Maastricht University

Faculty of Science and Engineering

Department of Advanced Computing Sciences (DACS)

Contents

Academic Calendar 2023 - 2024

Bachelor's Programme Data Science and Artificial Intelligence (year 1, 2, 3)
Bachelor's Programme Computer Science (year 1) and
Master's Programmes Artificial Intelligence and Data Science for Decision Making (year 1, 2)

Inkom Maastricht University
21-25 August 2023

Education periods
Period 1: 4 September - 20 October 2023
Period 2: 30 October - 15 December 2023
Period 3: 8 January - 26 January 2024
Period 4: 5 February - 28 March 2024
Period 5: 8 April - 31 May 2024
Period 6: 10 June - 28 June 2024

Project weeks
Period 3: 8 January - 26 January 2024
BA year 1: Final presentation: January 2024
BA year 2: Final presentation: January 2024
BA year 3: Final presentation: January 2024
MA year 1: Project seminar: January 2024

Period 6: 10 June - 28 June 2024
BA year 1: Final presentation: June 2024
BA year 2: Final presentation: June 2024
MA year 1: Project seminar: June 2024

BA Thesis Winter Conference: December 2023
Resit BA Thesis Winter Conference: January 2024
BA Thesis Summer Conference: June 2024
Resit BA Thesis Summer Conference: August 2024

Introduction days
29 August 2023 (Bachelor DSAI)
30 August 2023 (Bachelor CS)
31 August 2023 (Masters, premasters, exchange)
31 August 2023 (BBQ event for all new students)

Exam and Resit periods
Period 1 - Exams: 23 - 27 October 2023
Period 2 - Exams: 18 - 22 December 2023
Period 3 - Resits semester 1:
29 January - 2 February 2024
Period 4 - Exams: 2 - 5 April 2024
Period 5 - Exams: 3 - 7 June 2024*
Period 6 - Resits semester 2: 1 - 5 July 2024*
** Please note: resits of BAY3 period 4 take place in the exam week of period 5.*

Graduation
t.b.d. (Bachelor)
t.b.d. (Masters)

(Public) Holiday, no courses
Christmas: 25 December 2023 - 5 January 2024
Carnival break: 12 - 16 February 2024
Good Friday: 29 March 2024
Easter Monday: 1 April 2024
Ascension Day and Bridging Day:
9 - 10 May 2024
Whit Monday: 20 May 2024

DACS students will have to register for their own courses and resits through the Student Portal.
Note: exams can also be scheduled in the evening, from 18h00 until 21h00.

August						
M	T	W	T	F	S	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

September						
M	T	W	T	F	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

October						
M	T	W	T	F	S	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

November						
M	T	W	T	F	S	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

December						
M	T	W	T	F	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

January						
M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

February						
M	T	W	T	F	S	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29			

March						
M	T	W	T	F	S	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

April						
M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

May						
M	T	W	T	F	S	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

June						
M	T	W	T	F	S	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

July						
M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	23	24	25	26	27
29	30	31				

1 Education: The profile of the study Data Science, Artificial Intelligence, and Decision Making

1.1 What are the programmes about?

The Bachelor's Programmes Computer Science and Data Science & Artificial Intelligence, and the Master's Programmes Artificial Intelligence and Data Science for Decision Making of the Faculty of Science and Engineering at Maastricht University are embedded within the Department of Advanced Computing Sciences.

At the Department of Advanced Computing Sciences (DACS), research and education are tightly linked. Our academic staff consists of experienced lecturers and researchers that are well known in the international scientific community. You will be taught by lecturers from DACS, and you will become part of a tight-knit community consisting of approximately 900 bachelor's and master's students and 100 staff members. Together, we come from over 50 different countries. The department has its own dedicated study association, MSV Incognito, of which all students automatically become a member.

