UNIVERSITY OF PADERBORN

FACULTY FOR COMPUTER SCIENCE, ELECTRICAL ENGINEERING AND MATHEMATICS
DEPARTMENT OF COMPUTER SCIENCE

MODULE HANDBOOK
MASTER PROGRAM COMPUTER SCIENCE V3 (IMA V3), ENGLISH

DATE: 30. SEPTEMBER 2026

Inhaltsverzeichnis

1 Preamble and Notes **3**

2 Pflichtmodule **4**

3 Wahlpflichtmodule **18**

4 Focus Areas **149**
4.1 Algorithm Design 149
4.2 Computer Systems 149
4.3 Data Science 150
4.4 Intelligence and Data 151
4.5 Networks and Communication 152
4.6 Security 152
4.7 Software Engineering 153

5 Modules in Winter Semester **155**

6 Modules in Summer Semester **156**

7 Modules in English **157**

1 Preamble and Notes

For technical reasons, the preamble of the module handbook has been removed. It can be found under Academic Rules and Regulations on the study pages of the Institute of Computer Science. Please take note of this preamble. If you have any questions about this preamble, please contact the Study Service Computer Science.

Please also note that

1. this module handbook lists all modules provided for in the examination regulations, even if they are not offered in the corresponding semester.
2. this module handbook contains the data available at the time of writing. All information is without liability.

2 Pflichtmodule

<table>
<tr><th colspan="5">Master-Abschlussarbeit</th></tr>
<tr><td colspan="5">Master Thesis</td></tr>
<tr><td colspan="2">Module number:
null</td><td>Workload (h):
900</td><td>Credits:
30</td><td>Regular Cycle:
module.frequency.en</td></tr>
<tr><td colspan="2"></td><td>Semester number:
4. Semester</td><td>Duration (in sem.):
1</td><td>Teaching Language:
de / en</td></tr>
<tr><td>1</td><td colspan="4">Module structure:
<table>
<tr><th></th><th>Course</th><th>form of teachin</th><th>contact-time (h)</th><th>self-study (h)</th><th>status (C/CE)</th><th>group size (TN)</th></tr>
<tr><td>a)</td><td>Master Thesis – Work Plan</td><td></td><td>30</td><td>120</td><td>C</td><td>1</td></tr>
<tr><td>b)</td><td>Master Thesis</td><td></td><td>30</td><td>720</td><td>C</td><td>1</td></tr>
</table></td></tr>
<tr><td>2</td><td colspan="4">Options within the module:
none</td></tr>
<tr><td>3</td><td colspan="4">Admission requirements:
Module examinations in the major subject of at least 48 credits must have been successfully completed. At least three modules in the specialization area must be successfully completed.</td></tr>
<tr><td>4</td><td colspan="4">Contents:
Contents of the course Master-Abschlussarbeit – Arbeitsplan:
After having agreed on a topic, the student draws up a work plan. The work plan includes the targeted results, the techniques and methods used and important milestones.

Contents of the course Master-Abschlussarbeit:
The Master Thesis consists of working on a demanding subject, including a written report and an oral presentation. With the thesis the student shows her/his ability to work independently and systematically on a demanding topic which also includes developing her/his own ideas. On a state-of-the-art basis the methods of computer science should be applied systematically.
Topics for master theses are published regularly on the webpages of the research groups in the Department for Computer Science.</td></tr>
</table>

5	**Learning outcomes and competences:** Finishing their master thesis students show that they are able • to solve a problem in computer sience within an appropriate time frame using scientifically sound Methode • to apply the techniques and methods that they learned during their studies to a new and demanding problem. **Non-cognitive Skills:** • Commitment • Learning competence • Learning motivation • Motivation • Literacy (scientific) • Self-monitoring
6	**Assessments:** ☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a) - b)	Thesis	30-120 pages	100%

7	**Study Achievement:** none
8	**Prerequisites for participation in examinations:**
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted with 50 credits.
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Informatik v3
12	**Module coordinator:** Studiengangsbeauftragter Informatik

13	**Other Notes:** The master's thesis is an examination paper that concludes the scientific education and is intended to show that the candidate has the ability to work on a problem in computer science according to scientific methods within a certain period of time. The assignment should be designed to be equivalent to five months of full-time work. The thesis must be submitted five months after having been issued. As a rule, the thesis should not exceed 120 DIN A4 pages. *Remarks of course Master-Abschlussarbeit – Arbeitsplan:* **Implementation method** In agreement with supervisor. **Learning Material, Literature** Depending on the topic. *Remarks of course Master-Abschlussarbeit:* **Implementation method** Independent studies supported by individual advice and supervision **Learning Material, Literature** Depending on the thesis topic.

Projektgruppe

Project Group

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4041	600	20	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	2	2	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	Project Group	PG	240	360	C	15

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Projektgruppe:
Recommended Proficiencies
Depending on the topic.

4 **Contents:**

Contents of the course Projektgruppe:
In a project group a group of usually 8-16 students works together over a period of one year (two semesters) on a research topic determined by the group organizer. Project groups introduce students to current research topics that are usually related to the group organizer's special area of interest and the team working of the project group should be a preparation for industrial practice. Topics of project groups cover the whole range of research interests of the research groups in the Department of Computer Science.

5 **Learning outcomes and competences:**

In project groups, participating students gain first-hand practical experience in working in a team and organizing a project; in doing so, they become prepared for daily work in their later professions. The students personally experience how to carry out extensive development processes in a team. Since the tasks are divided among the individual team members, the participating students become skilled in reporting their progress and research findings to the other group members.
Non-cognitive Skills

- Commitment
- Team work
- Learning competence
- Learning motivation
- Motivation
- Literacy (scientific)
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Partial Module Exam		100%

In the Project Group module, the successful completion of projects must be demonstrated by submitting software and documentation as a phase-related examination. A grade is awarded for the entirety of the projects worked on.
The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Practical work		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted as 8 credits.

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Studiengangsbeauftragter Informatik

13	**Other Notes:** *Remarks of course Projektgruppe:* **Implementation method** • The number of participants is limited to 16 people. • Developing knowledge on the selected systematic approaches, methods and tools relevant to the research topic- usually done in an introductory seminar phase. • Logical assigning "jobs" (assigning responsibilities to the individual group members). • Discovering and promoting the participants' special individual talents, which are either already apparent or which can be developed throughout the project - such as through seminar presentations or appropriate job assignments. • Setting up a process-oriented personnel structure, similar to the structure of an industrial design team; delegating subtasks to smaller subgroups who report their findings. • Regular progress reports made by individuals and subgroups. • Writing a highly distributed interim report and final report. **Learning Material, Literature** Depending on the topic.

Seminar I

Seminar I

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4045	150	5	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	2	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	2024.7092b Seminar	S2	30	120	C	15

2 **Options within the module:**

Seminars from the Master Program Computer Science.

3 **Admission requirements:**

Prerequisites of course Seminar:
Recommended Proficiencies
Depending on the seminar topic.

4 **Contents:**

Contents of the course Seminar:
In seminars, students work independently on an individual research topic by using background literature from various sources. They describe their research topic in a presentation followed by discussion and a written report. The presentation material and the written report serve two different purposes: Whereas the presentation material supports the lectures (held within a specific time period), the written report provides students the opportunity to acquire detailed information on the reported topic at a later date.
Seminars usually consist of 8 to 15 related subtopics, each of which is researched by one participating students. Seminar topics cover the whole spectrum of research topics of the research groups in the Department of Computer Science.

5 **Learning outcomes and competences:**

In seminars, students learn the techniques for independent research work on non-trivial topics and how to present these topics in a presentation and in written form. The seminar participants are encouraged to familiarize themselves with a research-oriented subfield of computer science. They learn how to plan a presentation and hold it within the determined time frame (usually 45 to 60 minutes), and to prioritize the contents of the presentation. The participants experience how an audience obtains knowledge from a presentation, and how to exchange opinions and information in discussions. Seminars also teach rhetorical skills for presentations and discussions. Participating students learn how to structure a presentation according to its contents and how to use various means to illustrate complex issues. They also learn how to handle the background literature appropriately.

Non-cognitive Skills

- Commitment
- Cooperation
- Learning competence
- Media competence
- Motivation
- Literacy (scientific)
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Presentation and seminar paper	45-60 minutes, 15-30 pages	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	none		

8 **Prerequisites for participation in examinations:**

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11	**Reuse in degree courses or degree course versions :** Masterstudiengang Informatik v3
12	**Module coordinator:** Studiengangsbeauftragter Informatik
13	**Other Notes:** *Remarks of course Seminar:* **Implementation method** Seminar paper and presentation **Learning Material, Literature** Depending on the seminar topic.

Seminar II

Seminar II

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4046	150	5	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	2024.7092b Seminar	S2	30	120	C	15

2 **Options within the module:**

Seminars from the Master Program Computer Science.

3 **Admission requirements:**

Prerequisites of course Seminar:
Recommended Proficiencies
Depending on the seminar topic.

4 **Contents:**

Contents of the course Seminar:
In seminars, students work independently on an individual research topic by using background literature from various sources. They describe their research topic in a presentation followed by discussion and a written report. The presentation material and the written report serve two different purposes: Whereas the presentation material supports the lectures (held within a specific time period), the written report provides students the opportunity to acquire detailed information on the reported topic at a later date.
Seminars usually consist of 8 to 15 related subtopics, each of which is researched by one participating students. Seminar topics cover the whole spectrum of research topics of the research groups in the Department of Computer Science.

5 **Learning outcomes and competences:**

In seminars, students learn the techniques for independent research work on non-trivial topics and how to present these topics in a presentation and in written form. The seminar participants are encouraged to familiarize themselves with a research-oriented subfield of computer science. They learn how to plan a presentation and hold it within the determined time frame (usually 45 to 60 minutes), and to prioritize the contents of the presentation. The participants experience how an audience obtains knowledge from a presentation, and how to exchange opinions and information in discussions. Seminars also teach rhetorical skills for presentations and discussions. Participating students learn how to structure a presentation according to its contents and how to use various means to illustrate complex issues. They also learn how to handle the background literature appropriately.

Non-cognitive Skills

- Commitment
- Cooperation
- Learning competence
- Media competence
- Motivation
- Literacy (scientific)
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Presentation and seminar paper	45-60 minutes, 15-30 pages	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	none		

8 **Prerequisites for participation in examinations:**

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11	**Reuse in degree courses or degree course versions :** Masterstudiengang Informatik v3
12	**Module coordinator:** Studiengangsbeauftragter Informatik
13	**Other Notes:** *Remarks of course Seminar:* **Implementation method** Seminar paper and presentation **Learning Material, Literature** Depending on the seminar topic.

Studium Generale – Master

General Studies – Master

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
null	360	12	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	4	de / en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	Studium Generale – Master	L6 Ex3	135	225	C	30

2 **Options within the module:**

Any courses outside of computer science may be chosen.

3 **Admission requirements:**

Prerequisites of course Studium Generale – Master:
Recommended Proficiencies
Depending on the courses chosen.

4 **Contents:**

Contents of the course Studium Generale – Master:
Depending on the courses chosen.

5 **Learning outcomes and competences:**

Students expand their scientific horizons beyond the boundaries of computer science and their chosen minor. Depending on the chosen course, they have acquired competencies in communication skills, teamwork and presentation techniques.
Non-cognitive Skills

- Commitment
- Cooperation
- Media competence
- Literacy (scientific)
- Self-monitoring

<table>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)
<table><tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr><tr><td>a)</td><td>Exam in general studies</td><td></td><td>100%</td></tr></table>The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.</td></tr>
<tr><td>7</td><td>Study Achievement:
<table><tr><th>zu</th><th>Type of achievement</th><th>Duration or Scope</th><th>SL / QT</th></tr><tr><td>a)</td><td>Qualified participation in general studies</td><td></td><td>QP</td></tr></table>Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the qualified participation will be conducted.</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted as 4 credits.</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Informatik v3</td></tr>
<tr><td>12</td><td>Module coordinator:
Studiengangsbeauftragter Informatik</td></tr>
<tr><td>13</td><td>Other Notes:
If no minor subject is selected, any combination of courses outside of computer science and in the scope of 12 LP must be selected. The given distribution of the LP to courses is only exemplary.</td></tr>
</table>

3 Wahlpflichtmodule

<table>
<tr><th colspan="5">Advanced Algorithms</th></tr>
<tr><td colspan="5">Advanced Algorithms</td></tr>
<tr><td colspan="2">Module number:
M.079.4002</td><td>Workload (h):
180</td><td>Credits:
6</td><td>Regular Cycle:
module.frequency.en</td></tr>
<tr><td colspan="2"></td><td>Semester number:
null</td><td>Duration (in sem.):
1</td><td>Teaching Language:
en</td></tr>
<tr><td>1</td><td colspan="4">Module structure:
<table>
<tr><th></th><th>Course</th><th>form of teachin</th><th>contact-time (h)</th><th>self-study (h)</th><th>status (C/CE)</th><th>group size (TN)</th></tr>
<tr><td>a)</td><td>L.079.05701
Advanced Algorithms</td><td>L3
Ex2</td><td>75</td><td>105</td><td>CE</td><td>50/25</td></tr>
</table></td></tr>
<tr><td>2</td><td colspan="4">Options within the module:
none</td></tr>
<tr><td>3</td><td colspan="4">Admission requirements:
Prerequisites of course Advanced Algorithms:
Recommended Proficiencies
Willingness and ability to learn the creative process of algorithm design and efficiency analysis using mathematical methods. Basic Knowledge of some basic algorithms and data structures and their analyses is assumed.</td></tr>
<tr><td>4</td><td colspan="4">Contents:
Contents of the course Advanced Algorithms:
This course presents advanced algorithms and algorithmic paradigms for basic problems. In particular, methods such as randomization and derandomization, as well as the concepts of approximation and online algorithms, are presented using important algorithmic problems. In all cases, proof of correctness and run-time analyzes are carried out.
• Randomized algorithms, derandomization, examples: Randomized Rounding and others
• Online algorithms, examples: scheduling problems and others
• approximation algorithms, examples: NP-hard problems</td></tr>
</table>

5 **Learning outcomes and competences:**

Students apply advanced algorithmic design methods such as randomization, approximation, and online algorithms to new problems and analyze them using combinatorial and probabilistic methods.

Non-cognitive Skills

- Learning competence
- Learning motivation

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Christian Scheideler

13	**Other Notes:** *Remarks of course Advanced Algorithms:* **Implementation method** • Lecture with beamer and blackboard. • Exercises in small groups. • expected student activities: active participation in exercises, homework. • Exercise sheets, solutions are presented and discussed in tutorials. • In exercises and homework, design and analysis of algorithms are practiced on selected examples. **Learning Material, Literature** Standard textbooks, slides of the lecture, exercise sheets

Advanced Complexity Theory (v3)

Advanced Complexity Theory (v3)

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4004	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05803 Advanced Complexity Theory	L3 Ex2	75	105	CE	25

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Advanced Complexity Theory:
Recommended Proficiencies
Basic knowledge about complexity theory (e.g., Turing machines, NP-completeness)

4 **Contents:**

Contents of the course Advanced Complexity Theory:
Complexity Theory deals with determining the amount of resources (e.g. runtime, memory consumption) necessary and sufficient for solving a given algorithmic problem (e.g. Travelling Salesperson Problem (TSP)) on a given machine model (e.g. Turing machine). One approach is to define complexity classes like P, NP, PSPACE, in order to classify problem complexity by means of completeness in such classes, like the famous class of NP-complete problems. This gives conditional results like “If NP is not equal P, then TSP is not solvable in polynomial time.” This branch of Complexity Theory is often referred to as Structural Complexity Theory. In contrast, proving explicit lower bounds for given problems is the topic of the so-called Concrete Complexity Theory. As nobody is currently able to prove superlinear time bounds for explicitly defined problems on general computation models like Turing machines, one considers somewhat restricted models like 1-tape Turing machines, monotone Boolean circuits, Boolean circuits with bounded depth, algebraic computation models, and several kinds of parallel computation models. This lecture surveys approaches to prove such lower bound on various such models.

