kiron

Module Catalogue:

Computer Science[1]

Last update: 16.01.2018, Department Coordinator: Cornelia Gamst, Dipl. Math.

[1] Please note that some courses in this module catalogue may change due to expiration of some MOOCs that are limited in time.

Table of Contents

Computer Science at Kiron

Computing and computer technology are part of just about everything that touches our lives, from the cars we drive, to the movies we watch, to the numerous ways we all interact with each other. Whether you want to create the next hot application or want to make a positive difference in the world by developing innovative technology in science, studying computing gives our students a foundation of knowledge, problem solving skills and habits of logical thinking that will provide them with a competitive edge and a career that will never be obsolete.

The Kiron Computer Science curriculum gives a profound introduction to the mathematical and engineering basics of the discipline and equips students with the tools and knowledge necessary for a successful research, industrial and entrepreneurial career in computing.

Key Learning Outcomes

A series of core courses gives students a thorough foundation in **analysis, linear algebra, programming principles, algorithms, complexity management**, and **computing systems**. It also offers considerable flexibility in selecting more advanced courses on topics such as cryptography, artificial intelligence, statistics and web programming. Interdisciplinary skills in business administration and entrepreneurship are also covered to complement the computer science curriculum.

By successfully completing the computer science track, our students will

- be able to analyze a complex problem, then develop and implement an efficient solution.
- have gained a profound understanding of both hardware and software
- be able to work independently and in a structured manner
- be able to develop comprehensive computer programs as part of a team
- have knowledge of contemporary issues related to Computer Science & Engineering.

Module Overview

Module	Name	Credit Points
PREP_1 PREP_E1 PREP_E2 PREP_Ma	Introduction to Online Learning and Scientific Thinking Academic English: Speaking and Listening Academic English: Writing Papers Mathematics	

PREP_Ph	Physics	
CS_Pr1	Introduction to Computer Science	**8**
CS_Pr2	Object-Oriented Programming with Java	**6**
CS_Pr3	Software Engineering	**6**
CS_Ma1	Linear Algebra	**6**
CS_Ma2	Discrete Structures	**6**
CS_Ma3	Calculus	**6**
CS_Th1	Data Structures	**6**
CS_Th2	Algorithms	**6**
CS_Th3	Theoretical Computer Science	**6**
CS_Sy1	Computer Architecture	**6**
CS_Sy2	Computer Networks	**6**
CS_Sy3	Operating Systems	**6**
CS_Sy4	Databases	**6**
Electives		choose **10**
CS_Pr4	Programming Paradigms	6
CS_Pr5	Web Development	6
CS_Pr6	Object-Oriented Programming with C++	4
CS_Ma4	Stochastics and Statistics	9
CS_Sy5	Electrical Engineering	9
CS_Ad1	Computer Graphics	4
CS_Ad2	Cryptography	3
CS_Ad3	Artificial Intelligence	4
CS_GS1	Business Administration	5
CS_GS2	Entrepreneurship	5
Total		**80+10 = 90 CP**

Semester Overview

Kiron Semester	Cluster	Module Name	Credit Points	Total Workload
PREP	Prep	Introduction to Online Learning and Scientific Thinking		
	Prep	Academic English: Speaking and Listening		
	Prep	Academic English: Writing Papers		
	Prep	Mathematics		

	Prep	Physics		
1	Prog	Introduction to Computer Science	8	**32**
	Prog	Object-Oriented Programming with Java	6	
	Math	Linear Algebra	6	
	Math	Discrete Structures	6	
	Sys	Computer Architecture	6	
2	Theo	Data Structures	6	**30**
	Theo	Algorithms	6	
	Math	Calculus	6	
	Sys	Operating Systems	6	
		Elective	6	
3	Prog	Software Engineering	6	**28**
	Theo	Theoretical Computer Science	6	
	Sys	Computer Networks	6	
	Sys	Databases	6	
		Elective	4	
Total Workload				**90**

Prep Semester

During their first semester at Kiron, students are encouraged to take the courses offered in our preparatory section in order to refresh and advance the skills necessary for studying at university level under the special circumstances of online studies. These courses include an introduction to Online Learning and Scientific Thinking, Academic English as well as study track specific refreshers e.g. in Mathematics.

Programming

CS_Pr1: Introduction to Computer Science

Module Description	
Module Name	Introduction to Computer Science
Module Content	This module gives students an introduction into the concepts of computer science. It aims to provide students with an understanding of the role computation can play in solving problems. The topics covered include: ● problem solving strategies ● primitive data types and arithmetic expressions

	- branching and loops - arrays - string manipulation - file I/O - simple event-driven programming - recursion and abstract data types
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to - explain fundamental concepts of computer science such as algorithms and data structures, - develop programs with conditionals and loops, - implement iterative and recursive algorithms.