The department's research activities span the disciplines and interfaces of artificial intelligence, data science, computer science, applied mathematics, and robotics, covering the entire spectrum from curiosity-driven research to responsible societal applications, often taking on an interdisciplinary character. Contributions to areas such as multi-agent systems, (medical) signal and image processing, machine learning, explainable AI, FAIR principles, game theory, intelligent search techniques, computer vision, and affective computing are internationally recognized. You will be exposed to the department's research through several semester projects, and your thesis project, among all

1.2 Study System

1.2.1 Bachelor Data Science & Artificial Intelligence

The bachelor's programme in Data Science and Artificial Intelligence is a three-year programme. We chose for a broad setup of the curriculum, so that students can decide on the way they would like to specialize during the final stage.

Year 1

Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Procedural Programming;	Objects in Programming;	PROJECT	Data Structures and Algorithms;	Computational and Cognitive Neuroscience;	PROJECT
Discrete Mathematics;	Calculus;		Linear Algebra;	Numerical Methods;	
Introduction to Data Science and Artificial Intelligence	Logic		Principles of Data Science	Software Engineering	
PROJECT			PROJECT		

Year 2

Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Databases;	Machine Learning;	PROJECT	Human Computer Interaction and Affective Computing Mathematical Modelling;	Philosophy and Artificial Intelligence;	PROJECT
Probability and Statistics;	Simulation and Statistical Analysis;		1 out of the electives: * Theoretical Computer Science * Introduction to Image and Video Processing	Linear Programming;	
Graph Theory	Reasoning Techniques			Natural Language Processing	
PROJECT			PROJECT		

Year 3

Period 1*	Period 2*	Period 3	Period 4	Period 5	Period 6
Semantic Web;	Large Scale IT and Cloud Computing;	PROJECT	Data Analysis;	BACHELOR'S THESIS	
Game Theory;	Computer Security;		Operations Research Case Studies;		
Computer Security;	Logic for AI;		Intelligent Systems		
Robotics and Embedded Systems;	Parallel Programming;				
Digital Society;	Introduction to Bio-Informatics;				
Recommender Systems;	Software and Systems Verification;				
	Quantum C omputing				
PROJECT			BACHELOR'S THESIS		

** Third year students choose 6 optional courses (3 in period 3.1 and 3 in period 3.2) in addition to the semester project in semester 1 of year 3. In case students have passed both electives of period 2.4, either the course Theoretical Computer Science or Introduction to Image & Video Processing can replace 1 of the third year electives. In semester 1 of year 3, student can also choose (1) elective courses at other UM bachelor programmes of at most 18 ECTS (2) the minor Entrepreneurship or (3) the educational minor. (4) Alternatively, students can study abroad for a semester at one of our exchange partners (see internationalization section below). Please contact the student adviser for more information. Also, check the Study Abroad section in the "My Organisations" section of the Canvas.*

Periods 1 and 2 last eight weeks in total. During week 1-7 there are classes and in week 8 exams. Periods 4 and 5 will last 9 weeks due to several holidays in this period. Three courses are offered during each period, each course is good for 4 credits (ECTS). Per course, five to seven hours of class are offered each week in year 1, and about five hours in year 2 and 3.

Next to these courses, you participate in a group project of 6 credits that will last the whole semester. Skill trainings and project meetings are mandatory: you are expected to be present during 100% of the skill classes and 100% of the project meetings in each academic year. For more details about attendance of skill classes and project meetings, please check the Education and Examination Regulations (EER) and the Rules and Regulations (R&R) that are published on the Student Portal.

If a student does not contribute sufficiently to the project, the project examiners may deviate from the group grade for this individual student. The project of semester 1 runs during period 1, 2 and 3. The project of semester 2 runs during period 4, 5 and 6. For specific details on the project curriculum in year 1 see section Project 1-1 and Project 1-2 with the course descriptions of year 1.