- Deterministic, non-deterministic and probabilistic time and space complexity classes, hierarchies, completeness
- Lower bounds for size and depth of different variants of Boolean circuits
- Lower bounds for algebraic computations

5 **Learning outcomes and competences:**

The students get to know fundamental techniques in the area of complexity theory. They can decide in which complexity class the storage space and the runtime requirements of algorithmic problems can be classified. They can classify new problems into complexity classes.
Non-cognitive Skills

- Attitude
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Johannes Blömer

13	**Other Notes:** *Remarks of course Advanced Complexity Theory:* **Implementation method** • Lecture with beamer and blackboard • Practice in small groups • Expected activities of the students:contributions to presence exercises, homework **Learning Material, Literature** • C.H. Papadimitiriou, Computational Complexity, Addison-Wesley • S. Arora, B. Barak, Computational Complexity - A Modern Approach, Cambridge University Press • Slides of the lecture, exercise sheets

Advanced Computer Architecture

Advanced Computer Architecture

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4005	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05724 Advanced Computer Architecture	L3 Ex2	75	105	CE	50/25

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Advanced Computer Architecture:
Recommended Proficiencies
Basic knowledge in computer architecture.

4 **Contents:**

Contents of the course Advanced Computer Architecture:
The course teaches concepts and methods used in modern processor architecture to exploit the available parallelism at the levels of instructions, data and threads.

- Fundamentals of computer architectures (refresher)
- Memory hierarchy design
- Instruction-level parallelism
- Data-level parallelism: Vector, SIMD and GPU architectures
- Thread-level parallelism
- Warehouse-scale computer

5 **Learning outcomes and competences:**

After attending the course, the students

- are able to explain principles of modern memory hierarchies,
- to analyze different levels of parallelism,
- to assess the suitability of different architectural concepts and thus
- to evaluate modern developments in computer architecture.

Non-cognitive Skills

- Team work
- Learning competence

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Christian Plessl, Prof. Dr. Marco Platzner

13	**Other Notes:** *Remarks of course Advanced Computer Architecture:* **Implementation method** • Lecture with projector and board • Interactive exercises in the lecture room item Computer-based exercises with simulation tools • Analysis of case studies **Learning Material, Literature** • Lecture slides and exercise sheets • Exercise sheets and technical documentation for the for the computer-based exercises • Hennessey, Patterson: Computer Architecture: A Quantitative Approach (5th edition or newer), Morgan Kauf- mann, 2012. • Information about alternative and additional literature as well as teaching material on the course's website and in the lecture slides

Advanced Distributed Algorithms and Data Structures

Advanced Distributed Algorithms and Data Structures

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4006	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05700 Advanced Distributed Algorithms and Data Structures	L3 Ex2	75	105	C	30

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Advanced Distributed Algorithms and Data Structures:
Recommended Proficiencies
Algorithms and data structures, distributed algorithms and data structures

4 **Contents:**

Contents of the course Advanced Distributed Algorithms and Data Structures:
The lecture will cover advanced topics in distributed algorithms and data structures. Topics covered in the course are access control, synchronization, consensus, information dissemination, hybrid networks, scheduling, and optimization. In addition to presenting solutions to these topics, also concrete applications will be presented.

5 **Learning outcomes and competences:**

Students get to know advanced methods and algorithms for currently very relevant distributed systems. They are able to adapt algorithms to new situations and to determine their complexity. They can implement basic distributed algorithms.
Non-cognitive Skills

- Team work
- Learning competence
- Literacy (scientific)
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Christian Scheideler

13 **Other Notes:**

Remarks of course Advanced Distributed Algorithms and Data Structures:
Implementation method
Lecture with tutorials and software project
Learning Material, Literature
Lectures notes

Advanced Networked Systems			
Advanced Networked Systems			
Module number: M.079.4096	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** module.frequency.en
	Semester number: 1-3	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.079.05820 Advanced Networked Systems	L2 Ex3	75	105	CE	50/25

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Advanced Networked Systems:
Recommended Proficiencies
Knowledge of computer networks, operating systems, programming languages, C/C++ and Python programming in the Linux environment, and a keen interest in understanding how things work under the hood. Ability to read scientific papers professionally. Ability to code in a complex setting.

4 **Contents:**

Contents of the course Advanced Networked Systems:
The course will cover concepts and designs for modern networked systems adopted by the Internet and cloud data centers to meet the ever-increasing demands of data transfer and computation driven by big data and machine learning applications.

- Networking fundamentals (refresher)
- Data center networks (architectures, congestion control)
- Software-defined networks (SDN, OpenFlow)
- Programmable networks (P4, eBPF/XDP)
- Programmable network device architectures (RMT, SmartNICs)
- Network monitoring and in-network computing (caching, aggregation)
- Systems for machine learning (model training, systems for LLMs)

Much of the course will be based on discussions of cutting-edge research topics, complemented with hands-on programming assignments.

5 **Learning outcomes and competences:**

Upon completion of this course, students will be able to learn the following.

- Gain knowledge of current research topics in networked systems.
- Understand the design of these new networked systems technologies and reason about the design choices therein.
- Build complex networked systems by applying some of these designs, analyze and evaluate the merits and limitations of these designs, and explain the design choices for the built systems.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Lin Wang

13	**Other Notes:** *Remarks of course Advanced Networked Systems:* **Implementation Method** The course content will be taught with slides-based lectures, interactive exercises, and programming-based project assignments. **Learning Material, Literature** • Lecture slides and exercise sheets • Project description documents for the programming assignments • Additional literature (e.g., research papers) on the course website and in the lecture slides

Agentic Software Engineering

Agentic Software Engineering

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4212	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05705 Agentic Software Engineering		75	105	C	80

2 **Options within the module:**

none

3 **Admission requirements:**

none

4	**Contents:** *Contents of the course Agentic Software Engineering:* Since the emergence of generative AI (GenAI), and particularly since the use of GenAI within agents, software engineering (SE) has been undergoing a radical transformation that raises many questions: In which phases of SE can GenAI be deployed effectively? What are the opportunities, but also the limitations and pitfalls involved? How does GenAI change the competency profile that will be expected of software engineers in the future? In this course, the use of GenAI and agents across the various phases of the software lifecycle (requirements analysis, implementation, testing, etc.) is discussed, made tangible through hands-on experience, and — where appropriate — critically examined. We will also cover current research on AI-driven software development methods. In practical exercises, students are introduced to current technologies that enable or simplify the design and deployment of agents in software engineering. Topics covered: • Fundamentals of generative AI (concise recap as a starting point) • Retrieval-Augmented Generation and agents: how each works and how they differ • Economic perspective: productivity effects (empirical studies), cost, latency, and model economics in production agent deployment • Knowledge and context engineering for SE tasks: providing repositories, issues, tests, and documentation as context (information curation, context windows, repo maps, RAG over codebases) • Tool use and function calling: interfaces between agents and development tools (compiler, test runner, linter, debugger, version control), and standards such as the Model Context Protocol (MCP), Agent Client Protocol (ACP) • Common types of agents (ReAct, Plan-and-Execute, Reflexion, etc.) • Multi-agent systems (incl. Agent2Agent protocol) • Benefits and challenges of agents across the various phases of the software lifecycle (requirements analysis, implementation, testing, etc.) • Feedback through testing and symbolic methods (e.g., static analysis and verification) as a corrective and safety net • Reliability and limitations: hallucinations, the out-of-distribution problem with proprietary codebases, correctness guarantees • Security of agentic systems: prompt injection, autonomous execution, supply-chain risks from generated code • Human-in-the-loop, accountability, and code ownership: the changing role and competency profiles of software engineers

5 **Learning outcomes and competences:**

After completing this module, students are able to

- explain how generative AI, retrieval-augmented generation, and agentic systems work and distinguish their respective areas of application in software engineering,
- apply current agent technologies to tasks across the software lifecycle and provide them with suitable context through repositories, tests, and tool interfaces,
- analyse the benefits and challenges of using GenAI and agents in the individual phases of software development,
- assess the reliability, security, and limitations of agentic systems, including hallucinations, the out-of-distribution problem, and correctness guarantees,
- design their own agent-based solution for a software engineering task and justify the architectural choices involved,
- evaluate the economic and organisational implications of deploying agents productively, including effects on cost, productivity, and the role of software engineers.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4

12 **Module coordinator:**

Prof. Dr. Eric Bodden

13	**Other Notes:** *Remarks of course Agentic Software Engineering:* **Implementation Method** The content is presented in lectures and deepened in accompanying exercises. In hands-on exercises, students apply current agent technologies themselves and explore the design and deployment of agents on concrete software engineering tasks. Recent research findings are discussed and critically assessed on the basis of primary literature. **Learning Material, Literature** • Lecture slides and exercise sheets • Foundational literature on the agent architectures covered • Current research papers and conference contributions on AI-assisted software engineering, announced during the course • Specifications and documentation of relevant protocols (Model Context Protocol, Agent Client Protocol) • Further reading and online resources provided during the course.

Algorithms for Complex Virtual Scenes

Algorithms for Complex Virtual Scenes

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4009	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05803 Algorithms for Complex Virtual Scenes	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Algorithms for Complex Virtual Scenes:
Recommended Proficiencies
Willingness and ability to learn the creative process of algorithm design and efficiency analysis using mathematical methods. Basic Knowledge of some basic algorithms and data structures and their analyses is assumed.

4 **Contents:**

Contents of the course Algorithms for Complex Virtual Scenes:
Walkthrough systems allow viewing and walking through a virtual 3D scene and finds application in architecture programs, simulations or games. The efficiency of real-time rendering algorithms is crucial for a smooth and fast rendering of the virtual 3D scene in a walkthrough system. There are different algorithmic approaches to reduce highly complex 3D geometric data and to achieve a rendering of the scene in real time. The lecture introduces different algorithmic approaches, e.g., visibility culling, simplification, level of detail, image-based rendering.
The course includes the following contents:

- Walkthrough problem
- Spatial Data structures: kd-tree, BSP-tree, octree, loose octree
- Level of detail: adaptive LOD management, mesh simplification, progressive meshes
- Visibility culling: view frustum culling, potentially visible sets (PVS), dynamic analysis of PVS, hierarchical z-buffer, hierarchical occlusion maps, coherent hierarchical culling, aspect graph, visibility space partition
- Replacement: color cubes, randomized z-buffer, hierarchical image caching
- Parallel rendering: classification and modeling, parallel rendering as a sorting problem, hybrid sort-first/sort-last rendering

5 **Learning outcomes and competences:**

Students will be able to

- name, explain and apply basic algorithms and data structures for problems in the rendering of complex virtual scenes.
- identify basic algorithmic problems in application problems of the rendering of complex virtual scenes and select suitable algorithms and data structures for them
- analyze, compare and investigate runtime and memory estimation of spatial data structures and algorithms
- evaluate what impact the choice of spatial data structures has on the efficiency of algorithms for rendering complex virtual scenes
- develop own efficient visibility algorithms based on spatial data structures for other virtual scenes with special characteristics
- develop own efficient approximation algorithms based on spatial data structures for other virtual scenes with special characteristics
- discuss problems of the rendering of complex virtual scenes and corresponding solution proposals with experts in the field

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	120-180 min or 30-45 min or 30 min	100%

<table>
<tr><td>7</td><td>Study Achievement:
<table>
<tr><th>zu</th><th>Type of achievement</th><th>Duration or Scope</th><th>SL / QT</th></tr>
<tr><td>a)</td><td>Assignments, course paper or progress reports</td><td></td><td>CA</td></tr>
</table></td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
Passing of course achievement</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Informatik v3, Masterstudiengang Informatik v4</td></tr>
<tr><td>12</td><td>Module coordinator:
Dr. Matthias Fischer</td></tr>
</table>

13	**Other Notes:** *Remarks of course Algorithms for Complex Virtual Scenes:* **Implementation Method** The contents are taught by means of a presentation within the framework of a lecture. The lecture is usually held with beamer and blackboard. In exercises and assignments, design and analysis of algorithms are presented on selected examples and are deepened and developed by the students themselves in tutorials in small groups as well as in self study and supplemented by practical exercises. Sample solutions of exercise sheets are presented in central exercises. The expected activities of the students are participation in presence exercises and independent work on assignments. **Learning Material** • Lecture slides, exercise sheets, sample solutions if applicable, lecture recordings from previous years, blackboard transcription **Literature** • Real-Time Rendering; Tomas Akenine-Möller, Eric Haines; AK Peters, 2002. • Level of Detail for 3D Graphics; David Luebke, Martin Reddy, Jonathan D. Cohen; Morgan Kaufmann Publishers, 2002. • Algorithmen in der Computergraphik; Thomas Rauber; Teubner, 1993. • Wavelets for Computer Graphics: Theory and Applications; Eric Stollnitz, David H. Salesin, Anthony D. DeRose; Morgan Kaufmann Publishers, 1996. • Graphic Gems; Andrew S. Glassner; Academic Press; 1990. • Game Programming Gems; Mark DeLoura; Charles River Media; 2000. Computational Geometry • Computational Geometry - Algorithms and Applications; Mark de Berg, Marc de Kreveld, Mark Overmars; Springer Verlag, 2000. • Computational Geometry in C; Joseph O'Rourke; Cambridge University Press, 1998. • Algorithmic Geometry; Jean-Daniel Boissonnat, Herve Bronniman;Cambridge University Press, 1998. • Algorithmische Geometrie Grundlagen, Methoden, Anwendungen; Rolf Klein; Springer Verlag, 2005. General principles of computer graphics • 3D Computer Graphics; Alan Watt; Addison Wesley, 1999. • Computer Graphics, Principles and Practice; James Foley, Andries van Dam, Steven Feiner, John Hughes; Addison Wesley, 1995. • Computer Graphics; Donald Hearn, M. P. Baker; Prentice Hall, 2003. Additional literature will be announced in the course.

Approximate Computing

Approximate Computing

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4068	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.09758 Approximate Computing	L3 Ex2	75	105	C	100/30

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Approximate Computing:
Recommended Proficiencies
Knowledge of the Bachelor-level courses Digital Design and Computer Architecture are beneficial.

4 **Contents:**

Contents of the course Approximate Computing:
Approximate Computing is an emerging paradigm that trades-off computational accuracy for a significant reduction in energy, execution time, or chip area. This research-oriented course introduces to the field of Approximate Computing and its most remarkable aspects, and explains the main methods used to implement efficient computing systems by reducing accuracy. The course discusses approximations at all levels of a computing system, from applications down to hardware technologies. In exercise/tutorial sessions the efficiency of these techniques in various domains are examined, including deep learning and digital signal processing.

- Introduction and motivation for approximate computing
- Approximation at the application level, e.g., in machine learning and digital signal processing
- Programming languages/compilers for approximate computing
- Approximate microarchitectures
- Automated synthesis of approximate circuits
- Inexact arithmetic components and performance optimization via accuracy trade-offs
- Approximation techniques at the technology level
- Exercises/tutorial: Approximating deep learning and digital signal processing algorithms

5 **Learning outcomes and competences:**

After attending this course, the students are able

- to name and explain approximation techniques at all levels of a computing system,
- to identify major engineering/research problems when building approximate computing systems,
- to judge the suitability of approximation techniques for different application domains, and
- to apply approximation techniques to realize efficient hardware accelerators, in particular for deep learning and digital signal processing

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4, Masterstudiengang Robotics and Artificial Intelligence Engineering v1 (RAIMA v1)

12 **Module coordinator:**

Prof. Dr. Marco Platzner

13	**Other Notes:** *Remarks of course Approximate Computing:* **Implementation method** • Lecture with projector and black/white board • Interactive exercises/discussions in the lecture room • Computer-based tutorials **Learning Material, Literature** • Lecture slides, exercise sheets, and tutorial assignments • Adrian Sampson, Luis Ceze, and Dan Grossman: Good-Enough Computing. IEEE Spectrum, 50(10):54-59, 2013 • Ravi Nair. Big Data Needs Approximate Computing: Technical Perspective. Communications of the ACM, 58(1): 104, 2015. • Sparsh Mittal. A Survey of Techniques for Approximate Computing. ACM Computing Surveys, 48(4), 2016. • Qiang Xu, Todd Mytkowitz, and Nam Sung Kim. Approximate Computing: A Survey. IEEE Design & Test, 33(1):8-22, 2016. • Weiqiang Liu and Fabrizio Lombardi (Editors), Approximate Computing. Springer, 2022. • Additional resources and links to current research papers are provided in the lecture.