Courses

Workload	MOOCs	Frequency /Start	Duration
30 h	Computer Science 101 (Stanford University)	self-paced	6 weeks
180 h	Introduction to Computer Science (Harvard University)	self-paced	12 weeks

Credits	8 CP
Examination	Quizzes, assignments, exams
Prerequisites	

CS_Pr2: Object-Oriented Programming with Java

Module Description	
Module Name	Object-Oriented Programming with Java
Module Content	This module introduces the paradigm of object-oriented programming, its concepts and implementation through the programming language Java. The module covers basic topics like ● control structures and data types with emphasis on structured data types and array processing ● encapsulation ● classes ● subclasses ● inheritance ● polymorphism ● class hierarchies
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● explain object-oriented programming, ● recognize object-oriented data structures and encapsulation, ● recognize class inheritance and polymorphism, ● create basic programs in Java.

Courses			
Workload	MOOCs	Frequency /Start	Duration
50 h	Introduction to Programming with Java Part 1: Starting to Code with Java (Universidad Carlos III Madrid)	Self-paced	5 weeks
60 h	Software Construction: Data Abstraction (University of British Columbia)	Self-paced	6 weeks
60 h	Software Construction: Object-Oriented Design (University of British Columbia)	Self-paced	6 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		
Prerequisites			

CS_Pr3: Software Engineering

<table>
<tr><th colspan="4">Module Description</th></tr>
<tr><td>Module Name</td><td colspan="3">Software Engineering</td></tr>
<tr><td>Module Content</td><td colspan="3">This module introduces the fundamental principles of software engineering. Both the conceptual and practical aspects of software engineering will be covered in this module. The main topics of the course include:
● software development process, life-cycle models, planning and executing software projects,
● requirements engineering,
● software design, including UML diagrams, patterns and components,
● quality assurance and testing practises,
● version control systems and integrated development environments.</td></tr>
<tr><td>Teaching Methods</td><td colspan="3">Video lectures, readings, exercises</td></tr>
<tr><td>Learning Outcomes</td><td colspan="3">After successfully completing this module, students will be able to
● identify the importance of software engineering,
● identify software development phases and processes,
● apply software engineering methods,
● organize requirements,
● design software using UML diagrams, patterns and components,
● produce and evaluate test case specifications,
● explain how to use common development tools,
● identify the reasons for software refactoring.</td></tr>
<tr><th colspan="4">Courses</th></tr>
<tr><td>Workload</td><td>MOOCs</td><td>Frequency /Start</td><td>Duration</td></tr>
<tr><td>35 h</td><td>Introduction to Programming with Java Part 2: Writing Good Code
(Universidad Carlos III Madrid)</td><td>Self-Paced</td><td>5 weeks</td></tr>
<tr><td>60 h</td><td>Software Engineering: Introduction
(University of British Columbia)</td><td>Self-Paced</td><td>6 weeks</td></tr>
<tr><td>70 h</td><td>Software Development Capstone Project
(University of British Columbia)</td><td>Self-Paced</td><td>7 weeks</td></tr>
<tr><td>Credits</td><td colspan="3">6 CP</td></tr>
<tr><td>Examination</td><td colspan="3">Quizzes, assignments, exams</td></tr>
<tr><td>Prerequisites</td><td colspan="3">CS_Pr1, CS_Pr2</td></tr>
</table>

CS_Pr4: Programming Paradigms

Module Description	
Module Name	Programming Paradigms
Module Content	This module introduces students to basic principles of programming and data modelling with various programming paradigms, with a focus on principles of functional and logical programming. Emphasis is placed on conceptual clarity and the theoretical foundations of programming. The main topics of the course include: ● strengths and weaknesses of different paradigms, such as declarative and imperative style, ● the notion of a function, recursion and termination, ● higher-order functions and currying, ● data types, polymorphism, type classes, type inference and type checking, ● pattern matching, ● lazy evaluation, strictness, ● input and output and other side effects.
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● explain paradigms of functional and logical programming, ● implement small projects in a functional programming language such as Haskell, ● implement small projects in a logical programming language such as Prolog, ● apply learned techniques in programming, ● choose the right language for a given task.