Periods 3 and 6 last four weeks, and during the first three weeks, students work fulltime to finish their project assignment. After the project weeks, it is possible to resit previous periods during the fourth week. During the three-week project periods in periods 3 and 6, you will work full-time on a project assignment. This project assignment is announced in the beginning of periods 1 or 4, along with the group composition. At the end of week 7 of the 8-weeks periods, each group separately gives a brief presentation for the examiners of the project. Each group shows their interpretation of the problem and discusses their approach and schedule for the following phase. The examiners give their feedback to the content and progress of a project. These interim presentations will be assessed. The project is concluded at the end of the project weeks with the handing in of the final report, handing in of the product, and with an oral presentation of the project's results for all groups. The assessment of the report, the presentation, and the product in principle result in the same mark for all the group members. However, there can be diversification, see the Education and Examination Regulations (EER) and the Rules and Regulations (R&R).

The final stage of your bachelor's programme, period 5 and 6 of year 3 is reserved for writing your bachelor's thesis that equals 18 ECTS. Every student has to conduct a short scientific research focussed on a relevant topic. This can be empirical or theoretical research. Students have acquired information on these different research domains throughout their educational programme. Each student has to hand in a signed bachelor thesis project plan to the Bachelor's thesis coordinator. Each student is supervised by a thesis supervisor. In the second period of this semester, the students conduct their own research. In the end of the last period of the semester, each student must present their results.

1.2.2 Bachelor Computer Science

The bachelor's programme in Computer Science is a three-year programme. The programme is designed to provide a solid background in fundamental computer science, software development, and mathematics.

Year 1

Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Introduction to Computer Science;	Objects in Programming;	PROJECT	Data Structures and Algorithms;	Databases;	PROJECT
Procedural Programming;	Calculus;		Linear Algebra;	Statistics;	
Discrete Mathematics;	Logic		Object Oriented Modelling	Algorithmic Design	
PROJECT			PROJECT		

Periods 1 and 2 last eight weeks in total. During week 1-7 there are classes and in week 8 exams. Periods 4 and 5 will last 9 weeks due to several holidays in this period. Three courses are offered during each period, each course is good for 4 credits (ECTS). Per course, five to seven hours of class are offered each week in year 1.

Next to these courses, you participate in a group project of 6 credits that will last the whole semester. Skill trainings and project meetings are mandatory: you are expected to be present during 100% of the skill classes and 100% of the project meetings in each academic year. For more details about attendance of skill classes and project meetings, please check the Education and Examination Regulations (EER) and the Rules and Regulations (R&R) that are published on the Student Portal.

If a student does not contribute sufficiently to the project, the project examiners may deviate from the group grade for this individual student. The project of semester 1 runs during period 1, 2 and 3. The project of semester 2 runs during period 4, 5 and 6. For specific details on the project curriculum in year 1 see section Project 1-1 and Project 1-2 with the course descriptions of year 1.

Periods 3 and 6 last four weeks, and during the first three weeks, students work fulltime to finish their project assignment. After the project weeks, it is possible to resit previous periods during the fourth week. During the three-week project periods in periods 3 and 6, you will work full-time on a project assignment. This project assignment is announced in the beginning of periods 1 or 4, along with the group composition. At the end of week 7 of the 8-weeks periods, each group separately gives a brief presentation for the examiners of the project. Each group shows their interpretation of the problem and discusses their approach and schedule for the following phase. The examiners give their feedback to the content and progress of a project. These interim presentations will be assessed. The project is concluded at the end of the project weeks with the handing in of the final report, handing in of the product, and with an oral presentation of the project's results for all groups. The assessment of the report, the presentation, and the product in principle result in the same mark for all the group members. However, there can be diversification, see the Education and Examination Regulations (EER) and the Rules and Regulations (R&R).

The final stage of your bachelor's programme, period 5 and 6 of year 3 is reserved for writing your bachelor's thesis that equals 18 ECTS. Every student has to conduct a short scientific research focussed on a relevant topic. This can be empirical or theoretical research. Students have acquired information on these different research domains throughout their educational programme. Each student has to hand in a signed bachelor thesis project plan to the Bachelor's thesis coordinator. Each student is supervised by a thesis supervisor. In the second period of this semester, the students conduct their own research. In the end of the last period of the semester, each student must present their results.

