# IX. Course Descriptions

## 1. Modules in the First Year

In the following, the modules for the first year are described.

### 1820113 Mathematics for IT

**Learning Outcomes**

- understand mathematical concepts and develop skills regarding the approach towards mathematical thinking and discussions
- apply methods of axiomatic set theory and applied mathematics in the context of computer science
- describe natural science, technical and computer science aspects with the mathematical methods of linear algebra and analysis
- apply numeric proceedings and gain knowledge according to software and the usage of the software

**Content Description**

Topics are sets and relations, real numbers, functions, exponential functions, inverse functions and logarithms, limits, continuity, derivatives, chain rule, linearization, extreme values of functions, monotonic functions, convexity, antiderivatives, Riemann sum, indefinite integrals and the substitution method, sequences, infinite series, geometric series, Taylor series, functions of severable variables, partial differentiations, and gradient. Key aspects in this module will be:

- Set and Relations
- Sequences and Series
- Functions
- Differentiation
- Integration

**Recommended Literature:**

- S. Abbott, Understanding Analysis (Undergraduate Texts in Mathematics), 2nd ed. 2015, Corr. 2nd printing 2016 Edition. New York: Springer, 2016.
- K. A. Ross, Elementary Analysis: The Theory of Calculus, 2nd ed. 2013 Edition. New York: Springer, 2013.
- D.W. Cunningham: Set Theory: A First Course (Cambridge Mathematical Textbooks), Cambridge University Press, 2016
- Comment by the authors: Mathematics for Computer Science is different than mathematics for engineers. There is a big area of overlap, however, basics of sets and relations have to be included and applied.

## 1820101 Introduction to Computer Science

**Learning Outcomes**

- understand the structure and services of the system environment of a computer
- work with the general environment of a personal computer and are familiar with the Windows operating system
- understand and operate a UNIX system

**Content description**

The course provides a comprehensive introduction to computers and computer science as a problem-solving discipline. Topics covered in the course are: technical representation of information, digital number systems, basic architecture of processors, components of a computer system, connections and peripheral devices, memory hierarchy (cache, virtual memory), processing of instructions in a simple processor (data path, control), role and basics of operating systems (windows and Linux), problem solving and programming languages basic structures of data processing systems,

**Recommended Literature:**

Comment by the authors: More focus on the application of Windows and Linux as well as practice in the lab.

## 1820111 Introduction to Computer Science Lab

**Learning Outcomes**

- apply and practice of the theoretical concepts of the module Introduction to Computer Science, especially Windows and Linux.

**Content Description**

Exercise and practical application of the theory regarding the operating systems Windows and Linux.

## 1820102 Fundamentals of Programming

**Learning Outcomes**

- understand the different paradigms of programming (procedural, object oriented, functional) and the difference between static and dynamic programming languages
- apply appropriately different Datatype and control structures
- compose small software modules

**Content Description**

The course focuses on software, algorithmic problem solving and fundamentals of programming (procedural, object oriented, functional). It communicates the basic principles of programming (static and dynamic) as well as elementary abstraction mechanisms of software development. Overview of software development and its importance, technical and formal basics of programming, basic linguistic aspects (syntax and semantics of programming languages), Introduction to programming (value, elementary data types, function, variable, condition, control structures, statement, procedure)

**Recommended Literature:**

- W. Savitch, Java: An Introduction to Problem Solving and Programming, 8th ed. Boston: Pearson, 2017.
- Y. D. Liang, Introduction to Java Programming and Data Structures, Comprehensive Version, 12th ed. Hoboken, NJ: Pearson, 2019
- R. Nixon, Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5 (Learning PHP, MYSQL, JavaScript, CSS & HTML5), O'Reilly media, 2018

**Digital Assistance**

- Pearson Introduction to Java Programming and Data Structures
  Comprehensive Course Contents including LMS integration
- W3 Schools Java Tutorial
- C. Kerningham, Ritchie: C Programming Language, Perason, 1988

## 1820112 Fundamentals of Programming Lab

**Learning Outcomes**

- write small programs
- choose the right datatype, data structures and control structures according to the programming language

**Content description**

Application and practice of the theoretical concepts of the module *Fundamentals of Programming* the students the students apply the theory in small programming exercises.