Courses			
Workload	MOOCs	Frequency /Start	Duration
20 h	Paradigms of Computer Programming – Fundamentals (Université catholique de Louvain)	irregular	5 weeks
24 h	Paradigms of Computer Programming – Abstraction and Concurrency (Université catholique de Louvain)	irregular	6 weeks
107 h	CS404: Programming Languages (Saylor Academy)	self-paced	10 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Pr1		

CS_Pr5: Web Development

Module Description			
Module Name	Web Development		
Module Content	In this module students learn about the protocols, architecture and software components used in web applications. The students will study examples of web applications and understand their functionality and architecture. The main topics of the course include: ● HTML and XML, ● stylesheets, ● scripting languages such as Javascript, ● dynamic websites using AJAX, ● security aspects of web applications.		
Teaching Methods	Video lectures, readings, exercises		
Learning Outcomes	After successfully completing this course, students will be able to ● create simple websites using markup languages together with script languages, ● use recent web technologies to meet the present standards of web development, ● use common practices to secure web applications, ● recognize the architecture of simple web applications.		
Courses			
Workload	MOOCs	Frequency /Start	Duration
125 h	CS305: Web Development (Saylor Academy)	self-paced	10 weeks
30 h	HTML, CSS, and JavaScript for Web-Developers (Johns Hopkins University)	monthly	5 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Sy2, CS_Sy4		

CS_Pr6: Object-Oriented Programming with C++

Module Description			
Module Name	Object-Oriented Programming with C++		
Module Content	This module introduces the paradigm of object-oriented programming, its concepts and implementation through the programming language C++. The module covers basic topics like ● control structures and data types with emphasis on structured data types and array processing ● encapsulation ● classes ● subclasses ● inheritance ● polymorphism ● class hierarchies		
Teaching Methods	Video lectures, readings, exercises		
Learning Outcomes	After successfully completing this course, students will be able to ● explain object-oriented programming, ● recognize object-oriented data structures and encapsulation, ● recognize class inheritance and polymorphism, ● create basic programs in C/C++.		
Courses			
Workload	MOOCs	Frequency /Start	Duration
72 h	Programming Basics (IIT Bombay)	self-paced	9 weeks
32 h	Object-Oriented Programming (IIT Bombay)	self-paced	4 weeks
Credits	4 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Pr1		

Mathematics

CS_Ma1: Linear Algebra

Module Description			
Module Name	Linear Algebra		
Module Content	Students familiarize themselves with basic mathematical notions and methods, laying the foundation for the studies of a wide range of topics in mathematics and computer science. Topics covered include: ● linear transformations of vector spaces and matrices ● methods to solve systems of linear equations such as the Gaussian elimination ● matrix calculus, including determinants and inverse matrices ● eigenspaces of linear mappings		
Teaching Methods	Video lectures, readings, exercises		
Learning Outcomes	After successfully completing this module, students will be able to ● handle abstract notions and develop easy mathematical proofs, ● summarise the basics of vector space theory and explain the concepts of vectors, linear maps and matrices, ● apply concepts of Linear Algebra in order to analyze and solve systems of linear equations, ● analyze and solve vector analysis problems that may arise in real world applications.		
Courses			
Workload	MOOCs	Frequency /Start	Duration
150 h	Linear Algebra - Foundations to Frontiers (LAFF) (University of Texas, Austin)	1-2 times a year	15 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	PREP_Ma		

CS_Ma2: Discrete Structures

Module Description	
Module Name	Discrete Structures
Module Content	This module aims to provide an overview of the basic concepts of discrete mathematics which are relevant in the field of Computer Science. The topics covered include: ● Mathematical logic ● Sets, relations, mappings, induction, recursive definitions ● Introduction to combinatorics and discrete probability theory ● Fundamentals of graph theory ● Number theory, especially modular arithmetic
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● describe fundamental mathematical entities and their properties ● use fundamental techniques for proving simple statements ● describe different discrete structures and their properties, ● explain important methods from discrete mathematics, ● apply methods from discrete mathematics to problems in computer science.

Courses			
Workload	MOOCs	Frequency /Start	Duration
30 h	What is a Proof? (University of California, San Diego)	Monthly	6 weeks
30 h	Combinatorics and Probability (University of California, San Diego)	Monthly	6 weeks
25 h	Introduction to Graph Theory (University of California, San Diego)	Monthly	5 weeks
20 h	Number Theory and Cryptography (University of California, San Diego)	Monthly	4 weeks
15 h	Delivery Problem (University of California, San Diego)	Monthly	3 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	PREP_Ma		