1.2.3 Masters

Master Data Science for Decision Making

Year 1

Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Data Mining:	**Model Identification and Data Fitting:**		**Computational Statistics**	**Algorithms for Big Data**	
1 of the electives:	*1 of the electives:*		*1 of the electives:*	*1 of the electives:*	
Signal and Image Processing;	Advanced Concepts in Machine Learning;	PROJECT	Dynamic Game Theory;	Symbolic Computation and Control;	PROJECT
Mathematical optimization;	Deep Learning for Image and Video Processing;		Planning and Scheduling;	Information Retrieval and Text Mining;	
Stochastic Decision-making	Advanced Natural Language Processing		Building and Mining Knowledge Graphs;	Computer Vision;	
			Data Fusion;	Introduction to Quantum Computing for AI and Data Science	
			Explainable AI		
PROJECT			PROJECT		

Year 2

Semester 3			Semester 4		
Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Elective Semester: Courses and Research Project; Research Internship; Professional Internship; Study Abroad			**THESIS**		

Master Artificial Intelligence

Year 1

Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Intelligent Search and Games;	**Advanced Concepts in Machine Learning**		**Autonomous Robotic Systems**		
**1 of the electives:*	*1 of the electives:*		*1 of the electives:*	*1 of the electives:*	
Stochastic Decision Making;	Deep Learning for Image and Video Processing;	PROJECT	Dynamic Game Theory;	Information Retrieval and Text Mining;	PROJECT
Data Mining	Advanced Natural Language Processing		Planning and Scheduling;	Computer Vision;	
			Explainable AI	Reinforcement Learning;	
			Building and Mining Knowledge Graphs	Introduction to Quantum Computing for AI and Data Science	
PROJECT			PROJECT		

Year 2

Semester 3			Semester 4		
Period 1	Period 2	Period 3	Period 4	Period 5	Period 6
Elective Semester: Courses and Research Project; Research Internship; Professional Internship; Study Abroad			**THESIS**		

Periods 1 and 2 will last eight weeks each in total. During week 1-7 there are classes and in week 8 exams. Periods 4 and 5 will last 9 weeks each due to several holidays in these periods. Several courses are offered during each period, each course equals 6 credits (ECTS). Per course, five hours of classes are offered, on average, per week, in which a teacher will explain the theory of the subject or in which you do some practical training.

Depending on the master´s programme you enrolled in, you are required to take and pass several mandatory courses (4 for the AI master, and 4 for the DSDM master). These courses are underlined in the tables above. Besides these

Contents
Education
Bachelor
Master
Facilities
Staff

courses, there is a selection of electives in year 1. You are required to take 1 elective in each period in year 1. There are possibilities to take other Master courses taught at the Department of Advanced Computing Sciences as electives in year 2.

Next to these courses, in year 1 of your studies you participate every semester in a project of 6 credits that will last the whole semester. As a student, you are expected to participate actively in doing tasks with respect to the project skills training and project meetings. In addition, students are expected to cooperate actively with their group in order to successfully finish their project assignment. If a student fails to do so, the student might not be allowed to participate in the examination of the project, or the project examiners may deviate from the group grade for this individual student (see the Education and Examination Regulations (EER) and the Rules and Regulations (R&R) for details). The project of semester 1 runs during period 1, 2, and 3. The project of semester 2 runs during period 4, 5, and 6. Periods 3 and 6 last four weeks, and during the first three weeks of those periods students work fulltime to finish their project assignment. The fourth week is a resit week for the preceding two periods. The project assignment will be announced in the beginning of periods 1 and 4, and the group composition and project topic assignment is based on student preference in so far as possible. At the end of period 1, 2, 4, and 5, each group will give a short interim presentation. Each group will describe their project tasks and discuss their approach and schedule for the further elaboration. The examiners will give their reaction and judge whether the subjects are sufficiently covered in the project. These interim presentations are assessed. The elaboration stage of the project weeks will be concluded with the handing in of the final report, a dissemination website, handing in of the product, and a public oral presentation of the project result for each group. The assessment of the report, website, presentation and the product will - in principle - yield the same mark for all the group members. However, there can be diversification, see the Education and Examination Regulations (EER) and the Rules and Regulations (R&R).