**Additional Comment**

We would recommend focussing only on a small amount of languages, e.g. one per paradigm (procedural, object-oriented, functional) and one dynamic programming language like JavaScript or PHP. Here we recommend Java, C and PHP. The focus should be to apply the programming language and gain programming experience instead of knowing a lot of different languages on a sketchy high level. In the later course of study Python programming is included, e.g., with Data Science, Artificial Intelligence, Data Mining.

## 1820205 Object Oriented Programming

**Learning Outcomes**

- understand the basic concepts of object-oriented programming
- develop and implement their program
- perform the according tests including debugging. They can work with common development tools in the environments.
- apply basic design concepts to develop their programs and are able to apply example structures
- analyse and evaluate existing programs

**Content description**

Students learn the practical aspects of designing, implementing, and debugging object-oriented software (e.g. Java). Topics covered include reusing design patterns and developing, documenting, and testing representative applications using object-oriented frameworks and Java. Design and implementation based on design patterns and frameworks are the central themes to enable the construction of reusable, extensible, efficient, and maintainable software. There's a lab included

**Recommended Literature:**

- Handbook of Software Construction, Microsoft Press, 2004
- R.C. Martin, Clean Code: A handbook of Agile Software Craftsmanship, Pearson,2008
- Y. Liang, Y.D. Liang, Introduction to Java Programming and Data Structures, Comprehensive Version, Pearson, 2019
- S. McConnell: Code Complete: A Practical

### Additional Comments

Since Software Engineering is taught in the second year the focus is on programming, IDE, documentation, testing, frameworks. Parallel there is the module Data Structures and Algorithms I, where programming practice can be gained extensively

## 1820142 Linear Algebra for IT

### Learning Outcomes

- ability to choose and apply the mathematical methods for modelling their problem.
- know the basic concepts of the mathematical aspects in regard to problems relevant for computer science.
- choose according methods to address the relevant aspects.
- model their computer science problems in mathematical terms
- apply and implement software-oriented applications to solve the problems.

### Content description

Students learn systems of linear equations, Gaussian elimination, invertible matrices, diagonal matrices, determinants by cofactor expansion, vectors in 2-space, 3-space, and n-space, norm, dot product, distance, orthogonality, real vector spaces, subspaces, linear independence, basis, null space, rank, eigenvalues, eigenvectors, diagonalization, orthogonal diagonalization, singular value decomposition, and quadratic forms. Key aspects in this lecture will be: Vector Spaces, Linear Systems, Matrices (and their properties), Linear Mappings

### Recommended Literature:

- K. Singh, Linear Algebra: Step by Step, Illustrated Edition Oxford, United Kingdom: Oxford University Press, 2013.
- S. Axler, Linear Algebra Done Right, 3rd ed. 2015 Edition. New York: Springer, 2014.

**Digital Assistance:**

- Open Course Ware (class on Linear Algebra only, free of charge)

**Additional Comments:**

The numeric Course in the proposal from Prof. Schuster (the GIZ expert) is not recommended by the authors. It is especially necessary for Computer Graphics which is not contained in the curriculum. Numeric knowledge is in typical application software development very seldom necessary.

## 1820203 Data Structures and Algorithms I

**Learning Outcomes**

- understand the basic algorithms for common problem in computer science and can apply the algorithms in the context of the problems.
- choose the right algorithm for their problem and implement the algorithm.
- evaluate the complexity of algorithms and take the complexity into account in problem solving to implement fast regarding their problems.

**Content description**

This course introduces basic algorithms design and encompasses abstract thinking, problem solving, using pseudo-code and implementing data structures and algorithms with C and Java.