CS_Ma3: Calculus

Module Description	
Module Name	Calculus
Module Content	The module gives a hands-on introduction into analysis and its applications. The main focus is on the development of mathematical methods and insights. The topics include: ● Real numbers, complex numbers ● Functions, domain, range, composition, inverse functions ● Sequences and series, limit of a function, continuous functions ● Elementary rational and transcendental functions ● Differentiable functions in one variable, extreme value and intermediate value theorem and consequences ● Higher order derivatives, taylor expansion and taylor series ● Definite and indefinite integral, improper integrals, integration techniques ● Applications of differentiation and integration
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● apply elementary concepts and techniques of mathematical analysis to calculus problems ● summarize the concepts and important statements of continuity, differentiation and integration, ● differentiate and integrate real functions, ● explain the physical meaning and geometric interpretation of the derivative and integral, ● determine convergence properties of functions, sequences and series and evaluate limits, ● explain central integral theorems such as the intermediate value theorem, ● apply integrals to solve real world problems.

Courses			
Workload	MOOCs	Frequency /Start	Duration
32 h	Calculus: Single Variable Part 1 - Functions (University of Pennsylvania)	monthly	4 weeks
24 h	Calculus: Single Variable Part 2 - Differentiation (University of Pennsylvania)	monthly	3 weeks
32 h	Calculus: Single Variable Part 3 - Integration (University of Pennsylvania)	monthly	4 weeks
40 h	Calculus: Single Variable Part 4 - Applications (University of Pennsylvania)	monthly	5 weeks
40 h	Calculus: Single Variable Part 5 - Discrete Calculus (University of Pennsylvania)	monthly	5 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		

Prerequisites	PREP_Ma

CS_Ma4: Stochastics and Statistics

Module Description	
Module Name	Stochastics and Statistics
Module Content	In this module students learn standard notions and results of probability theory and statistics as required for the confident analysis of data arising from real-life contexts and its implementation with computer programs. Topics covered include: ● principles of counting, basic combinatorics ● events, probability spaces, discrete random variables, important discrete distributions ● conditional probability, Bayes formula, stochastic independence, joint distribution, conditional distribution ● expectation, variance, covariance, correlation ● continuous random variables, density function, important continuous distributions ● multidimensional distributions ● law of large numbers, central limit theorem ● descriptive statistics, histograms, collocation, measures for data ● correlation, regression ● basics of inferential statistics: parameter estimation, maximum likelihood, confidence intervals, hypothesis testing
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this module, students will be able to ● define a probability space and random experiments such as coin tossing, ● explain the concept and main properties of random variables and probability distributions, ● apply methods of descriptive statistics to summarize and represent data, ● distinguish correlation from causality and perform simple linear regressions, ● formulate and investigate statistical hypotheses, ● assess when and why a statistical technique is helpful to advance a scientific cause.

Courses			
Workload	MOOCs	Frequency /Start	Duration
192 h	Introduction to Probability - The Science of Uncertainty (Massachusetts Institute of Technology)	irregular	16 weeks
27 h	Basic Statistics (University of Amsterdam)	monthly	8 weeks
21 h	Inferential Statistics (University of Amsterdam)	monthly	7 weeks

Credits	9 CP
Examination	Quizzes, assignments, exams
Prerequisites	CS_Ma3

Theoretical Computer Science

CS_Th1: Data Structures

<table>
<tr><th colspan="4">Module Description</th></tr>
<tr><td>Module Name</td><td colspan="3">Data Structures</td></tr>
<tr><td>Module Content</td><td colspan="3">This module gives an introduction to the most commonly used data structures, their implementation in a generic programming language as well as a glimpse towards the interaction between algorithms and data structures. The topics of this module include:
● singly and doubly linked lists,
● stacks,
● queues,
● trees,
● graphs.</td></tr>
<tr><td>Teaching Methods</td><td colspan="3">Video lectures, readings, exercises</td></tr>
<tr><td>Learning Outcomes</td><td colspan="3">After successfully completing this course, students will be able to
● identify and describe given data structures,
● explain the usage of specific data structures for certain problems,
● develop and implement efficient data structures,
● adapt the introduced data structures to modified problem settings.</td></tr>
<tr><th colspan="4">Courses</th></tr>
<tr><td>Workload</td><td>MOOCs</td><td>Frequency /Start</td><td>Duration</td></tr>
<tr><td>60 h</td><td>How to Code: Simple Data
(University of British Columbia)</td><td>Self-Paced</td><td>6 weeks</td></tr>
<tr><td>60 h</td><td>How to Code: Complex Data
(University of British Columbia)</td><td>Self-Paced</td><td>6 weeks</td></tr>
<tr><td>40 h</td><td>Data Structures
(University of San Diego)</td><td>Monthly</td><td>4 weeks</td></tr>
<tr><td>Credits</td><td colspan="3">6 CP</td></tr>
<tr><td>Examination</td><td colspan="3">Quizzes, assignments, exams</td></tr>
<tr><td>Prerequisites</td><td colspan="3">CS_Pr1, CS_Pr2, CS_Ma2</td></tr>
</table>