Data-Driven Innovation

Data-Driven Innovation

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4076	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05722 Data-driven Innovation	L2 Ex3	75	105	CE	60

2 **Options within the module:**

none

3 **Admission requirements:**

keine

4 **Contents:**

Contents of the course Data-driven Innovation:
Digitalization is changing the market services of tomorrow and the way in which they are developed. Traditional methods of strategic planning and system engineering leave potentials untapped, while data-driven solutions capture these potentials. The lecture provides an overview of the challenges and solutions of data-driven innovation. Theoretical principles and concepts are introduced and exemplary applications from practice are presented. The process is considered from data collection to possibilities for data evaluation and the development of innovative market services. The knowledge acquired is deepened and put into practice in the exercises.

5 **Learning outcomes and competences:**

Students gain a comprehensive understanding of data-driven solutions in the area of Data-Driven Innovation. Furthermore, students learn how to independently develop data-driven solutions.
Non-cognitive Skills

- Commitment
- Empathy
- Social and ethical judgement
- Team work
- Cooperation
- Motivation
- Self-monitoring

<table>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)
<table>
<tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr>
<tr><td>a)</td><td>Written or oral examination</td><td>90-120 min or 40 min</td><td>100%</td></tr>
</table>
The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.</td></tr>
<tr><td>7</td><td>Study Achievement:
none</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Informatik v3</td></tr>
<tr><td>12</td><td>Module coordinator:
Dr. Christian Koldewey, Prof. Dr.-Ing. Roman Dumitrescu</td></tr>
<tr><td>13</td><td>Other Notes:
Remarks of course Data-driven Innovation:
Implementation method
Lecture and exercise
Learning Material, Literature
Gausemeier, J.; Dumitrescu, R.; Echterfeld, J.; Pfänder, T.; Steffen, D.; Thielemann, F.: Innovationen für die Märkte von morgen – Strategische Planung von Produkten, Dienstleistungen und Geschäftsmodellen. Carl Hanser Verlag, München, 2019</td></tr>
</table>

Data Science for Software Engineering

Data Science for Software Engineering

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4101	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05817 Data Science for Software Engineering	L2 Ex3	75	105	C	30

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Data Science for Software Engineering:
Recommended Proficiencies
Good programming skills using Java and/or Python is helpful to make the assignments. Basic background on machine learning is helpful to understand some of the Data Science concepts.

4 **Contents:**

Contents of the course Data Science for Software Engineering:
Software engineers deal with software repositories in their daily work, such as when they develop source code in version control systems, or post issues in issue trackers, or communicate through emails in mailing lists, or discuss in forums and blogs. The big amount of data in software repositories, their continuous evolution, complexity and heterogeneity present a challenge for software engineers. In the past years, researchers proposed approaches that use techniques from the data science to support software engineers. This course will explain the application of data science techniques on software repositories to achieve common software engineering tasks. The course includes the following topics:

- Types and structure of software repositories.
- Clustering of source code.
- Natural language processing pipeline.
- Topic modeling.
- Word embedding.
- Information retrieval.
- Supervised machine learning.
- Statistical analysis.

Concepts are discussed in the lectures and applied using a set of group assignments to analyze opensource systems, ans achieve certain software architecture and maintenance tasks.

5 **Learning outcomes and competences:**

Students will be able to

- Clarify and discuss types and structure of software repositories.
- Clarify and discuss main concepts of data science techniques, and their application on sotware repositories.
- Apply data science techniques on large-scale software repositories.
- Derive useful implications from the analysis results.
- Summarize and report analysis results in a scientific format.
- Work in teams.
- Write scientific reports
- Present research results

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 30-45 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments and short presentations		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4

12 **Module coordinator:**

Dr. Mohamed Aboubakr Mohamed Soliman

13	**Other Notes:** *Remarks of course Data Science for Software Engineering:* **Implementation Method** The course focus on the application of data science methods in software engineering more than the mathematical background of data science methods. The main concepts of methods are conveyed through a presentation as part of a lecture and the application of methods is further investigated through group assignments and presentations. **Learning Material, Literature** Beside the slides, further learning materials from prominent publications in the software engineering literature will be provided for each topic.

Designing code analyses for large-scale software systems 1

Designing code analyses for large-scale software systems 1

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4070	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	Designing code analyses for large-scale software systems 1	L3 Ex2	75	105	C	30

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Designing code analyses for large-scale software systems 1:
Recommended Proficiencies
A mature understanding of the Java programming languages and object-oriented programming will be helpful.

4	**Contents:** *Contents of the course Designing code analyses for large-scale software systems 1:* Static code analysis has the goal of finding programming mistakes automatically, by searching for suspicious anti-patterns in a program's code. This course will explain how to design static code analysis that are inter-procedural, i.e., consider the whole program, across procedure boundaries. Designing such analyses is challenging, as they need to handle millions of program statements efficiently and precisely. Example applications are drawn from the area of IT security. This course is part of a combination DECA 1/2. In DECA 2 we will be covering current approaches directly our of research. We strongly recommend attending DECA 1 before DECA 2. **Topics covered include:** • Type systems and flow-insensitive, contraint-based analysis • Lattices and fixed points • Intra-procedural flow-sensitive static code analysis • Interval analysis, widening and narrowing • Call-graph construction • Pointer Analysis • Inter-procedural program analysis • Call-strings approach to context-sensitive analysis • Functional approach to context-sensitive analysis • Value-based termination, VASCO • Distributive analyses using IFDS • Sensible arrangements of Flow Functions • Distributive analyses using IDE Throughout, we will discuss applications to software security.
5	**Learning outcomes and competences:** After having attended this course, students will have learned. . . • how to make educated design decisions when designing automated code analysis for large-scale software systems, • which algorithms have which properties when using them to implement static code-analyses, • how to design real–world code analyses for practical problem cases from the area of IT security • how to interpret important terminology such as context, flow, field and object sensitivity • how to evaluate and explain the important limitations of static code analysis • which typical security code analyses exist (OWASP Top 10 etc.) and how they relate to the analysis frameworks explained in the course. **Non-cognitive Skills** • Learning competence • Learning motivation

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Eric Bodden

13	**Other Notes:** *Remarks of course Designing code analyses for large-scale software systems 1:* **Implementation method** Lectures and group exercises as well as practical programming labs using worldwide leading frameworks for static code analysis **Learning Material, Literature** • Thomas Reps, Susan Horwitz, and Mooly Sagiv. 1995. Precise interprocedural dataflow analysis via graph reachability. POPL '95 • Shmuel Sagiv, Thomas W. Reps, and Susan Horwitz. 1995. Precise Interprocedural Dataflow Analysis with Applications to Constant Propagation. TAPSOFT '95 • Akash Lal, Thomas Reps, and Gogul Balakrishnan. 2005. Extended weighted pushdown systems. CAV 2005 • Nomair A. Naeem, Ondrej Lhoták, and Jonathan Rodriguez. 2010. Practical extensions to the IFDS algorithm. CC 2010 • Yannis Smaragdakis, Martin Bravenboer, and Ondrej Lhoták. 2011. Pick your contexts well: understanding object-sensitivity. POPL 2011 • Eric Bodden. 2012. Inter-procedural data-flow analysis with IFDS/IDE and Soot. SOAP 2012 • Rohan Padhye, Uday P. Khedker. Interprocedural Data Flow Analysis in Soot using Value Contexts. SOAP 2013

Designing code analyses for large-scale software systems 2

Designing code analyses for large-scale software systems 2

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4071	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05821 Designing code analyses for large-scale software systems 2	L3 Ex2	75	105	C	30

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Designing code analyses for large-scale software systems 2:
Recommended Proficiencies
We strongly recommend that attendees have completed DECA 1 beforehand. A mature understanding of the Java and/or C++ programming languages and object-oriented programming will be helpful.

4	**Contents:** *Contents of the course Designing code analyses for large-scale software systems 2:* Static code analysis has the goal of finding programming mistakes automatically, by searching for suspicious anti-patterns in a program's code. This course will explain how to design static code analysis that are inter-procedural, i.e., consider the whole program, across procedure boundaries. Designing such analyses is challenging, as they need to handle millions of program statements efficiently and precisely. Example applications are drawn from the area of IT security. This course builds on the DECA 1 course. In DECA 2, we discuss novel concepts directly from research, for example so-called demand-driven analyses, which are characterized by a more precise and at the same time more efficient analysis, but also pushdown systems, which provide a allow elegant modeling and at the same time fast execution of program analyses. Last but not least, we explain current solutions to practical problems in static analysis, such as the use of reflection and native code. **Topics covered include:** • Program analysis of software product lines • Modeling call stacks and field accesses with Pushdown Systems • Modeling auxiliary analysis information with Weighted Pushdown Systems • Efficiency and precision gains through Demand-driven Program Analysis • Synchronized Pushdown Systems in the Boomerang framework • Applied Android code analysis with FlowDroid • Dealing with Reflection through TamiFlex • Hybrid static and dynamic analysis with Harvester • Learning source, sink and sanitizer definitions with SWAN and SWAN Assist • Explainable static analysis Throughout, we will discuss applications to software security.
5	**Learning outcomes and competences:** After having attended this course, students will have learned. . . • how to make educated design decisions when designing automated code analysis for large-scale software systems, • which algorithms have which properties when using them to implement static code-analyses, • how to design real–world code analyses for practical problem cases from the area of IT security • how to interpret important terminology such as context, flow, field and object sensitivity • how to evaluate and explain the important limitations of static code analysis • which typical security code analyses exist (OWASP Top 10 etc.) and how they relate to the analysis frameworks explained in the course. **Non-cognitive Skills** • Learning competence • Learning motivation

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Eric Bodden

13	**Other Notes:** *Remarks of course Designing code analyses for large-scale software systems 2:* **Implementation method** Lectures and group exercises as well as programming exercises using widely used real-world static analysis frameworks (e.g. Soot, Phasar, FlowDroid) **Learning Material, Literature** • Context-, Flow-, and Field-sensitive Data-flow Analysis Using Synchronized Pushdown Systems (Johannes Späth, Karim Ali, Eric Bodden), In Proceedings of the ACM SIGPLAN Symposium on Principles of Programming Languages, pages 48:1–48:29, 3(POPL), 2019. • FlowDroid: Precise Context, Flow, Field, Object-sensitive and Lifecycle-aware Taint Analysis for Android Apps (Steven Arzt, Siegfried Rasthofer, Christian Fritz, Eric Bodden, Alexandre Bartel, Jacques Klein, Yves Le Traon, Damien Octeau, Patrick McDaniel), In Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation, pages 259–269, PLDI '14, ACM, 2014. • Codebase-Adaptive Detection of Security-Relevant Methods (Goran Piskachev, Lisa Nguyen Quang Do, Eric Bodden), In ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA), 2019. • Taming Reflection: Aiding Static Analysis in the Presence of Reflection and Custom Class Loaders (Eric Bodden, Andreas Sewe, Jan Sinschek, Hela Oueslati, Mira Mezini), In ICSE '11: International Conference on Software Engineering, pages 241–250, ACM, 2011.

Digitale Sprachsignalverarbeitung			
Digital Speech Signal Processing			
Module number: M.048.24001	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** module.frequency.en
	Semester number: 1.-3. Semester	**Duration (in sem.):** 1	**Teaching Language:** de / en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.048.24001 Digital Speech Signal Processing	2L 2Ex, SS	60	120	C	40/40

2 **Options within the module:**

None

3 **Admission requirements:**

None

Prerequisites of course Digitale Sprachsignalverarbeitung:
Recommended: Prior knowledge from the module Higher Mathematics.

4	**Contents:** *Contents of the course Digitale Sprachsignalverarbeitung:* **Short Description** The course introduces the basic techniques and theories of digital speech signal processing. A focal point of the first part of the lecture is the topic “Listening and Speaking”, which is concerned with psychological effects of human sound perception and speech production. Subsequently, time discrete signals and systems, as well as computer based data processing are discussed. Further topics are non-parametric short-time analysis of speech signals, speech coding and IP-phones. **Contents** • Listen and talk – Generating voice: human vocal tract, source filter model, vocoder – Acoustic waves – Listen: human ear, psycho acoustics and physiology of listening, loudness, acoustic occlusion, frequency groups • Time-discrete signals and systems – Basics: Elementary signals, LTI systems – Transformations: Fourier transformation of time-discrete signals, DFT, FFT – Time-discrete filtering in frequency domain: Overlap-Add, overlap-Save • Statistical speech signal analysis – Basics in theory of probabilities – Short-run analysis of speech signals: Spectrogram, cepstrum • Estimation of speech signals – Optimal filters – LPC analysis – Spectral filtering for noise suppression: spectral subtraction, Wiener filter – Adaptive Filters: LMS adaptation algorithm, echo compensation • Speech coding – Time domain coding: signal shape coding, parametric coding, hybride coding techniques – Frequency domain coding – Amplitude quantization: uniform quantization, quantization with companders (ulaw, alaw)

5 **Learning outcomes and competences:**

Domain competence:
After attending the course, the students will be able to

- analyze digital signals, e.g., audio signals, in the time or frequency domain,
- represent audio signals efficiently and
- implement widely-used algorithms for speech analysis and speech processing in the frequency or time domain.

Key qualifications:
The students

- are able to explain effects in real signals based on the theoretical knowledge,
- are able to investigate theoretical approaches by a systematic analysis and
- are, due to the precise treatment of the contents, in a position to continue their learning themselves

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or Oral Examination or Presentation	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

none

8 **Prerequisites for participation in examinations:**

None

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination (MAP) was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

BF Informationstechnik Lehramt BK affine Fächer Master v5, Masterstudiengang Computer Engineering v3 (CEMA v3), Masterstudiengang Computer Engineering v3 (CEMA v3), englisch, Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Elektrotechnik v4 und v4b (EMA v4 und v4b), Masterstudiengang Elektrotechnik v5 (EMA v5), Masterstudiengang Informatik v3, Masterstudiengang Informatik v4, Masterstudiengang Robotics and Artificial Intelligence Engineering v1 (RAIMA v1), Masterstudiengang Wirtschaftsingenieurwesen Studienrichtung Elektrotechnik, Masterstudiengang Wirtschaftsingenieurwesen Studienrichtung Elektrotechnik V4

12 **Module coordinator:**

Dr.-Ing. Jörg Schmalenströer

13	**Other Notes:** *Remarks of course Digitale Sprachsignalverarbeitung:* **Course Homepage** `https://ei.uni-paderborn.de/en/nt/teaching/veranstaltungen/digital-speech-signal-processing` **Implementation** • Lectures using the blackboard and presentations, • Alternating theoretical and practical exercise classes with exercise sheets and computer and • Demonstration of real technical systems in the lecture hall. **Teaching Material, Literature** Allocation of a script; information on textbooks ; matlab scripts

Explainable Artificial Intelligence

Explainable Artificial Intelligence

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4091	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05807 Explainable Artificial Intelligence	L2 Ex1 P2	75	105	C	20

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Explainable Artificial Intelligence:
Recommended Proficiencies
Basic knowledge in machine learning and programming

4 **Contents:**

Contents of the course Explainable Artificial Intelligence:
Explaining the predictions of machine learning models is important in an increasing number of applications. For example, bank customers would like to know why their loan was denied; machine learning engineers would like to debug and improve their models; managers would like to ensure regulatory compliance. This course aims to explain the predictions of machine learning models and introduces different explanation methods to do so. Explanation methods can be distinguished whether they are specific to a certain model or model-agnostic and whether they explain an individual prediction or the entire model.