Choose your own curriculum

During the first semester of the second year of the master's programme, you can choose your own curriculum from available options, enabling you to pursue your personal interests. During this semester, you can obtain 30 credits by choosing elective courses of the other master's programmes offered at the Department of Advanced Computing Sciences (i.e., AI or DSDM). In addition, you can also:

- take a combination of elective courses from the master's AI or DSDM and courses at another master program of Maastricht University;
- participate in a research project (a research internship) of the academic staff of the Department of Advanced Computing Sciences or at another research university;
- participate in an internship at a company;
- follow an exchange programme at one of our partner universities abroad;

The final stage of your master's programme, during the second semester of the second year, is reserved for conducting and writing your master's thesis that counts for 30 credits. The thesis is produced individually and is the result of a master research project on a topic that you will be working on under the supervision of one of the academic staff members of the programme. In the preliminary phase, the emphasis is on self-study, subject determination, approaching a supervisor, planning, and some preliminary research. After approval of the thesis research plan by the Board of Examiners, the actual research starts. In this phase, the student carries out his/her own research. The senior researcher that acts as the supervisor of this research process will guide the student during a series of regular appointments. The final phase is used to accomplish, i.e. write, the master's thesis. The master's thesis project is completed by an individual presentation and discussion of the results at the department. Assessment will be based on the research, the thesis itself, the process, the software and the presentation and discussion of this thesis (i.e., public defence).

Note that all individual curriculum choices are guided by our student advisor and are subject to approval by the Board of Examiners.

1.2.4 Project Centred Learning

All of our Bachelor's and Master's programmes employ project-centred learning (PCL), a variant of Maastricht University's signature problem-based learning. . The PCL educational model is small-scale and student-oriented. You work in small groups on two complex and challenging projects per year (amounting to one per semester). These projects last around five months each and run parallel to courses, which follow a lecture- and tutorial-based setup. The level of the projects, and the products students need to deliver, matches the students' study progress by requiring knowledge obtained from coursework. In addition to technical skill development, this educational model trains employability skills such as teamwork, project planning, documenting, and presenting.

We base projects on real-life research and/or societal challenges provided by our staff and by companies. This gives our students the opportunity to gain invaluable experience by applying the learned knowledge to finding solutions to real-world problems. Together with fellow students, you research on what existing approaches can be adapted, and design and evaluate new approaches for the problem at hand. At the end of each project, you deliver a functional product and present your findings to your fellow students, the teachers and/or the client.

Project-Centred Learning has advantages:
- from the beginning you find out what teamwork means
- you learn project-related skills in a natural way
- you will be continuously placed in an active role
- you will be able to match theory with its applications
- PCL increases the student's motivation

Projects for instance require students to design a quadcopter platform for an autonomous swarm, to optimize timetables, to model human decision process from intercranial EEG, to automate the recognition of facial expressions, or to design and implement a traffic simulator.

1.3 Study Abroad

In the bachelor's Data Science and Artificial Intelligence and the two Master's Data Science for Decision Making and Artificial Intelligence we have one of the highest ratios of international students. More than 75% of the scientific staff and 75% of the students are non-Dutch, giving rise to an international study environment. Additionally, we host a number of international exchange students each year and offer our own students a number of opportunities for international experiences themselves. For example, we offer students the opportunity to study abroad for a semester during the elective semester. To this purpose, we collaborate with well-established partner universities.