Topics are asymptotic analysis of algorithms complexity followed by introduction of elementary data structures and their operations mainly Lists, Linked lists, Stacks, Queues, Sequences, and trees and their algorithms. Further Sorting, searching algorithms.

**Recommended Literature:**

- Sedgewick, Wayne: Algorithms, Addison-Wesley Professional;2011

- Y. D. Liang, Introduction to Java Programming and Data Structures, Comprehensive Version, 12th ed. Hoboken, NJ: Pearson, 2019.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 3rd Edition, 3rd ed. Cambridge, Mass: The MIT Press, 2009

**Digital Assistance**

- Pearson Introduction to Java Programming and Data Structures Comprehensive Course Contents including LMS integration
- Algorithm and Data Structures Tutorials I
- Algorithm and Data Structures Tutorials II
- Algorithm and Data Structures Tutorials III

## 1820213 Data Structures and Algorithms I Lab

**Learning Outcomes**

- develop practical experience in algorithms and data structure.
- choose the right data structure and algorithm for their problem
- implement complex algorithms

**Content description**

The students solve small programming tasks according to the topic of the module Data Structures and Algorithms I

**Additional Comments**

The chosen language should be one of the languages used in Module/Course Fundamentals of Programming, e.g. Java. Since there is no separate programming course, the programming practice must result from the module Fundamentals of Programming (1820102) together with Data Structures and Algorithms I and II. Dynamic Language experience like JavaScript or PHP can be gained in Web engineering. A goal should be to gain very good programming experience with data structure and algorithm knowledge in mind.

## 1820162 Web Engineering

**Learning Outcomes**

- understand different models
- select the appropriate method for typical web applications.

**Content description**

Students are introduced to web applications development frameworks and tools and best practices. All development life cycles related to web is covered and HTML and CSS should be covered as well as client-side/ server-side programming languages.  Students work on projects to have good exposure to activities that enhance their web development skills

**Recommended Literature:**

- J. Robbins, Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and eb Graphics, O'Reilly

# Modules in the Second Year

## 1820208 Data Structures and Algorithms II

**Learning Outcomes**

- list the basic algorithms for common problem in computer science.
- apply and implement the according algorithms and data structures to solve the problems.
- define the complexity of algorithms a take it into account in their problem solving to implement fast solutions

**Content description**

This course is a continuation of Data Structures & Algorithms I. Topics are Dictionaries, Heaps, and graph theory. The course focuses on design of data structures and relevant common algorithms and their complexity analysis. Brute Force, Dynamic programming, Divide and conquer, decrease, and conquer, and greedy algorithms design techniques and strategies are introduced

Optional: introduction of genetic and evolutionary algorithms

**Recommended Literature:**

- Sedgewick, Wayne: Algorithms, Addison-Wesley Professional;2011
- Y. D. Liang, Introduction to Java Programming and Data Structures, Comprehensive Version, 12th ed. Hoboken, NJ: Pearson, 2019.
- T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 3rd Edition, 3rd ed. Cambridge, Mass: The MIT Press, 2009
- E. Eiben and J. E. Smith, Introduction to Evolutionary Computing, 2nd ed. 2015. Heidelberg: Springer, 2015.

**Digital Assistance**

- Pearson Introduction to Java Programming and Data Structures
  Comprehensive Course Contents including LMS integration
- Algorithm and Data Structures Tutorials I
- Algorithm and Data Structures Tutorials II
- Algorithm and Data Structures Tutorials III

## 1820218 Data Structures and Algorithms II Lab

**Learning Outcomes**

- develop their practical experience in Algorithms and data structure.
- choose the right data structure and algorithm for their problem and can
- implement a complex algorithm.

**Content description**

The students solve small programming tasks according to the topics of Data Structures and Algorithms II

## 1820210 Operation Systems

**Learning Outcomes**

- classify components and their tasks of modern operating Systems how they interact together.
- write small programs for Unix (Linux), Windows or Android.
- compare the different operating systems and know the differences

**Content description**

Students will be introduced in the basic concepts, design, and implementation of modern operating systems for PC and mobile systems. Topics to be covered include operating system structures, process management, memory management, file systems and input/output systems. Concurrent programming and primitives will be covered with hands on using UNIX, windows and Android based systems. Comparative study between various modern operating systems will be conducted.