- Introduction (e.g., importance of interpretability, evaluation of interpretability, datasets used in case studies)
- Interpretable models (e.g., linear regression, logistic regression, decision trees, decision rules)
- Global model-agnostic methods (e.g., partial dependence plots, permutation feature importance, global surrogate models)
- Local model-agnostic methods (e.g., LIME, Anchors, SHAP, counterfactual explanations)
- Model-specific methods (e.g., for neural networks)

5 **Learning outcomes and competences:**

After completing the module, students will be able to

- recognize and discuss the importance of interpretability
- explain and apply important explanation methods (e.g., interpretable models, model-agnostic methods, and model-specific methods)
- recognize characteristics of datasets, machine learning tasks, and machine learning models in application problems and argue which explanation method is appropriate for a given problem
- implement simple explanation methods from scratch
- extend and modify existing explanation methods
- discuss problems and proposed solutions with experts in the field
- read and discuss research literature in the area of XAI

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 30-45 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits.

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Dr. Stefan Heindorf

13	**Other Notes:** *Remarks of course Explainable Artificial Intelligence:* **Implementation method** Slides and blackboard writing. Important concepts and techniques will be practiced through exercises in the lecture room and tutorials, and applied in a mini-project. **Learning Material, Literature** • Slides • Exercises • Book: Christoph Molnar. Interpretable machine learning. 2020. • Additional material and literature will be announced in the course.

Foundational Methods for Knowledge Representation and Reasoning

Foundational Methods for Knowledge Representation and Reasoning

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4210	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05811 Parameterized Algorithms and Complexity	L3 Ex2	75	105	C	35

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Parameterized Algorithms and Complexity:
Recommended Proficiencies
Basic knowledge of algorithms and their analysis, complexity theory (complexity classes, Turing machines, NP-completeness) as well as first-order logic

4 **Contents:**

Contents of the course Parameterized Algorithms and Complexity:
The course gives an introduction to parameterized complexity theory. It

- gives an introduction to the foundational ideas of parameterized complexity
- covers several algorithmic techniques that help constructing efficient algorithms (with respect to parameterized complexity), such as kernelization and bounded search trees,
- covers the central concepts and tools of the hardness theory in parameterized complexity, in particular fpt-reductions as well as the W-hierarchy and A-hierarchy, and
- gives a brief introduction to fine-grained complexity.

5 **Learning outcomes and competences:**

Students will be able to

- use the technical notions covered in the lecture competently and explain these notions
- reproduce and explain the proofs and proof techniques covered in the lecture, including algorithmic techniques
- in particular, reproduce and explain proofs of correctness and running time of the covered algorithms
- apply constructive proofs and formal methods from the lecture to specific instances
- apply the proof techniques and algorithmic techniques on new, similar problems

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or presentation	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or presentation		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4

12 **Module coordinator:**

Prof. Dr.-Ing. Anni-Yasmin Turhan

13	**Other Notes:** *Remarks of course Parameterized Algorithms and Complexity:* **Implementation Method** Content is presented via beamer presentation or on white board. Exercises provide opportunities to deepen the understanding of the content and train the competences obtained in the course. The lecture in most part follows select chapters of the following literature: • *Parameterized Algorithms* by M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh (Springer 2015). • *Parameterized Complexity Theory* by J. Flum, M. Grohe (Springer 2006). • Additional literature might be announced in the course.

Foundations of Cryptography

Foundations of Cryptography

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4020	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05801 Foundations of Cryptography	L3 Ex2	75	105	C	25

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Foundations of Cryptography:
Recommended Proficiencies
Basic Knowledge in IT-Security and cryptography useful but not necessary, basic concepts of complexity theory and probability theory

4 **Contents:**

Contents of the course Foundations of Cryptography:
The most important primitives of modern cryptography will be presented. These include encryption schemes, digital signatures, identification protocols, and multiparty computations. In each case we will define precise security notions. Starting from precisely stated assumptions, we develop constructions that provably satisfy these security definitions.

- Symmetric and asymmetric encryption schemes
- Pseudorandom generators, one-way functions, trapdoor permutations
- Hashfunctions and message authentication codes
- Digital signatures, one-time signatures, random oracles
- Identification protocols, Σ protocols
- Secure multiparty computation

5 **Learning outcomes and competences:**

Students understand fundamental concepts and methods of modern cryptography. They are able to choose appropriate cryptographic tools for various security problems. Students are able to combine and modify basic cryptographic primitives, they are able to define new security concepts, they are able to the the security of new constructions with respect to the security concepts.
Non-cognitive Skills

- Commitment
- Team work
- Learning motivation
- Literacy (scientific)
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Johannes Blömer

13	**Other Notes:** *Remarks of course Foundations of Cryptography:* **Implementation method** Lectures, exercises, reading groups **Learning Material, Literature** • Oded Gorldreich, Foundations of Cryptography I,II, • Jonathan Katz, Yehuda Lindell, Introduction to Modern Cryptogtraphy • Slides from the lectures

Foundations of Knowledge Graphs

Foundations of Knowledge Graphs

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4054	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05809 Foundations of Knowledge Graphs	L2 Ex3	75	105	C	24

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Foundations of Knowledge Graphs:
Recommended Proficiencies
Graph theory, logics

4 **Contents:**

Contents of the course Foundations of Knowledge Graphs:
Knowledge graphs are used in an increasing number of applications. Large organisations such as Google, Yahoo! and the BBC rely on these technologies to organise and manage the access to the the large amounts of data they manage. This lecture aims to present approaches for building, storing, querying and using knowledge graphs, especially in intelligent applications. We will being by studying knowledge representation techniques for knowledge graphs. Technologies for querying and storing knowledge (e.g., graph databases) will be presented subsequently. Machine learning techniques for embeddings knowledge graphs and making them amenable to applications driven by classical machine learning will follow. Finally, we will study dedicated approaches for learning on knowledge graphs. The key topics of the module are hence as follows:

- Semantic networks
- RDF(S) graphs
- Property graphs
- Description logics and OWL
- Knowledge graph embeddings
- Explainable machine learning on knowledge graphs

5 **Learning outcomes and competences:**

The students can carry out the following after the completion of the module:

- Model knowledge graphs including their semantics
- Query knowledge graphs
- Compute embeddings for knowledge graphs
- Use knowledge graphs in intelligent applications

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min bzw. 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Axel-Cyrille Ngonga Ngomo

13	**Other Notes:** *Remarks of course Foundations of Knowledge Graphs:* **Implementation method** 2 SWS of lectures within which the students will be presented with novel content weekly. The lecture will be self-contained with the students being presented with the premises for understanding particular aspects of knowledge graphs as well as with the corresponding conclusions and approaches derived from these premises. 1 SWS of exercises allow the students to deal with the concepts presented in the lecture through formal analysis and programming. The 2 SWS of mini-projects ensure that the students obtain a holistic understanding of the concepts learned by applying them to a more complex task than the one addressed in the exercises. **Learning Material, Literature** Slides, homework assignments

Geometric Deep Learning

Geometric Deep Learning

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4209	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05711 Geometric Deep Learning	L2 Ex3	75	105	C	50/25

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Geometric Deep Learning:
Recommended Proficiencies
Basic knowledge in the field of machine learning can be helpful. Experience with Python and PyTorch can be helpful.

4	**Contents:** *Contents of the course Geometric Deep Learning:* This course covers principles and techniques relevant in the field of machine learning, in particular based on neural networks, whenever inputs, outputs, or tasks are involved that are decidedly geometric. The special challenges of learning on Riemannian instead of Euclidean domains are discussed. The aspect of invariances and equivariances of tasks is considered, followed by principles to explicitly take these into account in the design or the training of neural networks. Based on that, general techniques and specific neural architecture components are discussed, which enable the application of these principles depending on the relevant form of data representation. The following forms of data representation are covered: • Point clouds • Multi-view images • Voxel sets • Unfoldings • Graphs • Meshes • Distance fields • Radiance fields The following foundational principles, aspects, and techniques are covered: • Invariances and equivariances • Scale separation • Convolution • Pooling • Positional encoding
5	**Learning outcomes and competences:** Students will be able to • reproduce and implement fundamental components of neural architectures for geometric data. • design neural architectures for important types of tasks based on basic components, taking into account context and task-specific geometric priors. • explain fundamental forms of representation for geometric, in particular three-dimensional data, and compare these with respect to their suitability for input and output encoding in the context of machine learning. • demonstrate that the exploitation of geometric structures and geometric priors can be of high relevance in the context of machine learning. • analyze neural architectures in terms of their inherent invariances and equivariances.

<table>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)
<table>
<tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr>
<tr><td>a)</td><td>Written or oral examination or report</td><td>90-120 min or 30-45 min or 30 min</td><td>100%</td></tr>
</table></td></tr>
<tr><td>7</td><td>Study Achievement:
none</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Informatik v3, Masterstudiengang Informatik v4</td></tr>
<tr><td>12</td><td>Module coordinator:
Prof. Dr. Marcel Campen</td></tr>
<tr><td>13</td><td>Other Notes:
Remarks of course Geometric Deep Learning:
Implementation Method
The content is delivered through lectures using presentation slides and board notes. In-person exercises and independent study reinforce the material, supplemented by practical exercises.
Learning Material, Literature
• Lecture slides, assignments
• Additional literature will be announced in the course.</td></tr>
</table>

Geometry Processing

Geometry Processing

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4205	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05803 Geometry Processing	L3 Ex2	75	105	C	60/30

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Geometry Processing:
Recommended Proficiencies
Basic knowledge of programming, linear algebra and analysis. Experience with C++ can be helpful.

4 **Contents:**

Contents of the course Geometry Processing:
The course covers concepts, techniques, and algorithms for 3D geometry processing. It begins with the study of smooth surfaces using mathematical methods and transitions to the analysis of discrete surfaces through computer science methods. Various digital representation techniques are explored, with a particular focus on representing discrete surfaces using polygon and triangle meshes. The integration of individual algorithms into processing pipelines is discussed through applications in fields such as reverse engineering and digital fabrication.
The course includes the following contents:

- Mesh generation
- Mesh optimization
- Mesh simplification
- Mesh smoothing
- Geometric and topological shape analysis
- Deformation and interactive modeling
- Surface parameterization
- Exact geometric computing

5 **Learning outcomes and competences:**

Upon completion of the module, students will be able to

- analyze the curvature properties of parametrically defined surfaces
- analyze the curvature properties of discrete surfaces
- reproduce and implement fundamental algorithms for mesh generation, optimization, simplification, smoothing, and parameterization
- explain fundamental algorithms for mesh generation, optimization, simplification, smoothing, and parameterization and compare their strengths and weaknesses
- demonstrate that such algorithms are typically applied in the form of geometry processing pipelines and analyze these pipelines
- design context-dependent geometry processing pipelines based on fundamental algorithms
- evaluate geometric operations and algorithms for numerical risks

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3, Masterstudiengang Informatik v4

12 **Module coordinator:**

Prof. Dr. Marcel Campen

13	**Other Notes:** *Remarks of course Geometry Processing:* **Implementation Method** The content is delivered through lectures using presentation slides and board notes. In-person exercises and independent study reinforce the material, supplemented by pracical exercises. **Learning Material, Literature** • Lecture slides, assignments • Additional literature will be announced in the course.

Hardware Security

Hardware Security

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4211	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	K.079.42111 Hardware Security	L3 Ex2	75	105	C	30

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Hardware Security:
Knowledge in programming, IT security and basic knowledge in cryptography

4 **Contents:**

Contents of the course Hardware Security:
All of the software we use and rely on ends up being executed on actual hardware. The interactions between software and hardware it runs on are becoming more and more complex. Thus, secure software may turn out to be insecure when ran on particular hardware. Attackers can utilize implementation bugs, logic bugs, or side-channels like timing or cache access to break applications or extract secrets out of cryptographic implementations. This lecture will cover various topics of securing and using secure hardware. The course includes the following contents:

- Brief background in cryptography
- Secure programming and implementation bugs
- Embedded and secure hardware
- Smartcards, U2F tokens
- Timing and micro-architectural attacks
- Passive and active side-channel attacks
- Side-channel attack defences
- Trusted Platform Modules, Secure boot
- Trusted Execution Environments
- Hardware Security Modules

5 **Learning outcomes and competences:**

Students will be able to

- Identify common software and hardware vulnerabilities that undermine secure systems, including memory bugs and API misuse.
- Explain the basic ideas behind side-channel and timing attacks.
- Identify general approaches to protecting implementations against these attacks.
- Describe the architecture, threat models, and security properties of trusted hardware components like TPMs, TEEs, and smartcards.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v3 (CEMA v3), Masterstudiengang Computer Engineering v3 (CEMA v3), englisch, Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4, Masterstudiengang Robotics and Artificial Intelligence Engineering v1 (RAIMA v1)

12 **Module coordinator:**

Jan Jancar

13	**Other Notes:** *Remarks of course Hardware Security:* **Implementation Method** The topics are conveyed through lecture presentations with student participation and questions expected. Tutorials add hands-on tasks involving the material. Finally, homework assignments require individual work and self-study of additional material. **Learning Material, Literature** • Lecture and tutorial slides • The Hardware Hacking Handbook (ISBN: 9781593278748) and its associated notebooks at hardwarehacking.io • Power Analysis Attacks: Revealing the Secrets of Smart Cards (ISBN 9780387308579) • Embedded Cryptography 1 (ISBN: 9781394351879) • Embedded Cryptography 2 (ISBN: 9781394351909) • Embedded Cryptography 3 (ISBN: 9781394351930)

Human Factors in Security and Privacy			
Human Factors in Security and Privacy			
Module number: M.079.4092	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** module.frequency.en
	Semester number: null	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.079.05733 Human Factors in Security and Privacy	L2 Ex3	75	105	CE	60/20

2 **Options within the module:**

none

3 **Admission requirements:**

keine

4 **Contents:**

Contents of the course Human Factors in Security and Privacy:
Humans are important actors in security. A provable secure system is only useful if it can be actually used by users, and system designers need to account for human behavior if they wish to have both security and usability. In this class, we will examine factors of usability of security and privacy through a research-based, project-driven examination. We will cover core areas of security and privacy, as well as cover methods in human interaction (HCI) that can be used to measure the usability of security and privacy. Students are expected to complete problem sets on the topic and complete a research-based project.
The course includes the following contents:

- How to write a scientific research paper?
- How to conduct an independent scientific study in the field of Human Factors in Security and Privacy?
- Methodology: qualitative, quantitative and “mixed” methods.
- Introduction to research and scientific ethics
- Introduction to literature research
- Presenting scientific results at a conference

5 **Learning outcomes and competences:**

Students will be able to

- understand and apply research methods in human factors in usable security and privacy.
- develop relevant hypotheses and research questions in the space of usable security and privacy.
- design and deploy a research study and analyze the results.
- describe, support, and effectively argue a result using the best practices of scientific writing.
- understand ethical issues related to human factors research in security and privacy.
- understand the major topics and themes of usable security and privacy.
- present research results in class.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 30-45 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Homework assignments, project work		AA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits.

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Yasemin Acar

13	**Other Notes:** *Remarks of course Human Factors in Security and Privacy:* **Implementation Method** The contents are presented and elaborated in the lecture. In the accompanying tutorial, the lecture topics are deepened and discussed both in plenary and in small groups. In addition, students are presenting and discussing their progress on the projects. **Learning Material, Literature** • Current freely available research papers will be provided in the course. • Redmiles, Elissa M., Yasemin Acar, Sascha Fahl, and Michelle L. Mazurek. A summary of survey methodology best practices for security and privacy researchers. 2017. https://drum.lib.umd.edu/bitstream/handle/1903/19227/CS-TR-5055.pdf • Additional literature will be announced in the course.

Introduction to Description Logics

Introduction to Description Logics

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4098	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05832 Introduction to Description Logics	L3 Ex2	75	105	C	50/25

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Introduction to Description Logics:
Recommended Proficiencies
Knowledge of contents from the modules *Modelling*, *Computatbility and Complexity* and *Complexity Theory* are beneficial – in particular knowedge on predicate logic and the fundamental complexity classes.