For bachelor students there there is an option to participate in an international study abroad programme during the fifth (elective) semester of the bachelor's programme. For master students there is an option to participate in an international study abroad during the third (elective) semester of the master's programmes.

Please note that for students entering the Master programmes during the February intake, fewer universities may be available for study abroad. More documentation and information about the participating exchange partner universities is available on Canvas under the "Going Abroad" section (see the application form). Also, here you can find the Study Abroad Guide.
For additional information, you can contact the international relations officer or your study advisor.

1.4 Degree

A successful conclusion of the bachelor's programme will provide you with a bachelor's certificate according to Dutch law, that is, a 'Bachelor of Science'. A successful conclusion of a master's programme will provide you with a master's certificate according to Dutch law, that is, a 'Master of Science'.

1.5 Study Feasibility and Quality Assurance

We strive for continuous improvement of the quality and feasibility of our study programmes. Student evaluations of each course help us in maintaining a high standard of educational quality and keeping the study programmes feasible. The study programme's feasibility means that a student with an appropriate background should be able to finish the study within the set number of years.

To maintain this standard, the quality assurance officer collects information about teaching, learning and assessment at the end of each period. The quality assurance officer then reports the outcomes of the student evaluations to the Education Programme Committee (EPC) of the respective Bachelor's or Master's programme. EPC consists of representatives of staff and students (5 staff representatives and five student representatives -- two students of Bachelor DSAI, one student of Bachelor CS, and two master students, one for each Master's programme). If the outcomes are unsatisfactory, the EPC will take action to improve the quality of a specific course (or project) or of the study as a whole. Therefore, student responses are essential in pointing out strong aspects and aspects for improvement of all educational activities.

As a student in one of our Bachelor's or Master's programmes, you are encouraged to give your opinion about each course, as it may help improve that course. Moreover, future students may benefit from the results and comments of your evaluation just as you may benefit from course evaluations of fellow students.

1.6 Courses at other Faculties or universities

Bachelor and Master

If a student from one of our Bachelor's or Master's programmes would like to participate in courses at other programmes of Maastricht University or other universities, approval from the Board of Examiners is needed in advance. For more information, the student counsellors will upload information on Canvas.

A special position among electives from outside the bachelor's programme is the Educational Minor. The Educational Minor leads to a limited second-degree teaching qualification. Upon successful completion of the minor, you are qualified to teach in the first, second and third year of VWO, HAVO and VMBO-tl (MAVO) level in the Netherlands.

Students in the BSc Data Science and Artificial Intelligence can - upon successful completion - acquire a teaching qualification for the main subject of Mathematics. The Educational Minor is organised in close cooperation with the Fontys Leraren Opleiding (Teacher Training) in Sittard (FLOS) and Tilburg (FLOT). The language of the Educational Minor is **Dutch**.

The programme contains several pedagogical-didactic courses in semester 5, along with education aimed at teaching methodology. There is also a mandatory practical internship, in the form of work placements, which is spread out over semesters 5 and 6. The education meetings mostly take place at UM and occasionally at the Fontys Leraren Opleiding in Sittard. The practical internship will be done at several secondary schools in the whole province of Limburg and will continue until the end of semester 6. During the practical internship, the student spends one day a week at a secondary school for the course of a full school year. In this way, the necessary teaching experience is obtained. Knowledge and practice are closely connected in the Educational Minor.
Successful completion of the educational minor yields 35 ECTS of which five are extracurricular. This means that these 5 ECTS cannot be used to replace any other components of the original bachelor program. Students in the BSc Data Science and Artificial Intelligence who wish to participate in the program should have accumulated, by the end of their second year, all 60 ECTS from the first-year components and at least 52 ECTS from the second year components. Prior to their enrolment in this minor, a motivated request for participation has to be submitted in Dutch to the Board of Examiners dacs-boe). Enrolment is dependent on selection and prior permission of the Board of Examiners.
If you have any questions regarding the contents of this educational minor, then please contact Prof. Dr. Frank Thuijsman (f.thuijsman@maastrichtuniversity.nl).

