vision. The course will start with the connection of the camera and the real world with camera calibration techniques. Colour domain processing in different spaces (RGB, CiE, Lab) and its relevance to our visual perception system will be presented. Students learn about linear and non-linear filtering in the spatial domain, segmentation, noise reduction, and smoothing, among others. Frequency domain transforms will be presented (Fourier, DCT), along with their use in filtering for image enhancement, de-noising, restoration, and the understanding of standards like JPEG. Video analysis will be introduced, with a focus on motion estimation and its relevance to compression standards like MPEG. The course will end with an introduction to recent computer vision systems with deep learning techniques for image recognition, object detection, and scene understanding.
- Project 2-1 Human-Computer Interaction: The central topic of this project is to develop and to evaluate a real-time human-computer interface using the parallax effect (https://en.wikipedia.org/wiki/Parallax) to generate the illusion of a 3-dimensional interactive scene. The project lets you practice the development of software that must run efficiently on decent computer hardware as well as the generation of interactive computer graphics that must react in real-time. You further practice the use of one of the most popular computer vision libraries: OpenCV, and apply your skills in performing user studies to evaluate your self-developed human-computer interface. You will use GitHub to host your code and effectively collaborate with your teammates, and end up with a project that you can proudly display on your GitHub profile as part of your portfolio of programming work. You are explicitly allowed to choose from a number of programming languages including Java, Python, C++ and computer graphics libraries.

Prerequisites

Students must have passed Project 1-1. Furthermore, the student has to have passed at least two out of the following three courses: Procedural Programming, Objects in Programming, and Data Structures and Algorithms. The student furthermore needs to be registered for or has already completed at least three courses of the programme in year 2, semester 1. This project is a prerequisite for Project 3-1.

- Project 1-1

BCS2710
Semester 1

Computer Science

2 Sep 2024
24 Jan 2025
Print course description
ECTS credits:
10.0
Coordinators:

- G. Tang
- O. D'Huys

Teaching methods:
Project-Centered Learning, Work in subgroups
Assessment methods:
Written exam, Presentation
Dept. of Advanced Computing Sciences

M2-1: AI and Machine Learning

Full course description

Elective Module Project 2-1 **> M2-1: Artificial Intelligence and Machine Learning**

Course: Machine Learning (period 2) +
Project: Project 2-1 Adaptive Systems (semester 3, bachelor year 2)

Each elective module comprises an elective course, an elective project, and related skill classes.

Second year students **choose one** out **of the two** electives (Module Project 2-1) during semester 3.

- Course Machine Learning (period 2): Machine learning is a major frontier field of artificial intelligence. It deals with developing computer systems that autonomously analyse data and automatically improve their performance with experience. This course presents basic and state-of-the-art techniques of machine learning. Presented techniques for automatic data classification, data clustering, data prediction, and learning include Decision Trees, Bayesian Learning, Linear and Logistic Regression, Recommender Systems, Artificial Neural Networks, Support Vector Machines, Instance-based Learning, Rule Induction, Clustering, and Reinforcement Learning. Lectures and practical assignments emphasize the practical use of the presented techniques and prepare students for developing real-world machine-learning applications.
- Project 2-1 Adaptive systems: In this project, you will use Artificial Intelligence (AI) and Machine Learning (ML) techniques to control autonomous agents in a real-time video game engine. There will be a strong emphasis on combining and using existing libraries and tools (e.g., a video game engine, and existing implementations of AI and ML techniques), and building your own new code around it. You will use GitHub to host your code and effectively collaborate with your teammates, include clear documentation (for yourselves and others) on how to correctly install any dependencies, and end up with a project that you can proudly display on your GitHub profile as part of your portfolio of programming work. The used programming languages will be C# and Python.

Computer Science

Prerequisites

Prerequisites: Students must have passed Project 1-1. Furthermore, the student has to have passed at least two out of the following three courses: Procedural Programming, Objects in Programming, and Data Structures and Algorithms. The student furthermore needs to be registered for or has already completed at least three courses of the programme in year 2, semester 1. This project is a prerequisite for Project 3-1.

- Project 1-1

BCS2720
Semester 1
2 Sep 2024
24 Jan 2025
Print course description
ECTS credits:
10.0
Coordinators:

- O. D'Huys
- E.N. Smirnov

Teaching methods:
Project-Centered Learning, Work in subgroups
Assessment methods:
Written exam, Presentation
Dept. of Advanced Computing Sciences

M2-2: High Performance Computing

Full course description

Elective Module Project 2-2 **> M2-2: High Performance Computing**

Course: High Performance Computing (period 5) +
Project: Project 2-2 High Performance Computing (semester 4, bachelor year 2)

Each elective module comprises an elective course, an elective project, and related skill classes.