**Recommended Literature:**

- Tanenbaum, H. Bos: Modern Operating Systems, Pearson 2014
- W. Stallings: Operating Systems: Internals and Design Principles, Pearson, 2017

## 1820220 Operation Systems Lab

**Learning Outcomes**

- apply and practice the theoretical concepts of the course 1820210 Operation Systems.

**Content description**

The students solve small tasks according to the topics of Operating Systems, 1820210

## 1820216 Automata Theory and Compilers

**Learning Outcomes**

- understand the construction of compilers and other language processing tool
- construct grammars for regular and context free languages and construct the appropriate automaton.
- apply tools to generate Scanners and Parsers.

**Content description**

Students were introduced in some fundamental concepts in automata theory and formal languages including   regular and context free languages including DFAs and NFAs. Scanning and parsing, Finite state machines and push down automata. Students learn how compilers and language processing tools work (e.g. programming environments, XML-processors) and deepen their understanding of programming and languages. Context free analyses: parsing, parser generation, design and use of abstract syntax, processing of XML documents.

Optional:  compilation techniques for procedural and object-oriented languages

**Recommended Literature:**

- J.E. Hopcroft et al: Introduction to Automata Theory, Languages, and Computation, Pearson, 2008
- P. Linz: An Introduction to Formal Languages and Automata, Jones & Bartlett Learning,2016
- A.V. Aho;Lam, R. Sethi , Jeffrey D. Ullman:  Compilers,  Pearson, 2013

## 1820201 Mathematics for Data Sciences

**Learning Outcomes**

- formulate the mathematical foundations for data science
- apply the mathematical foundations for data science
- understand the knowledge areas:  probability axioms, probability models, testing hypotheses, Markovian chain
- apply of methods learned in lecture using programming languages such as R and Python to experience real world scenarios

**Content description**

Students were introduced to statistical programming, sample spaces, probability in discrete and continuous sample spaces, sampling and repeated trials, axioms of probability, conditional probability, Bayes theorem, random variables, expectation, variance, probability models (binomial, Poisson, geometric, normal or Gauss, exponential and chi-square), bivariate random variables, correlation, Markov chain, statistical inference, likelihood function, maximum likelihood estimation, posterior distribution, Bayes estimation, and testing hypotheses. Key aspects in this lecture will be Probability Axioms, Probability Models, Testing Hypotheses, Markovian Chain.

Optional: Application of methods learned in lecture using programming languages such as R and Python to experience real world scenarios

**Recommended Literature:**

- R. Larsen and M. Marx, Introduction to Mathematical Statistics and Its Applications, An, 6th ed. Boston: Pearson, 2017.
- K. Stapor, Introduction to Probabilistic and Statistical Methods with Examples in R, 1st ed. Springer, 2020.

**Digital Assistance**

- https://www.datacamp.com

**Additional Comments:**

This Module/course is important for Data Sciences and Artificial Intelligence subjects.

## 1820109 Software Engineering

**Learning Outcomes**

- understand the foundations of the software development process. They can analyse the problem and develop computer supported approach. The approach is implemented and documented based on quality assurance aspects.
- compare different approaches and evaluate the approach, as well as the development of necessary changes.
- work individually and in teams to solve complex problems in projects.
- apply tools for software development
- project management and economic aspects in the software development projects.

**Content description**

This course introduces software engineering as a discipline, discusses stages of the software life cycle, compares development models such as waterfall, agile development, v-model, prototyping and incremental/iterative, covers requirements analysis, effort and cost estimation, compares structured and object-oriented analysis and design methods

**Recommended Literature:**

- Sommerville: Software Engineering, Pearson, 2015
- R. C. Martin, Clean Architecture: A Craftsman’s Guide to Software Structure and Design, 1st ed. Pearson, 2017.
- R.C. Martin, Clean Code: A handbook of Agile Software Craftsmanship, Pearson, 2008

## 1820119 Software Engineering lab

**Content description**

In this course introduces students to construct software of high quality. Principles of software engineering are accompanied by a student project in order to integrate practical experiences.