1.7 Department of Advanced Computing Sciences Honours Programmes

The Department of Advanced Computing Sciences offers its talented and top-performing Bachelor's students the possibility to participate in an Honours Programme. This programme offers two variants MaRBle 2.0 and KE@Work.

MaRBle 2.0

In MaRBle 2.0, you will get the opportunity to work on a state-of-the-art research project. Work will be organized in a similar way as in professional research institutes where participants work together as individual experts on a team project. More information will follow in the first semester. MaRBle takes place in year two of the Bachelor's programme.

KE@Work

KnowledgeEngineering@Work (KE@Work) Students admitted to the KE@Work path are placed at a company or organisation through a careful selection and matching process. During the full second and third year of the bachelor program, they spend 50% of the time in class and 50% at the company, where they work on solving academic challenges and complex business problems, under supervision of dedicated business and university supervisors. For more information on KE@Work, please contact our colleagues via kework@maastrichtuniversity.nl

Eligibility requirements for KE@Work and MaRBLe 2.0 entail that students:

- *have passed all courses/components in the first year of the Bachelor's programme at first opportunity;*
- *have obtained a GPA of at least 7.5 for all courses of year 1 (to be eligible for pre-selection a GPA of at least 7.5 has to be obtained for blocks 1 to 4);*
- *have not been convicted of fraud and have not been reprimanded for a violation of house rules or code of conduct.*
- *You can find the further criteria and leniency in the Rules and Regulations.*

Selection of honours students will happen in the second semester of year 1. If you successfully complete the honours programme, this will be certified on an honour's diploma supplement.

1.8 Regulations

There are a number of Rules and Regulations that we expect you to be familiar with. An overview of the regulations can be found here

For examination, The Education and Examination Regulations (EER), the Rules and Regulations (RR) and the Rules of Procedure for Examination are of particular interest:

https://intranet.maastrichtuniversity.nl/en/dacs-students/exams-rules-and-regulations/examination

2 Bachelor

2.1 Curriculum of the First Year of the Bachelor Programme Data Science & Artificial Intelligence

In order to learn how the transformation of raw data into useful information and knowledge is achieved, and how this transformation can be automated with the help of artificial intelligence, a thorough basic knowledge of specific mathematics and computer science subjects is required. This means that the first year is largely filled with mathematics and computer science subjects. Additionally, you will also follow courses on topics that help the understanding and broadening of the fields of Data Science and Artificial Intelligence, such as Computational & Cognitive Neuroscience and ICT & Knowledge Management.

The year is divided into four periods of eight weeks with three courses each, and two periods of four weeks during which you will work on a project. Each project is preceded by partial project assignments during the other periods. The week schedule works with two-hour clusters. In the overview below, the courses are indicated, as well as the study load in credits (ECTS). One ECTS stands for about 28 hours of study time (lectures, meetings and self-study). Besides the lectures that are given on the subjects, there will also be practicals and skills training.

Year 1		ECTS
Period 1.1	Introduction to Data Science and Artificial Intelligence (KEN1110) Procedural Programming (KEN1120) Discrete Mathematics (KEN1130) Project 1-1 (*)	4 4
Period 1.2	Objects in Programming (KEN1220) Calculus (KEN1440) Logic (KEN 1530) Project 1-1 (*)	4 4 4
Period 1.3	Project 1-1 (KEN1300)	6
Period 1.4	Data Structures and Algorithms (KEN1420) Linear Algebra (KEN1410) Principles of Data Science (KEN1435) Project 1-2 (*)	4 4 4
Period 1.5	Computational & Cognitive Neuroscience (KEN1210)) Numerical Methods (KEN1540) Software Engineering (KEN1520) Project 1-2 (*)	4 4 4
Period 1.6	Project 1-2 (KEN1600)	6

(*) Project 1-1 will start in period 1.1 and will run until period 1.3; Project 1-2 will start in period 1.4 and will run until period 1.6. The credits for the projects will become available at the end of period 1.3 and period 1.6, respectively. Please see the course description section Project 1-1 and Project 1-2 for more details on the project curriculum. For each period, we will give a short explanation of the various parts. Before the start of each period, the students will receive detailed information about the content, the study material, the teaching form, the schedule, and the examination method.