CS_Th2: Algorithms

Module Description	
Module Name	Algorithms
Module Content	This module gives an introduction to the analysis and development of efficient algorithms. The topics of this module include: ● runtime analysis: different types of runtime approximations (best-case, worst-case, expected runtime), asymptotic analysis of upper and expected complexity bounds, big O notation (definition and computation), important complexity classes (constant, logarithmic, linear, quadratic, and exponential), methods for empirical performance evaluation, time and space trade-off ● algorithms: sorting methods (e.g. Quicksort, Mergesort or Heapsort), graph algorithms (e.g. Dijkstra's and Floyd's algorithms, Prim's and Kruskal's algorithm), principles of hashing (simple hashing functions, basic collision strategies, dynamic hashing methods, e.g. linear hashing). ● algorithmic strategies: exhaustive search, greedy algorithms, divide-and-conquer, recursive backtracking, branch-and-bound
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● evaluate the runtime and memory requirements for a given algorithm, ● describe important complexity classes for the analysis of runtime and memory complexity, ● formally model and describe algorithmic problem settings, ● adapt the introduced algorithms to modified problem settings, ● develop efficient algorithms using algorithmic strategies, ● explain and apply different sorting algorithms, ● develop and implement programs based on the introduced algorithmic techniques, ● evaluate different solution approaches for a given problem based on formal analysis.

Courses			
Workload	MOOCs	Frequency /Start	Duration
48 h	Algorithmic Toolbox (University of San Diego)	Monthly	6 weeks
24 h	Algorithms on Graphs (University of San Diego)	Monthly	6 weeks
32 h	Algorithms on Strings (University of San Diego)	Monthly	4 weeks
37 h	Advanced Algorithms and Complexity (University of San Diego)	Monthly	5 weeks
30 h	Genome Assembly Programming Challenge, Capstone (University of San Diego)	Monthly	3 weeks
Credits	6 CP		

Examination	Quizzes, assignments, exams
Prerequisites	CS_Pr1, CS_Pr2, CS_Ma3

CS_Th3: Theoretical Computer Science

Module Description	
Module Name	Theoretical Computer Science
Module Content	In this module students familiarize themselves with basic concepts of theoretical computer science and apply these to practical questions about the limitations of computer programs. Topics covered include: ● formal languages and their representation, ● finite automata: deterministic and non-deterministic, ● regular languages, regular expressions, context-free languages, ● Turing machines and computability, ● complexity theory: classes P and NP, NP completeness.
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● describe abstract notions and the connection of definitions and theorems of theoretical computer science, ● use standard jargon and explain commonly used concepts of theoretical computer science, ● define the capabilities and the limitations of different classes of automata and different classes of languages, ● explain the practical limits of computability in example applications that are deduced from a rigorous theoretical analysis.

Courses			
Workload	MOOCs	Frequency /Start	Duration
80 h	Automata Theory (Stanford University)	self-paced	8 weeks
74 h	Computability, Complexity & Algorithms (Georgia Institute of Technology)	self-paced	8 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Th1, CS_Th2, CS_Ma2		

Hardware and Systems

CS_Sy1: Computer Architecture

<table>
<tr><th colspan="4">Module Description</th></tr>
<tr><td>Module Name</td><td colspan="3">Computer Architecture</td></tr>
<tr><td>Module Content</td><td colspan="3">This module provides an overview of the components of modern von Neumann computers and their design. The topics of this module include:
● binary encoding of information,
● instructions, machine language and assembly language,
● Boolean algebra, circuit design and design of basic components,
● number representation and computer arithmetic,
● von Neumann architecture,
● memory hierarchy and caches,
● storage and I/O,
● and parallelization.</td></tr>
<tr><td>Teaching Methods</td><td colspan="3">Video lectures, readings, exercises</td></tr>
<tr><td>Learning Outcomes</td><td colspan="3">After successfully completing this course, students will be able to
● explain the fundamentals of digital circuit design (Boolean Algebra),
● apply the basics of digital encoding and digital signaling as well as the basics of combinational and sequential logic,
● design an instruction set for computer systems,
● explain the components of the von Neumann architecture,
● explain the interaction between the lower level components of a computer,
● interprete machine-level assembly language,
● translate high-level programs into sequences of computer instructions,
● explain the design of the datapath and control logic for a 32-bit computer,
● explain the role of caches in the memory hierarchy,
● discuss the concept of parallelization.</td></tr>
<tr><th colspan="4">Courses</th></tr>
<tr><td>Workload</td><td>MOOCs</td><td>Frequency /Start</td><td>Duration</td></tr>
<tr><td>60 h</td><td>Computation Structures - Part 1: Digital Circuits
(Massachusetts Institute of Technology)</td><td>irregular</td><td>10 weeks</td></tr>
<tr><td>109 h</td><td>CS301: Computer Architecture
(Saylor Academy)</td><td>self-paced</td><td>10 weeks</td></tr>
<tr><td>Credits</td><td colspan="3">6 CP</td></tr>
<tr><td>Examination</td><td colspan="3">Quizzes, assignments, exams</td></tr>
<tr><td>Prerequisites</td><td colspan="3">CS_Pr1</td></tr>
</table>