## 1820303 Database I

**Learning Outcomes**

- refine about architectures and components of database systems
- apply design concepts of database design and implement a database system
- understand database evaluation
- apply database normal form concepts
- classify transaction concepts

**Content description**

In this course students get an introduction to database systems. Starting with a historical retro perspective the main focus will be on relational databases. Beginning with architectures and design concepts including the application of a modelling language for database design. The introduction of the structured query language (SQL) in the context of data definition language and data manipulation language. The transaction concept is embedded in the perspective of multi-user and distributed database environments.

**Recommended Literature:**

- Ramez A. Elmasri, Shamkant B. Navathe: (current edition) Fundamentals of Database System, Pearson
- Petrov, A. (2019). Database Internals: A Deep Dive into How Distributed Data Systems Work. O'Reilly Media; 1. Edition

**Digital Assistance**

https://www.w3schools.com/sql/

## 1820333 Database I lab

**Learning Outcomes**

- implement database concepts and perform inquiries in the database
- understand the interaction between domain experts and database experts

**Content description**

The students solve small design and implementation tasks based on the theory according to the topic of the module Database I, and Software Engineering 1820109.

**Digital Assistance**

https://www.w3schools.com/sql/

## 1820281 Introduction to Networks

**Learning Outcomes**

- analyse of the communication and network technology
- compare network services and network protocols

- understand how computer architecture, networks, digital technology and programming are connected as a foundation for software architecture.

**Content description**

Students learn the principles, design, implementation, and performance of computer networks, Internet protocols and routing, local area networks, TCP, performance analysis, congestion control, switching and routing, mobile IP, peer-to-peer overlay networks, network security, and other current research topics

**Recommended Literature:**

- A.S. Tanenbaum, D.J. Wetherall: Computer Networks, Pearson, 2010 (digital 2020)
- J. Kurose, K. Ross: Computer Networking: a Top-Down Approach, Pearson, 2016

## 1820282 Introduction to Networks Lab

**Learning Outcomes**

- enhance their knowledge with practical experience in network and the communication technology.

**Content description**

The students solve small tasks according to the topics of Introduction to Networks, 1820108

# Modules in the Third Year

## 1820311 Computer Organization & Assembly

**Learning Outcomes**

- understand of the basic concepts of computer architecture and services as well as the functionality
- apply concepts and develop concepts in machine programming code
- evaluate the capacity and benefit for a practical application.

**Content description**

This course is designed to introduce students to the basics of Computer organization and low-level programming using Assembly language. Computer organization includes examining the hardware of a computer and figuring out how it works. Computer functions, Central Processing Unit Organization, Bus interconnection, Cache memory, internal memory, External memory, Input Output. Assembly language in the lab will focus on current processor assembly language, number representations, low-level data manipulation, instructions, and directives. The usage of assemblers, debuggers, linkers and loaders are also covered topics

**Recommended Literature:**

- D.A. Patterson, J.L. Hennessy: Computer Organization and Design (The Morgan Kaufmann Series in Computer Architecture and Design) , Morgan Kaufmann, 2016
- Tanenbaum, T. Austin: Structured Computer Organization

## 1820341 Computer Organization & Assembly Lab

**Learning Outcomes**

- apply practical experience and their understanding of basic processes in a computer in a practical approach.