4 **Contents:**

Contents of the course Introduction to Description Logics:
This lecture and the tutorials introduce Description Logics. In detail we cover the following content:

- introduction of the Description Logic *ALC* and concept and role operators to extend *ALC*.
- introduction of DL knowledge bases and fundamental reasoning problems for DLs.
- relation of *ALC* to predicate logic and to modal logic
- model theory of *ALC*
- tableau algorithm for satisfiability in *ALC*
- Complexity analysis of reasoning in *ALC*
- reasoning methods for variants of the DL *EL*
- Query answering in DLs and methods for computing such answers

5 **Learning outcomes and competences:**

Students will be able to

- use technical notions from the lecture competently and explain these notions
- master the syntax and semantics of the main elements of description Logic knowledge bases and their reasoning problems
- apply reasoning algorithms and can assess them in regard of computational complexity
- reconstruct proof methods for description logics and can apply them to instances

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 30-45 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3, Masterstudiengang Informatik v4

12 **Module coordinator:**

Prof. Dr.-Ing. Anni-Yasmin Turhan

13	**Other Notes:** *Remarks of course Introduction to Description Logics:* **Implementation Method** The course proceeds after the text book “An introduction to Description Logic”. The lecture is mainly presenting the content by slides. Proofs will be written at the black board or covered as “flipped class room”. The tutorials deepen and complement the content of the lecture. **Learning Material and literature** • “An Introduction to Description Logic” by Franz Baader, Ian Horrocks, Carsten Lutz, Uli Sattler (This book is freely available as E-Book, if accessed from the network of Paderborn university.) • Additional literature will be announced in the course.

Introduction to Quantum Computation

Introduction to Quantum Computation

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4059	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05807 Introduction to Quantum Computation	L3 Ex2	75	105	C	40

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Introduction to Quantum Computation:
Recommended Proficiencies
Linear Algebra, algorithms.

4 **Contents:**

Contents of the course Introduction to Quantum Computation:
This lecture introduces the fundamental concepts of quantum computation and information from a computer science perspective. This includes an introduction to quantum mechanics, quantum entanglement, quantum algorithms, quantum error correction, and quantum information theory.

- Quantum mechanics
- Quantum entanglement
- Quantum algorithms
- Quantum error correction
- Quantum information

5 **Learning outcomes and competences:**

Students are able to:

- Describe and apply the postulates of quantum mechanics
- Understand the use of entanglement as a resource
- Design and analyze fundamental quantum algorithms
- Apply the theory of error-correcting codes
- Understand and apply basic quantum information theory concepts such as entropy

Non-cognitive Skills

- Learning competence
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	120-180 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Sevag Gharibian

13	**Other Notes:** *Remarks of course Introduction to Quantum Computation:* **Implementation method** Slides and blackboard writing. All important concepts and techniques are further deepened with examples in exercises. **Learning Material, Literature** • Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press • Lecture slides, exercises

Logic Programming for Artificial Intelligence (MA v3)

Logic Programming for Artificial Intelligence

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4031	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05808 Logic Programming for Artificial Intelligence	L3 Ex2	75	105	CE	40/20

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Logic Programming for Artificial Intelligence:
Recommended Proficiencies
Students should have previous knowledge in programming as offered in the courses “Programmierung” and “Programming Languages” and knowledge in database query languages as offered in the course “Database Systems”.

4 **Contents:**

Contents of the course Logic Programming for Artificial Intelligence:
This course views various concepts and techniques from computer science, artificial intelligence, and computational linguistics from a different perspective, i.e. the perspective of programming in logic. Programming in logic in general and the programming language Prolog in particular offer the ability to describe many concepts in logic, i.e. in a declarative way, and to have them tested and executed by an interpreter at a same time. This is in particular useful for puzzles and quizes, but also for self-defined or domain specific languages.

- Introduction into logic programming using the Prolog language
- Constraint solvers, puzzles, and theorem provers
- Interpreters for term substitution systems
- Parsing programs, XML, and natural language
- Semantics construction, question answering systems, and text translation
- Meta interpreters, domain specific languages, and programming in “natural language”
- Feature term unification and applications in computer linguistics and ecommerce

5 **Learning outcomes and competences:**

Students learn
factual knowledge about

- the transformation of knowledge given as facts and rules into an executable programs
- how to program in logic and in self-designed languages

methodological knowledge, including

- the ability to define domain specific languages
- the ability to implement interpreters for domain specific languages
- the ability to develop small question answering systems
- the ability to develop software for theorem provers or constraint solvers solving puzzles

transfer skills

- the ability to transfer the methodolies and skills gained to other data sources, knowledge representation formats, or calculi
- the ability to transfer the parsing and semantics knowledge to domain specific languages

normative evaluation skills including the ability to assess

- the suitability and limitations of different data and knowledgerepresentation formats for different tasks
- the suitability of different programming paradigms for different projects
- the effort and feasibility of projects aiming natural language understanding
- the effort and feasibility of projects aiming at automated translation

Non-cognitive Skills

- Learning competence
- Learning motivation

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8	**Prerequisites for participation in examinations:** Passing of course achievement
9	**Prerequisites for assigning credits:** The credit points are awarded after the module examination was passed.
10	**Weighing for overall grade:** The module is weighted according to the number of credits (factor 1).
11	**Reuse in degree courses or degree course versions :** Masterstudiengang Informatik v3
12	**Module coordinator:** Prof. Dr. Stefan Böttcher
13	**Other Notes:** *Remarks of course Logic Programming for Artificial Intelligence:* **Implementation method** The theoretical concepts are explained in the lectures and consolidated in small groups during tutorials. The tutorials are carried out as practical exercises on the computer. **Learning Material, Literature** • Ivan Bratko: Prolog Programming for Artificial Intelligence. Pearson Education, Newest Edition. • Links to further material will be provided in the lecture.

Machine Learning 1

Machine Learning 1

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4032	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05701 Machine Learning 1	L3 Ex2	75	105	C	75/25

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Machine Learning 1:
Recommended Proficiencies
Basic knowledge in mathematics (linear algebra, statistics), programming and algorithms.

4 **Contents:**

Contents of the course Machine Learning 1:
Due to the ever increasing amount of data that is routinely produced in our information society, the topic of machine learning has become increasingly important in the recent years, not only as a scientific discipline but also as a key technology of modern software and intelligent systems. This lecture provides an introduction to the topic of machine learning, with a specific focus on supervised learning for classification and regression. The lecture covers theoretical foundations of generalisation as well as practical topics and concrete learning algorithms.

- Introduction
- Foundations (e.g., the learning problem, generalization theory, bias-variance tradeoff)
- Techniques (e.g., The linear model, non-linear techniques, SVM, tree-based methods, ensembles, deep learning)
- Validation and practical implementations (e.g., metrics, training vs testing, cross-validation, AutoML)

5 **Learning outcomes and competences:**

The students

- understand the statistical foundations of generalisation, i.e., the induction of models from data, and the complete machine learning pipeline.
- implement basic methods of supervised learning for classification and regression, apply them to real-world examples, and make necessary adaptations.
- understand and use practical tools for model validation and automated machine learning pipelines.
- assess the advantages and disadvantages of using various machine learning techniques in various situations.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4, Masterstudiengang Robotics and Artificial Intelligence Engineering v1 (RAIMA v1)

12 **Module coordinator:**

Prof. Dr. Heike Trautmann

13	**Other Notes:** *Remarks of course Machine Learning 1:* **Implementation method** Theoretical foundations and concepts of machine learning will be taught in the form of a lecture and deepened in practical exercise courses, group work as well as individual homework. **Learning Material, Literature** • Script • Y.S. Abu-Mostafa, M. Magdon-Ismail, H.T. Lin. Learning from Data, AMLBook, 2012. • P. Flach. Machine Learning, Cambridge Univ. Press, 2012. • E. Alpaydin. Machine Learning, Oldenbourg, 2008. • C.M. Bishop. Pattern Recognition and Machine Learning, Springer, 2006.

Model-Based Systems Engineering

Model-Based Systems Engineering

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4062	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	de

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05815 Model-Based Systems Engineering	L3 Ex2	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Model-Based Systems Engineering:
Recommended Proficiencies
Basics of Systems Engineerings

4 **Contents:**

Contents of the course Model-Based Systems Engineering:
The goal of the lecture is a comprehensive understanding of Model-Based Systems Engineering (MBSE) and its components. The students are taught the essential topics of MBSE. This includes fundamentals including languages, methods and IT tools, which are also tested in practice. The benefits of MBSE (an understanding of the system by all involved actors, a basis for communication and cooperation between different disciplines but also functional areas, . . .) will be conveyed to the students. Furthermore, essential analysis methods for testing system designs are covered. The focus is on multidisciplinary, software-intensive systems from the mechanical and plant engineering and automotive industries.

- Basics of MBSE
- SysML for multidisciplinary systems
- CONSENS
- further MBSE approaches
- design patterns
- MBSE Tools
- analysis methods based on the system model

5 **Learning outcomes and competences:**

Learning Outcomes
Students will be able to,

- Work in a model-based manner
- Apply systems thinking
- Create system architectures & derive requirements.

Non-Cognitive Competencies

- Self-monitoring
- Literacy (scientific)
- Learning competence
- Learning motivation

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

none

8 **Prerequisites for participation in examinations:**

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr.-Ing. Roman Dumitrescu

13	**Other Notes:** *Remarks of course Model-Based Systems Engineering:* **Implementation method** The module consists of three parts 1. lecture with slides: basics and concepts are explained in the lecture and illustrated with examples. 2. exercises (tutorial): In the exercises, knowledge is transferred and the concepts are applied. The exercises have to be prepared by the students themselves. 3. practical course (labs): In the practical course, the application of what has been learned takes place in group work. **Learning Material, Literature** • Friedenthal, S.; Moore, A.; Steiner, R.: A Practical Guide to SysML. The Systems Modeling Language. Morgan Kaufmann, Waltham, 2. Auflage, 2012 • Gausemeier, J.; Rammig, J.; Schäfer, W. (Eds.): Design Methodology for Intelligent Technical Systems. Develop Intelligent Technical Systems of the Future. Springer-Verlag, 2014 • Gausemeier, J.; Dumitrescu, R.; Steffen, D.; Czaja, A.; Wiederkehr, O.; Tschirner, C.: Systems Engineering in industrial practice. Heinz Nixdorf Institute, University • Haberfellner, R., L., D. W. O., Fricke, E., & Voössnersiegfried. (2019). Systems engineering: fundamentals and applications. Cham: Springer International Publishing • Incose Systemsengineeringhandbook: A Guide for System Life Cycle Processes and Activities(2015) • Weilkiens, Tim: Systems Engineering with SysML/UML: Modeling, Analysis, Design (The MK/OMG Press) (English Edition) • Dumitrescu, R.; Albers, A.; Riedel, O.; Stark, R.; Gausemeier, J. (Hrsg.): Engineering in Deutschland – Status quo in Wirtschaft und Wissenschaft, Ein Beitrag zum Advanced Systems Engineering, Paderborn, 2021 – English Version: www.advanced-systems-engineering.de

Multi-Objective Optimisation

Multi-Objective Optimisation

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4095	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05821 Multi-Objective Optimisation	L3 Ex2	75	105	C	30/15

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Multi-Objective Optimisation:
Recommended Proficiencies
Solid basic knowledge of algorithms and data structures, mathematics, as well as basic knowledge of optimization are beneficial.

4 **Contents:**

Contents of the course Multi-Objective Optimisation:
Optimization problems are ubiquitous, and we all (approximately) solve them in everyday life, such as when finding routes with Google Maps to quickly get from point A to point B or deciding on a checkout lane with the shortest waiting queue (shortest expected waiting time) at the supermarket. However, optimization problems are rarely single-criteria. Instead, they are typically multi-criteria in nature, with the individual objectives usually conflicting with each other. For example, in route planning, the distance traveled may be relevant (shorter is better), and fuel consumption may also be a consideration (lower is better). The shortest route may lead through the city center with many stop-and-go maneuvers at red lights, especially during peak hours. On the other hand, a longer route around the city may consume less fuel. Accordingly, the goal in multi-objective optimization is to find a set of optimal compromise solutions.
This course provides a comprehensive introduction to multi-objective optimization and the associated challenges. In addition to classical general approaches, exact methods for selected combinatorial optimization problems are presented, along with heuristic (nature-inspired) methods. The course also covers heuristic solution approaches for problems with more than three criteria (many-objective optimization).

5 **Learning outcomes and competences:**

Students will be able to

- Explain, implement, and apply important exact algorithms for multi-criteria minimum spanning tree problems and multi-criteria shortest path problems
- Understand the limitations of exact algorithms for multi-criteria problems
- Explain and apply biologically inspired heuristics for multi-objective problems
- Assess, evaluate, and visualize the quality of computed results from multi-criteria algorithms
- Understand the challenges of problems with more than three criteria and explain solution approaches

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4

12 **Module coordinator:**

Prof. Dr. Heike Trautmann

13	**Other Notes:** *Remarks of course Multi-Objective Optimisation:* **Implementation Method** Slide-based lecture with interspersed assignments. In the tutorial, the knowledge transfer and application of what has been learned takes place in both theoretical and practical assignments. **Learning Material, Literature** • Deb, Kalyanmoy. „Multi-Objective Optimization Using Evolutionary Algorithms“. • Ehrgott, Matthias. Multicriteria Optimization. Bd. 491. Lecture Notes in Economics and Mathematical Systems. Berlin, Heidelberg: Springer, 2000. • Additional literature will be announced in the course.

Post-Quantum Cryptography

Post-Quantum Cryptography

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4089	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05732 Post-Quantum Cryptography	L3 Ex2	75	105	C	20

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Post-Quantum Cryptography:
Recommended Proficiencies
Basics of cryptography and complexity theory

4 **Contents:**

Contents of the course Post-Quantum Cryptography:
IT security is largely based on modern cryptographic methods. These include many methods of so-called public-key cryptography such as the RSA and Elgamal encryption methods, the RSA signature method, and the various variants of the Digital Signature Algorithm (DSA). In 1994, Peter Shor presented an efficient algorithm for computing prime factorization of integers and for computing discrete logarithms in finite groups. Thus, all the aforementioned methods of public-key cryptography are insecure if quantum computers of sufficient size and complexity can be realized. It is therefore important to develop alternatives to classical public-key methods that, at least according to current research, cannot be broken by quantum computers. Important candidates (and some close to standardization) for such post-quantum secure methods rely on techniques of error-correcting codes and the geometry of numbers. In this lecture, we will present and discuss important candidates for post-quantum secure methods. The course includes the following contents:

- introduction to codes, lattices and discretised Gaussian distributions
- lattice and code based enryption
- lattice based signatures
- lattices and zero-knowledge proofs
- lattice based group signatures

5 **Learning outcomes and competences:**

Students will be able to

- understand and explain the difference between classical and post-quantum security.
- explain the importance of post-quantum cryptography for selected applications.
- explain and apply concepts from the field of geometry of numbers and error-correcting codes.
- explain important constructions from post-quantum cryptography and prove their security.
- explain security assumptions from post-quantum cryptography and apply them to new post-quantum primitives.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits.

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Johannes Blömer

13	**Other Notes:** *Remarks of course Post-Quantum Cryptography:* **Implementation Method** Basic concepts are presented in a lecture. In addition, theoretical concepts are deepened in tutorials in small groups as well as in written exercises. **Learning Material, Literature** References to current learning materials will be given in the lectures.

Privacy and Technology

Privacy and Technology

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4087	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05705 Privacy and Technology	L2 Ex3	75	105	C	40

2 **Options within the module:**

none

3 **Admission requirements:**

keine

4 **Contents:**

Contents of the course Privacy and Technology:
This course provides students with a basic understanding of privacy risks, the most common technologies for addressing them and the human factors that shape their design.

- Privacy metrics and adversary models
- Anonymous communications
- Data-perturbative privacy-enhancing technologies
- Anonymization algorithms for databases
- Homomorphic encryption and zero knowledge proofs
- Selective disclosure for identity management
- Usable privacy
- Applying privacy principles and case studies

5 **Learning outcomes and competences:**

The students

- are able to reason critically about privacy,
- gain knowledge in the evaluation of privacy risks,
- understand the design aspects of privacy-enhancing technologies,
- get familiar with the latest research in the field and
- analyze and discuss the space of solutions to a given privacy problem

Non-cognitive Skills

- Literacy (scientific)
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Practical work and discussion		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Patricia Arias Cabarcos

13	**Other Notes:** *Remarks of course Privacy and Technology:* **Implementation method** • Weekly theory lecture • Exercise and assignments to interactively discuss in the practical lectures **Learning Material, Literature** • Lecture slides, scientific literature and specific readings will be provided during the course.