CS_Sy2: Computer Networks

Module Description	
Module Name	Computer Networks
Module Content	In this module students learn about services, protocols and the architecture of computer networks. The concepts of each layer are discussed as well as their implementing protocols. The topics of this module include: ● OSI reference model and the TCP/IP stack, ● link layer, including error detection, framing and medium access control as well as Ethernet protocols,network layer, including addressing and routing algorithms, IPv4 and IPv6, ● transport layer, including concepts such as congestion control and windows, as well as TCP and UDP, ● application level protocols like HTTP and SMTP, ● security aspects of network protocols.
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● describe design principles and architecture of a computer network, ● explain the concepts, algorithms and protocols in each layer, ● compare the OSI and TCP/IP protocol stacks, ● analyze security aspects of network protocols.

Courses			
Workload	MOOCs	Frequency /Start	Duration
60 h	Computer Networks and the Internet (FH Lübeck)	June 26, 2017	12 weeks
25 h	Fundamentals of Network Communication (University of Colorado)	monthly	5 weeks
25 h	Peer-to-Peer Protocols and Local Area Networks (University of Colorado)	monthly	5 weeks
25 h	Packet Switching Networks and Algorithms (University of Colorado)	monthly	5 weeks
25 h	TCP/IP and Advanced Topics (University of Colorado)	monthly	5 weeks

Credits	6 CP
Examination	Quizzes, assignments, exams
Prerequisites	CS_Sy1

CS_Sy3: Operating Systems

Module Description			
Module Name	Operating Systems		
Module Content	In this module students learn about the tasks and structure of operating systems as well as the design and implementation decisions in modern operating systems such as Linux. Practical examples are given in the C programming language. The topics of this module include: ● architecture of operating systems, ● processes and threads, ● synchronization, ● deadlocks, ● CPU scheduling, ● I/O, file system and disk management, ● memory allocation and access, ● basics of computer networking.		
Teaching Methods	Video lectures, readings, exercises		
Learning Outcomes	After successfully completing this course, students will be able to ● describe the role of operating systems, ● explain the rationale behind the current design and implementation decisions in modern operating systems (such as Linux), ● exemplify basic operating system abstractions, mechanisms and algorithms, ● implement simple system programs in a low-level language like C, ● apply synchronization mechanisms in software, ● evaluate various operating system aspects in a practical manner.		
Courses			
Workload	MOOCs	Frequency /Start	Duration
48 h	Introduction to Operating Systems (Georgia Institute of Technology)	self-paced	8 weeks
120 h	CS401: Operating Systems (Saylor Academy)	self-paced	10 weeks
Credits	6 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Sy1, CS_Pr2		

CS_Sy4: Databases

<table>
<tr><th colspan="4">Module Description</th></tr>
<tr><td>Module Name</td><td colspan="3">Databases</td></tr>
<tr><td>Module Content</td><td colspan="3">This course serves as an introduction to the field of database systems, covering theoretical and practical aspects of database design and database querying. The main topics of the course include:
● basic structure and conceptional design of databases via the entity/relationship model,
● theoretical aspects of relational database design, such as the relational data model, relational algebra, relational calculus and normal forms,
● query languages such as SQL,
● physical database design,
● integration of database operations in application programs.</td></tr>
<tr><td>Teaching Methods</td><td colspan="3">Video lectures, readings, exercises</td></tr>
<tr><td>Learning Outcomes</td><td colspan="3">After successfully completing this course, students will be able to
● illustrate the basics of database concepts, - models and relational algebra,
● professionally design and fit database systems to solve real life problems,
● evaluate how well a given database solution serves a particular purpose,
● set up common database systems to model given structures,
● create applications and use SQL to execute database operations.</td></tr>
<tr><th colspan="4">Courses</th></tr>
<tr><td>Workload</td><td>MOOCs</td><td>Frequency /Start</td><td>Duration</td></tr>
<tr><td>12h</td><td>Introduction to Databases: XML Data
(Stanford University)</td><td>self-paced</td><td>1 week</td></tr>
<tr><td>12h</td><td>Introduction to Databases: XPath and XQuery
(Stanford University)</td><td>self-paced</td><td>1week</td></tr>
<tr><td>12h</td><td>Introduction to Databases: XSLT
(Stanford University)</td><td>self-paced</td><td>1week</td></tr>
<tr><td>140 h</td><td>Database Systems Concepts and Design
(Georgia Institute of Technology)</td><td>May 14, 2018</td><td>14 weeks</td></tr>
<tr><td>Credits</td><td colspan="3">6 CP</td></tr>
<tr><td>Examination</td><td colspan="3">Quizzes, assignments, exams</td></tr>
<tr><td>Prerequisites</td><td colspan="3">CS_Pr2, CS_Th1</td></tr>
</table>