**Content description**

The students write small programs in assembler and solve tasks according to the topics of Computer Organization and Assembly

## 1820312 Distributed and Cloud Computing

**Learning Outcomes**

- diagnose the basic concepts of distributed and cloud computing
- understand the aspects and settings of modern cloud-based infrastructures
- apply concepts and set up environments according to the demands
- evaluate the benefit for a practical application

**Content description**

This course covers fundamental models for distributed systems, like inter process communication, synchronization, consistency, replication, cloud computing, fault tolerance and security in a distributed system. Foundation for the underlying architecture and parallel systems (SMP, Cluster-System)- Cloud Architecture / Grid computing- storage systems (Storage Area Network (SAN) und Network Attached Storage (NAS), etc.)

**Recommended Literature:**

- Coulouris, J.Dollimore, T.Kindberg: Distributed Systems: Concepts and Design, Pearson
- A.S. Tanenbaum, M.v.Steen: Distributed Systems: Principles and Paradigms, CreateSpace Independent Publishing Platform, 2016
- M. v. Steen, A.S. Tannenbaum: Distributed Systems, CreateSpace Independent Publishing Platform, 2017
- B. Scholl, T. Swanson, and P. Jausovec, Cloud Native: Using Containers, Functions, and Data to Build Next-Generation Applications, 1. Ed.. O'Reilly Media, 2019.

## 1820322 Distributed and Cloud Computing lab

**Learning Outcomes**

- apply practical experience in cloud computing and distributed computing.
- choose the right concept for their problem and can implement it.

**Content description**

The students write small programs solve tasks according to the topics of Distributed and Cloud Computing

## 1820313 Data Security& Data Privacy

**Learning Outcomes**

- prescribe the main aspects of security in the It environment
- analyse and develop counter measures to create and support an IT security concept
- evaluate strength and weaknesses and implement the according measures
- measure the consequences of dealing with critical and important data and base their decisions and action on this knowledge

**Content description**

This course covers aspects regarding security threads and according to security concepts and measures. The students learn to evaluate infrastructures and cryptography concepts. The students learn about legal aspects of data privacy.

**Recommended Literature:**

- W.Stalling, L.Brown: Computer Security: Principles and Practice, Pearson, 2017
- Jonathan Katz, Y. Lindell, Introduction to Modern Cryptography, Chapman & Hall CRC Press, Cryptography and Network Security - M. Bishop: Computer Security, Addison-Wesley-Longman

- W. Stallings, L. Brown: Computer Security: Principles and Practice, Pearson * Education
- C. Pfleeger, S. Lawrence Pfleeger, Security in Computing - Laurens Van Houtven, Crypto 101, www.crypto101.io
- Ivan Ristic, Bulletproof SSL, and TLS, Feisty Druck

## 1820323 Data Security Lab

**Learning Outcomes**

- apply practical experience in data security and data privacy.
- choose the right concept for their problem and implement it.

**Content description**

The students write small programs in assembler and solve tasks according to the topics of Data Security& Data Privacy

## 1820314 Mobile Computing

**Learning Outcomes**

- determine project development concepts in the area of mobile and web engineering
- implement and test application in Python und JavaScript
- apply and implement App development concepts
- understand the exchange formats
- evaluate the concepts for their practical approach

**Content description**

Students learn Python and foster their knowledge in JavaScript. The learn to use current frameworks and tools used in mobile applications. They become familiar in data exchange formats (JSON, XML, etc.)- platforms and tools, App-development. All development life cycles (including agile development) related to web and mobile applications is covered. and students work on projects to have good exposure to activities that enhance their web and mobile development skills.

**Recommended Literature:**

- E. Matthes: Python Crash Course, 2nd Edition: A Hands-On, Project-Based Introduction to Programming, No Starch Press 2019
- M.Lutz, Learning Python, O'Reilley, 2013

**Additional Comments**

Tools and framework changes in this area so fast, that it is generic formulated. Therefore, no further literature is advised. It must be selected depended on the taught topics.

## 1820324 Mobile Computing

**Learning Outcomes**

- Apply the practical experience in mobile computing.
- choose the right concept for their problem and can implement it.