Quantum Algorithms

Quantum Algorithms

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4072	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05797 Quantum Algorithms	L3 Ex2	75	105	C	20

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Quantum Algorithms:
Recommended Proficiencies
Linear Algebra, Quantum Computing

4 **Contents:**

Contents of the course Quantum Algorithms:
This lecture covers quantum algorithms from a computer science perspective. Topics include quantum circuits (e.g. Solovay-Kitaev theorem), quantum algorithms for algebraic problems (e.g. Hidden Subgroup problem), quantum walks, quantum query complexity, and adiabatic quantum computing.

- Quantum circuits
- Algebraic problems
- Quantum walks
- Query complexity
- Adiabatic computation

5 **Learning outcomes and competences:**

Students are able to:

- Describe universal gate sets
- Develop Quantum Fourier-Transform based algorithms
- Develop quantum walk-based algorithms
- Apply the quantum adiabatic theorem
- Give quantum query lower bounds

Non-cognitive Skills

- Learning competence
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written Exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Sevag Gharibian

13	**Other Notes:** *Remarks of course Quantum Algorithms:* **Implementation method** Slides and blackboard writing. All important concepts and techniques are further deepened with examples in exercises. **Learning Material, Literature** • Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press • Andrew M. Childs, Wim van Dam, Quantum algorithms for algebraic problems, Reviews of Modern Physics, volume 82, 2010 • Lecture slides, exercises

Quantum Complexity Theory

Quantum Complexity Theory

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4063	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05743 Quantum Complexity Theory	L3 Ex2	75	105	C	20

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Quantum Complexity Theory:
Recommended Proficiencies
Linear Algebra, Quantum Computing

4 **Contents:**

Contents of the course Quantum Complexity Theory:
This lecture provides a brief review of introductory quantum computation, and subsequently moves into quantum complexity theory. Beginning to advanced topics will be covered, including quantum analogues of P and NP (denoted BQP, QCMA, and QMA), quantum satisfiability problems, quantum interactive proofs, and tensor networks. Along the way, semidefinite programming will be introduced as an important tool.

- Complexity classes BQP, QCMA, QMA
- Quantum Satisfiability Problems
- Quantum Interactive Proofs
- Tensor Networks
- Semidefinite Programming

5 **Learning outcomes and competences:**

Students are able to:

- Describe and apply the postulates of quantum mechanics
- Work with complexity classes such as BQP and QMA
- Show QMA-hardness of computational problems
- Apply semidefinite programming techniques
- Use tensor networks to model entangled quantum states

Non-cognitive Skills

- Learning competence
- Self-monitoring

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Sevag Gharibian

13	**Other Notes:** *Remarks of course Quantum Complexity Theory:* **Implementation method** Slides and blackboard writing. All important concepts and techniques are further deepened with examples in exercises. **Learning Material, Literature** • Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press • S. Gharibian, Y. Huang, Z. Landau, S. W. Shin, Quantum Hamiltonian Complexity, Foundations and Trends in Theoretical Computer Science • Lecture slides, exercises

Quantum Information

Quantum Information

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4090	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	Quantum Information	L3 Ex2	75	105	C	40

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Quantum Information:
Recommended Proficiencies
Linear Algebra

4 **Contents:**

Contents of the course Quantum Information:
Over the last century, Quantum mechanics has had profound impacts on both fundamental science and technology. The emerging field of Quantum Information Theory studies a paradigm for information processing empowered by quantum mechanics. This field has demonstrated that quantum information processing can outperform its classical counterpart and is a revolutionary direction to investigate future information technologies. Quantum Information Science incorporates techniques from computer science, mathematics, and physics.
Of particular interest is quantum entanglement, which is the phenomenon that occurs when a group of particles is generated or interacts in a way such that the state of each particle cannot be described independently of the others, even when the particles are separated by arbitrarily large distances. Entanglement is a primary feature of quantum mechanics not present in classical physics and it is a resource behind most modern quantum technologies, such as quantum computers. This lecture introduces the advance concepts of quantum communication and information.
The contents include:

- Entanglement of two- and many-body systems
- Quantum information processing and applications
- Measures of Entanglement, Distance and Fidelity
- Higher local dimensions (qubits vs qudits)
- Quantum channels
- Classical and quantum error correcting codes and their differences

5 **Learning outcomes and competences:**

Students learn cutting-edge concepts at the intersection of computer science and quantum mechanics. This lecture equips students with advanced, interdisciplinary technical proficiency, enabling them to pursue careers in analysis intensive industries, technology start-ups, or research and development roles in leading technology companies or academia. To achieve this, the students get familiar with the basics of quantum mechanics and the related algebra. Furthermore, they will be able to:

- understand the underlying concepts of entangled systems (two-body and many-body),
- understand the fundamental idea of maximally entangled systems, classify and characterise them for practical applications,
- describe the basic notion of higher local dimension particles (qubits vs qudits),
- apply the theory of classical and quantum error correcting codes, and study their differences,
- to work on interdisciplinary topics and, in particular, to acquire the basics of different disciplines.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	120-180 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits.

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12	**Module coordinator:** Dr. Zahra Raissi
13	**Other Notes:** *Remarks of course Quantum Information:* **Implementation Method** Theoretical foundations and concepts will be taught in the form of lectures and deepened in practical exercise courses, group work as well as individual homework. **Learning Material, Literature** • Michael A. Nielsen, Isaac L. Chuang, Quantum Computation and Quantum Information, Cambridge University Press, 2000. • F. J. MacWilliams and N. J. A. Sloane. The Theory of Error-Correcting Codes, North-Holland Mathematical Library. North-Holland, Amesterdam, 1977. ISBN 9780444851932. • Ingemar Bengtsson and Karol Zyczkowski, Geometry of quantum states: an introduction to quantum entanglement, Cambridge university press, 2006, ISBN 9780511535048. • Lecture slides • Exercises

Real-Time Operating Systems: Design and Implementations

Real-Time Operating Systems: Design and Implementations

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4213	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1-3	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	Real-Time Operating Systems: Design and Implementations	L2 Ex3	75	105	C	70/35

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Real-Time Operating Systems: Design and Implementations:
Recommended Proficiencies
Knowledge of operating systems, embedded systems, and C/C++ programming is beneficial. Basic familiarity with computer architecture and concurrent programming is helpful but not required.

4 **Contents:**

Contents of the course Real-Time Operating Systems: Design and Implementations:
Real-time operating systems are essential for embedded and cyber-physical systems in which timely and predictable responses are required. This module introduces the design and implementation principles of real-time operating systems. It combines conceptual foundations with practical exercises in which students analyze, extend, and evaluate mechanisms of a real-time operating system on an embedded platform.
The course includes the following contents:

- real-time systems and timing requirements
- architecture and basic services of real-time operating systems
- task models, task states, and task management
- periodic execution and timer mechanisms
- inter-process communication and synchronization
- interrupt handling and interaction between tasks and interrupts
- context switching and task execution
- fixed-priority and dynamic-priority scheduling
- implementation of scheduling mechanisms, e.g. EDF
- resource sharing and synchronization protocols, e.g. PIP and PCP
- server-based scheduling and reservation mechanisms
- practical implementation and evaluation of RTOS mechanisms on an embedded platform

5 **Learning outcomes and competences:**

Students will be able to

- explain the role, structure, and services of real-time operating systems in embedded and cyber-physical systems;
- analyze timing requirements, task models, scheduling decisions, and resource-sharing situations in real-time systems;
- apply real-time scheduling and synchronization concepts to concrete system examples;
- implement selected RTOS mechanisms, such as scheduling or resource-sharing protocols, on an embedded platform;
- evaluate the behavior of RTOS mechanisms with respect to timing, resource management, and predictability;
- reason about design trade-offs between functionality, timing predictability, implementation complexity, and system robustness.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments, course paper or progress reports		CA

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4

12 **Module coordinator:**

Prof. Dr.-Ing. Kuan-Hsun Chen

13	**Other Notes:** *Remarks of course Real-Time Operating Systems: Design and Implementations:* **Implementation Method** The module combines lectures with practical tutorials and lab-oriented assignments. Lectures introduce the conceptual and analytical foundations of real-time operating systems. In the tutorials, students deepen these concepts through exercises and implement selected RTOS mechanisms on an embedded platform. The practical work is used to connect theoretical concepts with observable system behavior. **Learning Material, Literature** • Giorgio C. Buttazzo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, Springer. • Jane W. S. Liu, Real-Time Systems, Prentice Hall. • Hermann Kopetz, Real-Time Systems: Design Principles for Distributed Embedded Applications, Springer. • FreeRTOS documentation and selected technical documentation will be used for the practical assignments. • Additional literature will be announced in the course.

Real World Crypto Engineering

Real World Crypto Engineering

Module number:	Workload (h):	Credits:	Regular Cycle:
M.079.4067	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.079.05819 Real World Crypto Engineering	L3 Ex2	75	105	C	40

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Real World Crypto Engineering:
Recommended Proficiencies
Knowledge in programming, IT security and basic knowledge in cryptography

4 **Contents:**

Contents of the course Real World Crypto Engineering:
Strong cryptography is not always sufficient to protect primary security goals. Even if strong cryptographic algorithms are used, a lot can go wrong when they are implemented. This lecture will dive into the most important protocols and cryptographic protection mechanisms (e.g., TLS, SSH, WPA) and show their basic concepts. Then, we will present prominent attacks that ultimately break the desired security goals. Based on many cases, we will learn what is essential when designing and implementing cryptographic applications.

5 **Learning outcomes and competences:**

Upon successful completion, students have a comprehensive understanding of the technical aspects of applied cryptographic algorithms. They have recognized that cryptography alone is not sufficient to solve security-related problems. They have an overview of current cryptographic attacks and know how to practically prevent them.
Non-cognitive Skills

- Team work
- Literacy (scientific)

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr.-Ing. Juraj Somorovsky

13 **Other Notes:**

Remarks of course Real World Crypto Engineering:
Implementation method
Lectures, exercises
Learning Material, Literature
Lecture slides, scientific papers

Reconfigurable Computing

Reconfigurable Computing

Module number:	Workload (h):	Credits:	Regular Cycle:
M.079.4043	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.079.05703 Reconfigurable Computing	L2 Ex3	75	105	CE	50/20

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Reconfigurable Computing:
Recommended Proficiencies
Knowledge of “Digital Design” and “Computer Architecture” is beneficial.

4 **Contents:**

Contents of the course Reconfigurable Computing:
This lecture provides an understanding of architectures and design methods for reconfigurable hardware systems and presents applications in the areas of high performance computing and embedded systems.

- Introduction: evolution of programmable logic devices, market economics
- Architectures: FPGA architectures, reconfigurable devices, reconfigurable systems
- Design methods: CAD for FPGAs, high-level languages and compilers, system-level design
- Applications: custom computing machines, embedded systems

5 **Learning outcomes and competences:**

After attending the course, the students are able to

- explain the architectures of reconfigurable hardware devices,
- name and analyze the main design methods and
- judge the suitability of reconfigurable hardware for different application domains.

Non-cognitive Skills

- Team work
- Learning competence

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Marco Platzner

13	**Other Notes:** *Remarks of course Reconfigurable Computing:* **Implementation method** • Lecture with projector and board • Interactive exercises in the lecture room • Computer-based exercises with reconfigurable systems **Learning Material, Literature** • Lecture slides and exercise sheets • Exercise sheets and technical documentation for the for the computer-based exercises • S. Hauck and A. DeHon (editors): Reconfigurable Computing, Volume 1: The Theory and Practice of FPGA-Based Computation, Morgan Kaufmann, 2008 • Information about alternative and additional literature as well as teaching material on the course's website and in the lecture slides

Resource-Efficient AI: Architectures and Methodologies

Resource-Efficient AI: Architectures and Methodologies

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.048.55106	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1.-3. Semester	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.048.23023 Resource-Efficient AI: Architectures and Methodologies	2L 2Ex, WS	60	120	C	30/30

2 **Options within the module:**

None

3 **Admission requirements:**

None

Prerequisites of course Resource-Efficient AI: Architectures and Methodologies:
None

4 **Contents:**

Contents of the course Resource-Efficient AI: Architectures and Methodologies:
The course introduces methods and architectures for resource-efficient execution of modern artificial intelligence applications under performance, energy, and memory constraints. Topics include:

- Fundamentals of deep learning and training methodologies
- Characteristics of modern AI workloads (e.g., CNNs and transformers)
- Computational, memory, and energy bottlenecks during AI inference
- Techniques for efficient model design and optimization:
  - Quantization
  - Pruning and sparsity
  - Knowledge distillation
  - Efficient neural network architectures (e.g., MobileNet)
- Mapping AI workloads onto hardware platforms
- Architectures for AI acceleration:
  - SIMD and vector architectures
  - GPU architectures
  - Dataflow architectures and systolic arrays
  - Neural Processing Units (NPUs) and domain-specific accelerators
- Hardware-software co-design for efficient AI systems

5 **Learning outcomes and competences:**

After successful completion of the module, students are able to:

- explain fundamental principles of deep learning from a systems perspective,
- analyze AI workloads with respect to computational, memory, and communication requirements,
- identify performance, energy, and memory bottlenecks,
- apply model optimization techniques to improve efficiency,
- map AI workloads onto different hardware platforms and evaluate their suitability,
- compare different architectures for AI acceleration,
- analyze trade-offs between accuracy, latency, throughput, and energy efficiency,
- apply hardware-software co-design methodologies for resource-efficient AI systems.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or Oral Examination or Presentation	120-180 min or 30-45 min or 30 min	100%

<table>
<tr><td>7</td><td>Study Achievement:
<table>
<tr><th>zu</th><th>Type of achievement</th><th>Duration or Scope</th><th>SL / QT</th></tr>
<tr><td>a)</td><td>Presentation</td><td>10-20 min</td><td>AA</td></tr>
</table></td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
Prerequisite for the participation in the module final examination is the passing of the academic achievement on the course a).</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination (MAP) was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Informatik v3, Masterstudiengang Informatik v4</td></tr>
<tr><td>12</td><td>Module coordinator:
Prof. Dr.-Ing. Lennart Bamberg</td></tr>
<tr><td>13</td><td>Other Notes:
Module Homepage
PANDA (more information will be provided in the first lecture)
Implementation
Lectures and exercises (including some computer simulations)
Teaching Material, Literature
Handouts and tutorial questions; literature references will be given in the first lecture</td></tr>
</table>

Software Architecture Design and Recovery

Software Architecture Design and Recovery

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4094	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05708 Software Architecture Design and Recovery	L2 Ex3	75	105	CE	30

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Software Architecture Design and Recovery:
Recommended Proficiencies
A good understanding of Java and the principle of object-oriented programming is helpful.

4 **Contents:**

Contents of the course Software Architecture Design and Recovery:
Software architecture is concerned with the principal design decisions of a software system. These decisions have significant impact on the system's quality, such as maintainability, performance and security. This course will explain fundamental concepts of the software architecture field, as well as how to apply techniques to recover design decisions from existing software repositories. The course includes the following topics from software architecture field:

- Types of design decisions.
- Architectural components and recovery.
- Architectural solutions such as patterns, tactics and technologies.
- Architectural documentation.
- Software repositories.
- Architectural knowledge.
- Design processes.

Furthermore, the course discusses and applies common research methods:

- Grounded theory
- Case studies

Concepts are discussed in the lectures and applied using a set of group assignments on real open-source software systems.