CS_Sy5: Electrical Engineering

Module Description	
Module Name	Electrical Engineering
Module Content	In this course students will learn about basic electrical networks and electromagnetism which includes the study of electric and magnetic fields. Topics covered include: ● Electromagnetism: ○ electrostatic field: charge, field, potential, voltage, polarisation, capacity ○ stationary electric flow field: current, Ohm’s law, resistance, power ○ stationary magnetic field: Laplace law, inductivity, permeability, magnetic circles ○ induction: inductivity, energy, induction in motion, induction in rest ● Simple networks: ○ current and voltage sources, Kirchhoff’s laws, resistance networks, nonlinear network elements ○ concepts: filters, resonance, lumped circuit models ○ elements: capacitors, inductors, resistors, MOSFET transistors, Op-Amps ○ methods: node method, superposition, VHDL
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● name and calculate electrical quantities using different methods, ● describe and use the concept of electromagnetic fields, recognize different instances and apply it in practical situations, ● design and analyze circuits using different methods, ● use VHDL to describe and design circuits and networks ● construct simple digital gates using MOSFET transistors, ● compare circuit measurements with the behavior predicted by mathematical models and explain the discrepancies.

Courses			
Workload	MOOCs	Frequency /Start	Duration
70 h	Electricity & Magnetism, Part 1 (Rice University)	self-paced	7 weeks
70 h	Electricity & Magnetism, Part 2 (Rice University)	self-paced	7 weeks
30 h	Circuits and Electronics 1: Basic Circuit Analysis (Massachusetts Institute of Technology)	self-paced	5 weeks
40 h	Circuits and Electronics 2: Amplification, Speed, and Delay (Massachusetts Institute of Technology)	self-paced	5 weeks
48 h	Digital Systems: From Logic Gates to Processors (Universitat Autònoma de Barcelona)	monthly	8 weeks

Credits	9 CP
Examination	Quizzes, assignments, exams
Prerequisites	PREP_Ph, CS_Ma1, CS_Sy1

Advanced Topics

CS_Ad1: Computer Graphics

Module Description			
Module Name	Computer Graphics		
Module Content	In this module students are introduced to the mathematical foundations of computer graphics and the implementation of those algorithms with existing hardware and software tools. Topics include: ● Representation of shapes, curves and volumes, polygons ● 2D- and 3D- transformations ● Lighting and shading ● Ray tracing, rasterization ● colouring ● APIs, such as OpenGL		
Teaching Methods	Video lectures, readings, exercises		
Learning Outcomes	After successfully completing this module, students will be able to: ● describe and implement common data structures used for representing 2D images and 3D models, ● apply standard transformations on 2D and 3D objects, ● explain how learned techniques are applied in computer graphics applications, ● create computer programs interacting with 2D images and 3D scenes, ● make use of standard computer graphic APIs.		
Courses			
Workload	MOOCs	Frequency /Start	Duration
72 h	Computer Graphics (University of California, San Diego)	self-paced	6 weeks
35 h	Interactive Computer Graphics (University of Tokyo)	monthly	7 weeks
Credits	4 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Pr2, CS_Ma1, CS_Th1		

CS_Ad2: Cryptography

Module Description	
Module Name	Cryptography
Module Content	In this module students learn about the requirements and concepts of cryptographic systems and how to correctly implement them in real-world applications. The topics covered include: ● block and stream ciphers ● secret key (symmetric) encryption techniques such as DES or AES ● public-key (asymmetric) encryption techniques such as RSA ● Diffie-Hellman ● hash functions ● digital signatures and authentication
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this course, students will be able to ● discuss different classes of cryptographic methods, ● explain the inner workings of different cryptographic methods, in particular of public-key techniques, ● apply their knowledge to safely use cryptographic methods in real life applications.