**Content description**

Students work on projects to have good exposure to activities that enhance their web and mobile development skills

## 1820321 Artificial Intelligence

**Learning Outcomes**

- understand the basic concepts and technologies of artificial intelligence
- prescribe:  decision trees, neuronal networks, ensemble models and deep learning
- conduct programs using Python programming

**Content description**

This course will introduce the basic concepts and techniques known as components of artificial intelligence. By the end of this course, students will know how to create autonomous agents that efficiently make decisions in fully informed, partially observable, and adversarial situations. The course teaches how agents can draw inferences in uncertain environments and optimize actions for arbitrary reward structures. Key aspects in this lecture will be: Relational Database Service (Amazon RDS), Amazon DynamoDB, Amazon Redshift, and Amazon Aurora, Load Balancing and Lambda). Further topics are Decision Trees, Neuronal Networks, Ensemble Models and Deep Learning.

**Recommended Literature:**

- S. Russel, P. Norvig: Artificial Intelligence: A Modern Approach (Pearson Series in Artificial Intelligence), Pearson. 2020
- Geron: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, O'Reilly Media, 2019

**Digital Assistance:**

http://aima.cs.berkeley.edu/

**Additional Comments**

Python is used in *Mobile Computing*, in *Artificial Intelligence* and in Machine *Learning and Data Mining*. Therefore, a good practice in Python should be reached by the students.

## 1820331 Artificial Intelligence Lab

**Learning Outcomes**

- apply practical experience in artificial intelligence.
- choose the right concept for their problem and can implement it.

**Content description**

Hands on application of algorithms and techniques taught in lecture. The lab will focus to deliver application of algorithms and models in python including usage of current frameworks (such as Tensor Flow or Scikit learn).

## 1820307 Database II

**Learning Outcomes**

- evaluate the current concepts of database architectures and modern database technologies.
- model data structures and develop storage concepts in modern distributes database environments
- decide the demands and requirements of data warehouse concepts
- understand the development, implementation, and support of data warehouse concepts
- evaluate different technologies and choose the right concept for their problem and implement it

**Content description**

Students are introduced to new and current concepts in the area database architectures and modern database technologies. Students evaluate data storage and distribution options that provided be approaches, like for example NoSQL. They become familiar with the data ecosystem and data warehouse concepts and applications.

**Recommended Literature:**

- Redmond & Wilson: Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement; Pragmatic Programmers, current Ed. - White: Hadoop: The Definitive Guide; O’Reilly, current edition.
- White: Hadoop: The Definitive Guide; O’Reilly, current edition.

- John Wiley: The Data Warehouse Toolkit - William A. Giovinazzo: Data Warehouse Design, Prentice-Hall - Jiawei Han und Micheline Kamper: Data Mining: Concepts and Techniques Morgan, Kaufmann Publishers
- Vaisman/Zimányi: Data Warehouse Systems: Design and Implementation

## 1820317 Database II lab

**Learning Outcomes**

- apply practical knowledge in gathered in the module Database II.
- choose the right data concept for their problem and can implement it.

**Content description**

Students work in small projects on new database technologies, e.g., NoSQL and data warehouse applications.

# Modules in the Fourth Year

## 1820411 Human Computer Interaction

**Learning Outcomes**

- categorize different aspects of Human-Computer-Interaction and know the important guidelines.
- analyse interactive systems in interdisciplinary teams and know the methods for the development of these systems.
- conduct usability methods in the software development process
- evaluate the usability of interactive systems

**Content description**

HCI draws on a variety of disciplinary traditions, including psychology, ergonomics, computer science, graphic and product design, anthropology, and engineering. This course is concerned with designing interactions between human activities and the computational systems (also mobile systems) that support them, with constructing interfaces to afford those interactions, and with the study of major phenomena surrounding them. The emphasis of the course is on practical understanding, Usability Engineering, application and evaluation of HCI concepts and methods, User Experience

**Recommended Literature:**

- B. Schneiderman: Designing the User Interface, Pearson,2016
- H. Sharp: Interaction Design: Beyond Human Computer Interaction, Wiley, 2019
- S. Krug: Don’t make me think, New Riders, 2013
- D. Norman: The Design of Everyday things, Revised and Expanded Edition, Basic Books, 2013
- D. Platt: The Joy of UX: User Experience and Interactive Design for Developers (usability), Addison-Wesley, 2016

## 1820421 Human Computer Interaction lab

**Learning Outcomes**

- apply the concepts learned in the module Human Computer Interaction.
- choose the right concept for their problem and can implement it.