Second year students **choose one** out **of the three** electives (Module Project 2-2) during semester 4.

In this project, students will tackle the particle physics problem of particle track reconstruction, in which they will attempt to reconstruct the trajectories (tracks) of particles as they leave energy deposits in a Large Hadron Collider detector. They will develop a pattern recognition model and minimize the error on found tracks by applying filtering techniques. To efficiently solve track reconstruction, they will design and implement data reductions and a data-parallel approach to tracking on GPU architectures. Students will learn a systematic approach to performance engineering by iteratively designing, developing, and testing a high throughput solution to tracking.

Computer Science

Prerequisites

Students must have passed Project 1-2. Furthermore, the student has to have passed at least two out of the following three courses: Procedural Programming, Objects in Programming, and Data Structures and Algorithms. The student furthermore needs to be registered for or has already completed at least three courses of the programme in year 2, semester 2. This project is not a prerequisite for another project / course.

- Project 1-2

BCS2730
Semester 2
27 Jan 2025
20 Jun 2025
Print course description
ECTS credits:
10.0
Coordinators:

- C. Kouzinopoulos
- K. Schneider

Teaching methods:
Project-Centered Learning, Work in subgroups
Dept. of Advanced Computing Sciences

M2-2: Cybersec.&IoT-Information Security

Full course description

Elective Module Project 2-2 > **M2-2: Cybersec.&IoT-Information Security**

Course: Information Security (period 5) +
Project: Project 2-2 Cybersecurity & IoT (semester 4, bachelor year 2)

Each elective module comprises an elective course, an elective project, and related skill classes.

Second year students **choose one** out **of the three** electives (Module Project 2-2) during semester 4.

During this project, students will build a demonstrator of a secure distributed computational process using limited power hardware devices and low power data collection. They will develop a distributed approach to analytics by avoiding the need to centralize data, but instead working around the size, heterogeneity, and ownership of data by dynamically deploying micro-services in a federated learning setup from the service repository to the IoT edge on low power computational models. Attention will be paid to data security and privacy.

Prerequisites

Prerequisites: Students must have passed Project 1-2. Furthermore, the student has to have passed

at least two out of the following three courses: Procedural Programming, Objects in Programming, and Data Structures and Algorithms. The student furthermore needs to be registered for or has already completed at least three courses of the programme in year 2, semester 2. This project is not a prerequisite for another project / course.

- Project 1-2

BCS2740
Semester 2
27 Jan 2025
20 Jun 2025
Print course description
ECTS credits:
10.0
Coordinator:

- A.R.K. Sai

Teaching methods:
Project-Centered Learning, Work in subgroups
Dept. of Advanced Computing Sciences

M2-2: Cybersec.&IoT-Ubiquitous Comp.&IoT

Full course description

Elective Module Project 2-2 **> M2-2: Cybersec.&IoT-Ubiquitous Comp.&IoT**

Course: Ubiquitous Computing & Internet of Things (period 5) +
Project: Project 2-2 Cybersecurity & IoT (semester 4, bachelor year 2)

Each elective module comprises an elective course, an elective project, and related skill classes.

Second year students **choose one** out **of the three** electives (Module Project 2-2) during semester 4.

Students work on a project assignment in small groups of about six students. The concrete assignment is defined by the students given some umbrella topics that match the courses in the curriculum. Students indicate their umbrella topic preference at the beginning of period 2.4. The group composition stays the same for the whole project and is based on the topic preferences of the students. A least regret algorithm is used to ensure that the overall regret is minimized. Throughout the project, the groups are guided by a tutor with respect to the project management. In periods 2.4 and 2.5, the students work on the project, while also having to attend the courses of these periods. They meet their tutor approximately biweekly. In period 2.6, the students work three weeks full-time on the project and meet their tutor about twice a week.

The focus of this project lays on the project planning and communication of progress and results. During the project, the students have to hand in several deliverables such as a project plan after a few weeks from the start or the implemented code at the end of the project. Formative feedback sessions with the examiners on intermediate project plans will add to the quality of feedback the

students receive. Presentations throughout the project will be used to communicate the progress to the examiners.

Prerequisites

Students must have passed Project 1-2. Furthermore, the student has to have passed at least two out of the following three courses: Procedural Programming, Objects in Programming, and Data Structures and Algorithms. The student furthermore needs to be registered for or has already completed at least three courses of the programme in year 2, semester 2. This project is not a prerequisite for another project / course.

- Project 1-2

BCS2750
Semester 2
27 Jan 2025
20 Jun 2025
Print course description
ECTS credits:
10.0
Coordinator:

- G. Tang

Teaching methods:
Project-Centered Learning, Work in subgroups