5 **Learning outcomes and competences:**

Students will be able to

- clarify and discuss main concepts in the software architecture field, such as architectural solutions, components, and design processes
- analyze large-scale software systems for architectural design decisions
- execute design processes to make design decisions
- apply common research methods on software architecture problems
- summarize and report research results in a scientific format
- work in teams
- present their results to the audience

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

<table>
<tr><td>7</td><td>Study Achievement:
<table>
<tr><th>zu</th><th>Type of achievement</th><th>Duration or Scope</th><th>SL / QT</th></tr>
<tr><td>a)</td><td>Assignments</td><td></td><td>AA</td></tr>
</table>
Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
Passing of course achievement</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Informatik v3</td></tr>
<tr><td>12</td><td>Module coordinator:
Dr. Mohamed Aboubakr Mohamed Soliman</td></tr>
<tr><td>13</td><td>Other Notes:
Remarks of course Software Architecture Design and Recovery:
Implementation Method
Lectures and group assignments on large open-source software systems, as well as presentations.
Learning Material, Literature
• Bass, L., Clements, P., Kazman, R. (2012). Software Architecture in Practice. 3rd Edition, Addison-Wesley Professional.
• Kruchten P, Lago P, van Vliet H (2006) Building Up and Reasoning About Architectural Knowledge. In: Quality of Software Architectures, Springer Berlin Heidelberg.
• Additional literature will be announced in the course.</td></tr>
</table>

Software Quality Assurance			
Software Quality Assurance			
Module number: M.079.4048	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** module.frequency.en
	Semester number: null	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.079.05805 Software Quality Assurance	L3 Ex2	75	105	CE	90/30

2 **Options within the module:**

none

3 **Admission requirements:**

none

Prerequisites of course Software Quality Assurance:
Recommended Proficiencies
Programming, Modeling, Model-based software development

4 **Contents:**

Contents of the course Software Quality Assurance:
The aim of the lecture is to cover approaches, technologies and strategies related to quality assurance for software systems. These include on the one hand constructive approaches such as design patterns, anti-patterns, domain-specific languages, model driven development, model quality analysis, and architectural styles, and on the other hand analytic approaches such as static reviewing techniques and dynamic testing techniques.. Furthermore, approaches for the improvement of the software development process and international standards like ISO 9001, 9126, CMM etc. are covered.

- Introduction to software quality assurance
- Standards
  - Product-related Standards: ISO 9126
  - Process-related Standards: ISO 9001, CMM
- Constructive approaches
  - Patterns and styles: Design patterns, Anti-Patterns, Architectural styles
  - Model-driven development
  - Metamodeling
  - Domain Specific Languages
  - Design by contract
  - Research: Process constraints
- Analytical approaches
  - Reviews, inspections
  - Testing: Fundamental Test Process, Black Box Testing, White Box Testing

5 **Learning outcomes and competences:**

The students are able to explain quality characteristics of software development processes, software models as well as software systems. They have understood constructive and analytical techniques used to ensure quality properties, and they are able to apply them. They can describe standards for measuring process and product quality. They are able to understand new research approaches in the area of process and product quality.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination or report	90-120 min or 30-45 min or 30 min	100%

<table>
<tr><td>7</td><td>Study Achievement:
<table>
<tr><th>zu</th><th>Type of achievement</th><th>Duration or Scope</th><th>SL / QT</th></tr>
<tr><td>a)</td><td>Assignments, course paper or progress reports</td><td></td><td>CA</td></tr>
</table></td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
Passing of course achievement</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Informatik v3, Masterstudiengang Informatik v4</td></tr>
<tr><td>12</td><td>Module coordinator:
Dr. Enes Yigitbas</td></tr>
<tr><td>13</td><td>Other Notes:
Remarks of course Software Quality Assurance:
Implementation method
Partially slides and partially board writing. All essential concepts and techniques will be repeatedly applied in examples during the tutorial. In a lab part, the techniques will be employed using tools, particularly testing tools.
Learning Material, Literature
• Daniel Galin: Software Quality Assurance: From Theory to Implementation, Pearson / Addison Wesley, 2004
• Slides, Exercises</td></tr>
</table>

Statistical Signal Processing

Statistical Signal Processing

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.048.92004	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1. Semester	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.048.92004 Statistical Signal Processing	2L 2Ex, WS	60	120	C	30/30

2 **Options within the module:**

None

3 **Admission requirements:**

None

Prerequisites of course Statistical Signal Processing:
Recommended: Undergraduate courses in signal processing and probability

4 **Contents:**

Contents of the course Statistical Signal Processing:
Short Description
Statistical signal processing comprises the techniques that engineers and statisticians use to draw inference from imperfect and incomplete measurements. This course covers a selection of topics from the major domains of detection, estimation, and time series analysis.
Contents
Topics that may be covered in this course include correlation analysis, linear minimum mean-squared error estimation, performance bounds for parameter estimation, Neyman-Pearson detectors, wide-sense stationary, nonstationary and cyclostationary time series, and complex-valued random signals.

5 **Learning outcomes and competences:**

After attending this course, students will be familiar with the basic principles of statistical signal processing. They will understand how to apply statistical signal processing techniques to relevant fields in electrical engineering (such as communications). Students will develop confidence in their ability to solve mathematical problems of analysis and design. They will be able to apply the principles they have learnt in this course to other areas.

<table>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)
<table>
<tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr>
<tr><td>a)</td><td>Written or Oral Examination or Presentation</td><td>120-180 min or 30-45 min or 30 min</td><td>100%</td></tr>
</table></td></tr>
<tr><td>7</td><td>Study Achievement:
none</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
None</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination (MAP) was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Informatik v3, Master’s Program Electrical Systems Engineering v2 (ESEMA v2), Master’s Program Electrical Systems Engineering v3 (ESEMA v3)</td></tr>
<tr><td>12</td><td>Module coordinator:
Prof. Dr. Peter Schreier</td></tr>
<tr><td>13</td><td>Other Notes:
Remarks of course Statistical Signal Processing:
Course Homepage
<code>http://sst.upb.de/teaching</code>
Implementation
Lectures and tutorials
Teaching Material, Literature
Literature references are given in the first lecture.</td></tr>
</table>

Topics in Signal Processing

Topics in Signal Processing

Module number:	Workload (h):	Credits:	Regular Cycle:
M.048.92014	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1.-3. Semester	1	en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.048.92014 Topics in Signal Processing	2L 2Ex, WS	60	120	C	30/30

2 **Options within the module:**

None

3 **Admission requirements:**

None

Prerequisites of course Topics in Signal Processing:
Recommended: Signal and system theory, at least a basic understanding of probability and linear algebra

4 **Contents:**

Contents of the course Topics in Signal Processing:
Short Description
This course covers a selection of current topics in signal processing. One part of this course will follow a regular lecture format, while the other part will require active student participation.
Contents
This course will first review relevant aspects of linear algebra and probability theory. Then students will learn how to read, analyze, and present recent papers from the signal processing literature.

5 **Learning outcomes and competences:**

In this course, students will familiarize themselves with some current research topics in signal processing. They will learn to read and understand scientific publications and to critically evaluate results. Students will develop confidence in their ability to solve mathematical problems of analysis and design. They will be able to apply the principles they have learnt in this course to other areas.

<table>
<tr><td>6</td><td>Assessments:
☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)
<table>
<tr><th>zu</th><th>Type of examination</th><th>Duration or scope</th><th>Weighting for the module grade</th></tr>
<tr><td>a)</td><td>Written or Oral Examination or Presentation</td><td>120-180 min or 30-45 min or 30 min</td><td>100%</td></tr>
</table></td></tr>
<tr><td>7</td><td>Study Achievement:
none</td></tr>
<tr><td>8</td><td>Prerequisites for participation in examinations:
None</td></tr>
<tr><td>9</td><td>Prerequisites for assigning credits:
The credit points are awarded after the module examination (MAP) was passed.</td></tr>
<tr><td>10</td><td>Weighing for overall grade:
The module is weighted according to the number of credits (factor 1).</td></tr>
<tr><td>11</td><td>Reuse in degree courses or degree course versions :
Masterstudiengang Computer Engineering v3 (CEMA v3), Masterstudiengang Computer Engineering v3 (CEMA v3), englisch, Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4, Masterstudiengang Robotics and Artificial Intelligence Engineering v1 (RAIMA v1), Master’s Program Electrical Systems Engineering v2 (ESEMA v2), Master’s Program Electrical Systems Engineering v3 (ESEMA v3)</td></tr>
<tr><td>12</td><td>Module coordinator:
Prof. Dr. Peter Schreier</td></tr>
<tr><td>13</td><td>Other Notes:
Remarks of course Topics in Signal Processing:
Course Homepage
http://sst.uni-paderborn.de/teaching/courses/
Implementation
Lectures and tutorials with active student participation, student presentations
Teaching Material, Literature
References will be given in the first lecture.</td></tr>
</table>

Unsupervised Learning and Evolutionary Optimisation Using R

Unsupervised Learning and Evolutionary Optimisation Using R

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4093	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05829 Unsupervised Learning and Evolutionary Optimisation Using R	L3 Ex2	75	105	CE	70

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Unsupervised Learning and Evolutionary Optimisation Using R:
Recommended Proficiencies

- Basic knowledge and interest in mathematics, statistics and probability theory
- Basic knowledge of programming

4 **Contents:**

Contents of the course Unsupervised Learning and Evolutionary Optimisation Using R:
The course includes the formal and applied concepts of unsupervised machine learning and its implementation in the statistical programming language R.
In particular, the following topics are covered in a theoretical and applied manner:

- Introduction to the statistical programming language R
- Data pre-processing and quality aspects of data
- (Stream) clustering techniques
- Dimensionality reduction techniques
- Basic principles of evolutionary optimisation, both single- and multi-objective
- Practical application of the methods using R in individual and group work

5 **Learning outcomes and competences:**

After completing the module, students will be able to . . .

- properly assess data quality and select suitable techniques for data pre-processing
- explain and apply core methods of unsupervised learning
- understand the basic principles of evolutionary optimisation methods
- competently apply techniques to assess the quality of optimisation procedures
- use the statistical software R for statistical data analysis, unsupervised learning and evolutionary optimisation in a competent manner
- analyse problems in a team and present practice-relevant solutions

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Assignments		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Heike Trautmann

13	**Other Notes:** *Remarks of course Unsupervised Learning and Evolutionary Optimisation Using R:* **Implementation Method** An introduction to the statistical programming language R is given compactly in the first weeks of the course. Methods of unsupervised machine learning are covered within lecture presentations interleaved with interactive exercises. Methods understanding will be further deepened in tutorials focusing both on theory as well as application-oriented tasks using R. **Learning Material, Literature** Recommended for the statistical programming language R: • Hadley Wickham & Garrett Grolemund (2023). R for Data Science: Import, Tidy, Transform, Visualize, and Model Data. 2nd ed. O'Reilly • Torsten Hothorn and Brian S. Everitt (2014). A Handbook of Statistical Analyses Using R. Chapman & Hall/CRC Press, 3rd edition, 2014. • C. Heumann, M. Schomaker, and Shalabh. Introduction to Statistics and Data Analysis With Exercises, Solutions and Applications in R. Springer, 2017. The methods sections are based on a variety of references which will be announced in the lecture.

Usable Security and Privacy			
Usable Security and Privacy			
Module number: M.079.4086	**Workload (h):** 180	**Credits:** 6	**Regular Cycle:** module.frequency.en
	Semester number: null	**Duration (in sem.):** 1	**Teaching Language:** en

1 **Module structure:**

	Course	form of teachin	contact-time (h)	self-study (h)	status (C/CE)	group size (TN)
a)	L.079.05804 Usable Security and Privacy	L2 Ex3	75	105	CE	40

2 **Options within the module:**
none

3 **Admission requirements:**
keine

4	**Contents:** *Contents of the course Usable Security and Privacy:* Human factors and usability issues have traditionally played a limited role in security research and secure systems development. Usability issues have been largely disregarded by security experts due to their failure to acknowledge their significance and their insufficient knowledge to tackle them. Today there is consensus on the importance of understanding users behavior and improving usability to achieve true security. This course provides practical and research-oriented knowledge about usable security and privacy. Students will gain practical experience through focused presence exercises and work in small teams to conduct a semester-wide research project with the goal of designing and pretesting a user study on human-centered security and privacy. For that, the course will present research methods and give an introduction into HCI and usability concepts. The course will also address foundational and state-of-the-art research topics in the area, such as privacy and transparency enhancing tools, usable authentication, and developer-centered security. By reviewing relevant papers and giving presentations, the students will get familiar with the latest research in the field and gain knowledge about how to work scientifically. The course includes the following contents: • Security and privacy concepts • Foundations of cryptography • Privacy and transparency enhacing tools • HCI and usability research methods • Ethics in technology • Quantitative and qualitative data analysis • Usable authentication • Usable privacy • Developer-centered security
5	**Learning outcomes and competences:** Students will • gain an appreciation for the importance of usable security and privacy • learn about the history of the field and main research areas and challenges • are able to apply methodologies to conduct user research in security and privacy • get familiar with the latest research in the field **Non-cognitive Skills** • Literacy (scientific) • Self-monitoring • Team work

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	Type of examination	Duration or scope	Weighting for the module grade
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	Type of achievement	Duration or Scope	SL / QT
a)	Practical work with written report and discussion		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr. Patricia Arias Cabarcos

13 **Other Notes:**

Remarks of course Usable Security and Privacy:
Implementation method
Basic concepts are presented in a lecture style format. By engaging in presence exercises and conducting a research project in small groups focused on a user-study for usable security and privacy research throughout the semester, students can acquire more profound theoretical and practical knowledge.
Learning Material, Literature

- Lazar, J., Feng, J.H. and Hochheiser, H., 2017. Research methods in human-computer interaction. Morgan Kaufmann.
- Redmiles, E.M., Acar, Y., Fahl, S. and Mazurek, M.L., 2017. A summary of survey methodology best practices for security and privacy researchers.
- Slides and scientific literature references will be given during the course.

VLSI-Testing

VLSI-Testing

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.048.92027	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	1.-3. Semester	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.048.92027 VLSI Testing	2L 2Ex, WS	60	120	C	30/30

2 **Options within the module:**

None

3 **Admission requirements:**

None

Prerequisites of course VLSI Testing:
Recommended: Digital Design

4 **Contents:**

Contents of the course VLSI Testing:
Short Description
The course “VLSI Testing” focuses on techniques for detecting hardware defects in micro-electronic circuits. Algorithms for test data generation and test response evaluation as well as hardware structures for design for test (DFT) and on-chip test implementation (BIST) are presented.
Contents
In detail the following topics are covered:

- Fault models
- Testability measures and design for test (DFT)
- Logic and fault simulation
- Automatic test pattern generation (ATPG)
- Built-in self-test (BIST), in particular test data compression and test response compaction
- Memory test

5 **Learning outcomes and competences:**

Domain competence:
After attending the course, the students will be able

- to describe fault models, DFT techniques, and test tools,
- to explain and apply the underlying models and algorithms for fault simulation and test generation,
- to analyze systems with respect to their testability and to derive appropriate test strategies.

Key qualifications:
The students

- are able to apply the practiced strategies for problem solving across varying disciplines,
- have experience in presenting their solutions to their fellow students, and
- know how to improve their competences by private study.

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or Oral Examination or Presentation	120-180 min or 30-45 min or 30 min	100%

7 **Study Achievement:**

none

8 **Prerequisites for participation in examinations:**

None

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination (MAP) was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Computer Engineering v3 (CEMA v3), Masterstudiengang Computer Engineering v3 (CEMA v3), englisch, Masterstudiengang Computer Engineering v4 (CEMA v4), Masterstudiengang Computer Engineering v4 (CEMA v4), englisch, Masterstudiengang Informatik v3, Masterstudiengang Informatik v4, Masterstudiengang Robotics and Artificial Intelligence Engineering v1 (RAIMA v1), Master's Program Electrical Systems Engineering v2 (ESEMA v2), Master's Program Electrical Systems Engineering v3 (ESEMA v3)

12 **Module coordinator:**

Prof. Dr. Sybille Hellebrand

13 **Other Notes:**

Remarks of course VLSI Testing:

ATTENTION - IMPORTANT NOTICE
The course doesn't take place in winter term 2026/27 summer term 2026. Please see the notice boards of the group.