**Content description**

The students will create prototypical human-Computer-Interfaces, apply usability engineering methods and analyse and evaluate human-Computer-Interfaces and interactive systems

## 1820412 Machine Learning and Data Mining

**Learning Outcomes**

- categorize different aspects of machine learning
- understand symbolic and probabilistic learning concepts as well as foundations of neuronal networks
- evaluate and chose the according concepts and technology for applications in data mining
- analyse methods of data analysis, clustering, classification, regression as well as derivation detection
- understand visualization concepts

**Content description**

Students are introduced to different aspects of machine learning. The course covers Students symbolic and probabilistic learning concepts as well as foundations of neuronal networks and deep learning. Concepts and technology for applications in data mining are presented and applied. Key areas are methods of data analysis, clustering, classification, regression as well as derivation detection. In order to share and communicate the results according visualization concepts will be introduced.

**Recommended Literature:**

- Geron: Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, O'Reilly Media, 2019
- A.C. Müller, S. Guido: Introduction to Machine Learning with Python: A Guide for Data Scientists, O’Reilly, 2016
- Tan, Steinbach, Kumar. Introduction to Data Mining, Pearson.
- Han, Kamber. Data Mining: Concepts and Techniques, Morgan-Kaufmann Publishers.
- Ian H. Witten und Eibe Frank, Data Mining, Morgan-Kaufmann Publishers.
- John Wiley: The Data Warehouse Toolkit - William A. Giovinazzo: Data Warehouse Design, Prentice-Hall - Jiawei Han und Micheline Kamper: Data Mining: Concepts and Techniques Morgan, Kaufmann Publishers
- L. Moroney: AI and Machine Learning for Coders: A Programmer's Guide to Artificial Intelligence, O'Reilly Media, 2020

## 1820422 Machine Learning and Data Mining lab

**Learning Outcomes**

- apply practical experience in machine learning and data mining.
- apply data pre-processing
- choose the right concept for their problem and can implement it.

**Content description**

The students solve challenges and implement tasks based on the theory according to the topic of the module Machine Learning and Data Mining.

## 1820461 Advanced Software Engineering

### Learning Outcomes

- analyse methods for verification and validation and can apply it
- categorize the methods to ensure reliability
- review software and can apply the different methods for quality assurance (Metrics, Refactoring, etc.)
- understand the principals and Practices of Software Management

### Content description

This course introduces advanced software engineering topics:  verification/ validation, quality assurance, software reliability, design Patterns, Refactoring, Metrics, Software Management, and new trends in Software Engineering (e.g. TTD, MMD)

### Recommended Literature:

- T. Winters, T. Manshreck, H. Wright: Software Engineering at Google: Lessons Learned from Programming Over Time, O'Reilly, 2020
- L.Crispin, J.Gregory: Agile Testing: A Practical Guide for Testers and Agile Teams, Addison Wesley professional, 2009
- H. Percival: Test-Driven Development with Python: Obey the Testing Goat: Using Django, Selenium, and JavaScript, O'Reilly, 2017
- Martin Fowler, Refactoring: Improving the Design of Existing Code, Adison-Wesley
- Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides, Design Patterns, Addison-Wesley
- Ivar Jacobson, Magnus Christerson, Patrik Jonsson und- ITIL Service Lifecycle Publication Suite: German Translation, TSO Verlag

## 1820471 Advanced Software Engineering Lab

### Learning Outcomes

- apply practical experience in advanced software engineering concepts and project management application.
- choose the approach for their problem and can implement it.

### Content description

Students work in groups on projects about the topics of the module Advanced Software Engineering