Courses			
Workload	MOOCs	Frequency /Start	Duration
49 h	Cryptography I (Stanford University)	monthly	7 weeks
42 h	Cryptography II (Stanford University)	Fall 2018	6 weeks
Credits	3 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Ma1, CS_Ma2, Prep_CS		

CS_Ad3: Artificial Intelligence

Module Description	
Module Name	Artificial Intelligence
Module Content	In this module students will approach the basic research areas of artificial intelligence and expert systems and its applications. Practical examples with help of a programming language are also given. The topics covered include: ● problem solving and search ● knowledge representation and logic ● inference techniques and resolution ● supervised and unsupervised machine learning and neural nets ● robotics ● applications, such as computer vision and natural language processing
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this module, students will be able to ● explain ideas and techniques from different fields of artificial intelligence, ● name the major applications in the field of artificial intelligence, ● discuss the limitations of current AI systems, ● apply basic techniques to solve simple problems encountered in artificial intelligence.

Courses			
Workload	MOOCs	Frequency /Start	Duration
120 h	CS405: Artificial Intelligence (Saylor Academy)	self-paced	10 weeks
Credits	5 CP		
Examination	Quizzes, assignments, exams		
Prerequisites	CS_Ma4, CS_Th1		

General Studies

CS_GS1: Business Administration

<table>
<tr><th colspan="4">Module Description</th></tr>
<tr><td>Module Name</td><td colspan="3">Business Administration</td></tr>
<tr><td>Module Content</td><td colspan="3">This module introduces students to the subject and basic concepts, theories and principles of business administration. The contents covered include:
● main concepts and functions of companies,
● principles of provident and economic organisation,
● designing organisational structures,
● operative planning,
● strategic planning
● basics of human resources management and accounting.</td></tr>
<tr><td>Teaching Methods</td><td colspan="3">Video lectures, readings, exercises</td></tr>
<tr><td>Learning Outcomes</td><td colspan="3">After successfully completing this module, students will be able to
● describe the tasks involved when overseeing a company,
● describe the underlying principles of business administration,
● connect their theoretical knowledge to real life situations and make decisions related to managing a company.</td></tr>
<tr><th colspan="4">Courses</th></tr>
<tr><td>Workload</td><td>MOOCs</td><td>Frequency /Start</td><td>Duration</td></tr>
<tr><td>100 h</td><td>Einführung in die Betriebswirtschaftlehre (RWTH Aachen)</td><td>November 6, 2017</td><td>12 weeks</td></tr>
<tr><td>Credits</td><td colspan="3">5 CP</td></tr>
<tr><td>Examination</td><td colspan="3">Quizzes, assignments, exams</td></tr>
<tr><td>Prerequisites</td><td colspan="3">PREP_Ma</td></tr>
</table>

BE_Entre: Entrepreneurship

Module Description	
Module Name	Entrepreneurship
Module Content	This module explores insights to discover and create entrepreneurial opportunities and the expertise to successfully launch, manage, and grow a business project. It focus on the development of an entrepreneurial mindset and capabilities which lead to entrepreneurial effectiveness.
Teaching Methods	Video lectures, readings, exercises
Learning Outcomes	After successfully completing this module, students will be able to ● identify a business opportunity ● develop a business plan ● organize processes and resources within a diverse organization ● propose strategies to finance a business entity ● solve business problems in the field of entrepreneurship

Courses			
Workload	MOOCs	Frequency /Start	Duration
20 h	Becoming an Entrepreneur (Massachusetts Institute of Technology)	self-paced	6 weeks
8 h	Entrepreneurship 1: Developing the Opportunity (University of Pennsylvania)	bi-weekly starting dates	4 weeks
12 h	Entrepreneurship 2: Launching your Start-Up (University of Pennsylvania)	bi-weekly starting dates	4 weeks
8 h	Entrepreneurship 3: Growth Strategies (University of Pennsylvania)	bi-weekly starting dates	4 weeks
8 h	Entrepreneurship 4: Financing and Profitability (University of Pennsylvania)	bi-weekly starting dates	4 weeks
32 h	Entrepreneurship in Emerging Economies (Harvard University)	self-paced	6 weeks
30 h	Wharton Entrepreneurship Capstone (University of Pennsylvania)	December 11, 2017	6 weeks

Credits	5 CP
Examination	Quizzes, assignments, exams
Prerequisites	PREP_Ma