Course Homepage
`https://ei.uni-paderborn.de/en/electrical-engineering/date/teaching/electrical-engineering/overview`
Implementation

- Lecture based on slide presentation, extensions on blackboard
- Exercises in small groups based on exercise sheets with students presenting their own solutions
- Hands-on exercises using various software tools

Teaching Material, Literature
Additional material can be found in panda

- Michael L. Bushnell, Vishwani D. Agrawal, „Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits,“Boston, Dordrecht, London: Kluwer Academic Publishers, 2000
- Laung-Terng Wang, Cheng-Wen Wu, Xiaoqing Wen, „VLSI Test Principles and Architectures: Design for Testability,“ Morgan Kaufmann Series in Systems on Silicon, ISBN: 0123705975

Web Security

Web Security

Module number:	**Workload (h):**	**Credits:**	**Regular Cycle:**
M.079.4073	180	6	module.frequency.en
	Semester number:	**Duration (in sem.):**	**Teaching Language:**
	null	1	en

1 **Module structure:**

	Course	**form of teachin**	**contact-time (h)**	**self-study (h)**	**status (C/CE)**	**group size (TN)**
a)	L.079.05820 Web Security	L3 Ex2	75	105	C	40

2 **Options within the module:**

none

3 **Admission requirements:**

Prerequisites of course Web Security:
Recommended Proficiencies
Knowledge in programming, IT security and basic knowledge in cryptography

4 **Contents:**

Contents of the course Web Security:
Modern web applications and web services usually consist of multiple layers. They are based on different (often complex) technologies that are constantly being developed. Their complexity is often the reason for new types of attacks that can be observed on the web every day.
In this lecture, we will focus on the most important technologies and learn what you have to consider while securing your web applications. We will introduce prominent and widespread attacks and show how to prevent them. These range from typical attacks from the OWASP Top 10 list, such as XSS or SQL Injection, to attacks on web services and Single Sign-On standards (e.g., on SAML and OpenID Connect). Based on many cases, we will learn what is important in the design and implementation of secure web applications.

5 **Learning outcomes and competences:**

After successful completion, students have a comprehensive understanding of the technical aspects of web applications, web services, and various authentication mechanisms. They have learned that the web technologies used today are complex and that their complexity poses many security problems. Students have an overview of current web attacks and know how to prevent them practically.
Non-cognitive Skills

- Team work
- Literacy (scientific)

6 **Assessments:**

☒Final module exam (MAP) □Module exam (MP) □Partial module exams (MTP)

zu	**Type of examination**	**Duration or scope**	**Weighting for the module grade**
a)	Written or oral examination	90-120 min or 40 min	100%

The responsible lecturer announces type and duration of assessment modalities in the first three weeks of the lecture period at latest.

7 **Study Achievement:**

zu	**Type of achievement**	**Duration or Scope**	**SL / QT**
a)	Written exercises		CA

Within the first three weeks of the lecture period each respective lecturer will specify the manner in which the course achievement will be conducted.

8 **Prerequisites for participation in examinations:**

Passing of course achievement

9 **Prerequisites for assigning credits:**

The credit points are awarded after the module examination was passed.

10 **Weighing for overall grade:**

The module is weighted according to the number of credits (factor 1).

11 **Reuse in degree courses or degree course versions :**

Masterstudiengang Informatik v3

12 **Module coordinator:**

Prof. Dr.-Ing. Juraj Somorovsky

13 **Other Notes:**

Remarks of course Web Security:
Implementation method
Lecture with exercises
Learning Material, Literature

- Lecture slides
- Scientific papers

4 Focus Areas

4.1 Algorithm Design

Coordination

Prof. Dr. rer. nat. Johannes Blömer

Included Modules

- Advanced Algorithms
- Advanced Complexity Theory
- Advanced Distributed Algorithms and Data Structures
- Algorithms for Complex Virtual Scenes
- Foundations of Cryptography
- Geometry Processing
- Introduction to Quantum Computation
- Post-Quantum Cryptography
- Quantum Algorithms
- Quantum Complexity Theory
- Quantum Information

Description

In this focus area students can concentrate on studying the

- important techniques for the design of efficient algorithms
- application areas for the design of efficient algorithms, i.e. computer graphics, networks, big data, ...
- limits for the design of efficient algorithms, i.e. complexity theory
- constructive use of complexity theory in cryptography and security
- connection between efficient algorithms and verification and software design

4.2 Computer Systems

Coordination

Prof. Dr. Marco Platzner

Included Modules

- Advanced Computer Architecture
- Advanced Networked Systems
- Approximate Computing
- Human Factors in Security and Privacy
- Reconfigurable Computing
- Resource-Efficient AI: Architectures and Methodologies
- Usable Security and Privacy
- VLSI Testing

Description

The focus area “Computer systems” goes into technical depths of various aspects of modern computer systems. Main topics are the analysis and evaluation of computer architectures, systematic methods for design and optimisation of computer systems, in particular the interplay of hardware and software, and programming models and methods for parallel and specialised computer architectures, which are increasingly gaining importance.

4.3 Data Science

Coordination

Prof. Dr. Axel-Cyrille Ngonga Ngomo

Included Modules

- Advanced Algorithms
- Advanced Distributed Algorithms and Data Structures
- Data Science for Software Engineering
- Digitale Sprachsignalverarbeitung
- Explainable Artificial Intelligence
- Foundations of Knowledge Graphs
- Introduction to Description Logics
- Topics in Pattern Recognition and Machine Learning
- Topics in Signal Processing
- Unsupervised Learning and Evolutionary Optimisation Using R

Description

Data science is a young scientific discipline in the intersection of computer science, statistics, mathematics, and engineering, which has quickly developed into one of the most impactful areas in the current

research landscape. It is a main driving factor of the digitalization and “datafication” of a large portion of our society, including companies, research organizations, and even private homes and people. In science and research, it is often viewed as a “fourth paradigm”, next to the empirical, theoretical, and computational approach. Broadly speaking, the major goal of data science is to develop methodological and algorithmic foundations as well as computer systems for automating the extraction of useful knowledge and insight from data.

The focus area “data science” will provide the students with solid theoretical foundations as well as practical skills that constitute the profile of a modern data scientist. To this end, courses will be offered in three main directions: Mathematical and algorithmic foundations, data analytics, software and systems. Here, the students will learn how to acquire, archive, compress, and aggregate large amounts of heterogeneous data (text, image, audio and video, etc.), and how to analyze such data using methods from statistics, machine learning, and data mining. Moreover, they will be familiarized with relevant programming languages, software engineering techniques, and scalable information processing architectures. Finally, the students will broaden their practical experience and develop soft skills by specializing in application areas such as Industrial Data Science, Digital Humanities, Business Analytics and Cybersecurity.

4.4 Intelligence and Data

Coordination

Prof. Dr. Axel-Cyrille Ngonga Ngomo

Included Modules

- Deep Reinforcement Learning
- Explainable Artificial Intelligence
- Foundations of Knowledge Graphs
- Introduction to Description Logics
- Logic Programming for Artificial Intelligence
- Multi-Objective Optimisation
- Statistical Signal Processing
- Unsupervised Learning and Evolutionary Optimisation Using R

Description

Intelligent systems are computer systems the behavior of which is controlled by methods and algorithms from artificial intelligence (AI). Systems of that kind are becoming increasingly important, not only on a scientific level but also in a social context: Autonomous or semi-autonomous systems such as service robots, self-driving cars or medical diagnosis systems will have a deep impact on our future private and professional life. In addition to methodological advances and improved hardware, the “data explosion” can be seen as a main driving factor for the rapid development of AI-systems during the last decade:

Thanks to the availability of massive amounts of data or sensory feedback from their environment, intelligent systems are able to automatically improve their behavior through adaptation and learning.

This focus area covers important aspects of intelligent systems design and conveys corresponding theoretical and methodological foundations. This includes lectures on machine learning and data analysis, data management, computer graphical and visual data analysis, as well as swarm intelligence and robotics.

4.5 Networks and Communication

Coordination

Prof. Dr. Marco Platzner

Included Modules

- Advanced Distributed Algorithms and Data Structures
- Advanced Networked Systems
- Real-Time Operating Systems: Designs and Implementations
- Web Security

Description

The focus area “Networks and Communication” teaches architectures, methods and systems of modern communication technology. To this end, we investigate methods of various abstractions levels, starting from the lowest level physical transmissions up to and including application design in distributed environments. Different types of systems are considered, ranging from conventional mobile communication over ad hoc networks and vehicular communication systems to networking in data centers and architectures for the future Internet at large. In doing so, we strive to build the bridge to aspects of distributed systems design. Questions on architecture design and options for protocol designs are complemented by the evaluation of such systems. To answer those questions, we introduce experimental and statistical performance evaluation techniques.

4.6 Security

Coordination

Prof. Dr. Yasemin Acar

Included Modules

- Advanced Distributed Algorithms and Data Structures
- Designing code analyses for large-scale software systems 1
- Designing code analyses for large-scale software systems 2
- Foundations of Cryptography
- Hardware Security
- Human Factors in Security and Privacy
- Introduction to Quantum Computation
- Post-Quantum Cryptography
- Privacy and Technology
- Quantum Complexity Theory
- Quantum Information
- Real World Crypto Engineering
- Usable Security and Privacy
- Web Security

Description

In all areas of life, digital technologies, such as the (Industrial) Internet of Things, Cyber-Physical Systems, Digital Automotives, Digital Health or Industry 4.0, offer immense innovation potential. However, increasing digitization requires new approaches to safely exploit this potential. In order to be able to tackle this challenge, there is a great need in industry, research and teaching for well-trained computer science experts with in-depth knowledge of IT security. In the specialization area “IT Security”, solid theoretical basic knowledge is taught in combination with practical skills. The course covers technical skills from the field of IT security (e.g. software security, formal verification, basics of modern cryptography and communication security), in which typical security vulnerabilities and attack techniques are presented and countermeasures and their effectiveness are examined.

Since security cannot be seen independently of concrete applications and different applications have different security requirements, specialized competencies in modern application fields with special security requirements (e.g., communication protocols in the mobile and automotive fields) as well as supplementary qualifications in the areas of algorithms and quantum computing are also covered.

4.7 Software Engineering

Coordination

Prof. Dr. Eric Bodden

Included Modules

- Data-Driven Innovation
- Data Science for Software Engineering
- Data Science in Industrial Applications
- Designing code analyses for large-scale software systems 1

- Designing code analyses for large-scale software systems 2
- Human Factors in Security and Privacy
- Model-Based Systems Engineering
- Software Architecture Design and Recovery

Description

In this focus area, students can concentrate on studying concepts, languages, methods, techniques, and tools for the systematic development of software systems. These comprise

- constructive techniques for developing functional and non-functional aspects of a system,
- formal and informal analytical techniques to ensure high quality of a system,
- systematic techniques to enable situation-specific process models

5 Modules in Winter Semester

- M.079.4005 Advanced Computer Architecture 24
- M.079.4006 Advanced Distributed Algorithms and Data Structures 27
- M.079.4212 Agentic Software Engineering 32
- M.079.4070 Designing code analyses for large-scale software systems 1 48
- M.079.4054 Foundations of Knowledge Graphs 69
- Studium Generale – Master 16
- M.079.4209 Geometric Deep Learning 72
- M.079.4092 Human Factors in Security and Privacy 81
- M.079.4059 Introduction to Quantum Computation 87
- M.079.4031 Logic Programming for Artificial Intelligence (MA v3) 90
- M.079.4032 Machine Learning 1 93
- Master-Abschlussarbeit 4
- M.079.4062 Model-Based Systems Engineering 96
- M.079.4087 Privacy and Technology 105
- M.079.4041 Projektgruppe 7
- M.079.4090 Quantum Information 114
- M.079.4067 Real World Crypto Engineering 120
- M.079.4213 Real-Time Operating Systems: Design and Implementations 117
- M.079.4043 Reconfigurable Computing 122
- M.048.55106 Resource-Efficient AI: Architectures and Methodologies 125
- M.079.4045 Seminar I 10
- M.079.4046 Seminar II 13
- M.079.4094 Software Architecture Design and Recovery 128
- M.048.92004 Statistical Signal Processing 134
- M.048.92014 Topics in Signal Processing 136
- M.048.92027 VLSI-Testing 144

6 Modules in Summer Semester

- M.079.4002 Advanced Algorithms 18
- M.079.4004 Advanced Complexity Theory (v3) 21
- M.079.4096 Advanced Networked Systems 29
- M.079.4009 Algorithms for Complex Virtual Scenes 36
- M.079.4068 Approximate Computing 40
- M.079.4101 Data Science for Software Engineering 45
- M.079.4076 Data-Driven Innovation 43
- M.079.4071 Designing code analyses for large-scale software systems 2 52
- M.048.24001 Digitale Sprachsignalverarbeitung 56
- M.079.4091 Explainable Artificial Intelligence 60
- M.079.4210 Foundational Methods for Knowledge Representation and Reasoning 63
- M.079.4020 Foundations of Cryptography 66
- Studium Generale – Master 16
- M.079.4205 Geometry Processing 75
- M.079.4211 Hardware Security 78
- M.079.4098 Introduction to Description Logics 84
- M.079.4031 Logic Programming for Artificial Intelligence (MA v3) 90
- Master-Abschlussarbeit 4
- M.079.4095 Multi-Objective Optimisation 99
- M.079.4089 Post-Quantum Cryptography 102
- M.079.4041 Projektgruppe 7
- M.079.4072 Quantum Algorithms 108
- M.079.4063 Quantum Complexity Theory 111
- M.079.4045 Seminar I 10
- M.079.4046 Seminar II 13
- M.079.4048 Software Quality Assurance 131
- M.079.4093 Unsupervised Learning and Evolutionary Optimisation Using R 138
- M.079.4086 Usable Security and Privacy 141
- M.079.4073 Web Security 147

7 Modules in English

- M.079.4002 Advanced Algorithms 18
- M.079.4004 Advanced Complexity Theory (v3) 21
- M.079.4005 Advanced Computer Architecture 24
- M.079.4006 Advanced Distributed Algorithms and Data Structures 27
- M.079.4096 Advanced Networked Systems 29
- M.079.4212 Agentic Software Engineering 32
- M.079.4009 Algorithms for Complex Virtual Scenes 36
- M.079.4068 Approximate Computing 40
- M.079.4101 Data Science for Software Engineering 45
- M.079.4076 Data-Driven Innovation 43
- M.079.4070 Designing code analyses for large-scale software systems 1 48
- M.079.4071 Designing code analyses for large-scale software systems 2 52
- M.048.24001 Digital Speech Signal Processing 56
- M.079.4091 Explainable Artificial Intelligence 60
- M.079.4210 Foundational Methods for Knowledge Representation and Reasoning 63
- M.079.4020 Foundations of Cryptography 66
- M.079.4054 Foundations of Knowledge Graphs 69
- General Studies – Master 16
- M.079.4209 Geometric Deep Learning 72
- M.079.4205 Geometry Processing 75
- M.079.4211 Hardware Security 78
- M.079.4092 Human Factors in Security and Privacy 81
- M.079.4098 Introduction to Description Logics 84
- M.079.4059 Introduction to Quantum Computation 87
- M.079.4031 Logic Programming for Artificial Intelligence 90
- M.079.4032 Machine Learning 1 93
- Master Thesis 4
- M.079.4095 Multi-Objective Optimisation 99
- M.079.4089 Post-Quantum Cryptography 102
- M.079.4087 Privacy and Technology 105
- M.079.4041 Project Group 7
- M.079.4072 Quantum Algorithms 108
- M.079.4063 Quantum Complexity Theory 111
- M.079.4090 Quantum Information 114
- M.079.4067 Real World Crypto Engineering 120
- M.079.4213 Real-Time Operating Systems: Design and Implementations 117
- M.079.4043 Reconfigurable Computing 122
- M.048.55106 Resource-Efficient AI: Architectures and Methodologies 125
- M.079.4045 Seminar I 10
- M.079.4046 Seminar II 13

- M.079.4094 Software Architecture Design and Recovery 128
- M.079.4048 Software Quality Assurance 131
- M.048.92004 Statistical Signal Processing 134
- M.048.92030 Topics in Pattern Recognition and Machine Learning ..... **??**
- M.048.92014 Topics in Signal Processing 136
- M.079.4093 Unsupervised Learning and Evolutionary Optimisation Using R 138
- M.079.4086 Usable Security and Privacy 141
- M.048.92027 VLSI-Testing 144
- M.079.4073 Web Security 147

Erzeugt am 30. September 2026 um 15:07.