Period 1.1

Introduction to Data Science and Artificial Intelligence (KEN1110)

Examiner: Dr. Rachel Cavill, Dr. Pietro Bonizzi, and Prof. dr. Anna Wilbik

Desired Prior Knowledge: The course appears as desired prior knowledge for the courses Reasoning Techniques and Theoretical Computer Science.

Prerequisites: None.

Course description: The course Introduction to Data Science and Artificial Intelligence offers a comprehensive overview of the core topics in Data Science and Artificial Intelligence, both from a mathematical and from a computational perspective. Particular emphasis is put on the basic classes of techniques and methods, the theoretical underpinnings of data science and computational intelligence, and some example application domains of data science and artificial intelligence.
As such, the course provides an overview of many topics that are addressed in much more detail throughout the Bachelor's Data Science and Artificial Intelligence programme.

Knowledge and understanding: After successful completion of the course, students will be able to recognise what real world problems require the use of data science, and approach their solution by using a data science process, namely: explore the data, model the data, and perform simulations if required. Moreover, they will exhibit knowledge in the basic concepts of artificial intelligence, such as agents, search, artificial intelligence, decision trees.

Applying knowledge and understanding: Students learn to recognise applications of data science and artificial intelligence in different domains and apply the basic techniques they have learnt from both.

Making judgements: Upon completion of the course, students are able to recognise the relevant domains of data science and artificial intelligence when confronted with data science and artificial intelligence problems.

Communication: Students are able to explain the process they used to generate results and communicate the meaning of those results in context.

Learning skills: Students will be able to recognise small-scale data science problems and autonomously and critically reflect upon the appropriateness of data science and artificial intelligence methods for tackling those, and propose a primary solution.

Study material: Material will be provided during the course.

Recommended literature:
- S. Russell and P. Norvig (2010): Artificial Intelligence, A Modern Approach. Third edition, Pearson Education, ISBN 978-0-13-207148-2.
- C.D. Manning, P. Raghavan and H. Schütze (2008) Introduction to Information Retrieval. Cambridge University Press. ISBN 0521865719

Exam: There will be a closed book written exam at the end of the course.

ECTS: 4

Procedural Programming (KEN1120)

Examiner: Dr. Enrique Hortal, Dr. Tom Bitterman, and Dr. Daniel Cámpora

Desired Prior Knowledge: None. The course appears as desired prior knowledge for the courses Introduction to Computer Science 2, Data Structures and Algorithms, Software Engineering, Databases and Machine Learning.

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

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

Knowledge and understanding: The course offers preliminary methodological and theoretical bases for studying and applying computers and computer programming on which the rest of the curriculum builds.

Applying knowledge and understanding: Whenever a computer system or a programming system has to be designed and implemented the knowledge and insights acquired during the course can be used and applied.

Making judgements: After successful completion of the course, students will be able to judge the quality and correctness of simple non-object-oriented programs.

Communication: The skills acquired during the course will enable students to communicate about standard programming constructs and algorithmic basics.

Learning Skills: After successful completion of the course, students will be able to formalize, analyse and program solutions to simple software problems.

Study material: Lecture slides, example code and multimedia material that are made available before and after each lecture.

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

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

Exam: Closed-book written exam (80%) + Assignments (20%)

ECTS: 4

Discrete Mathematics (KEN1130)

Examiner: Dr. Marieke Musegaas, Dr. Otti D'Huys, and Dr. Stefan Maubach

Desired Prior Knowledge: None.

Prerequisites: None.

Description: In this course, we build a mathematical framework that is based on logic and reason. The main objective of the course is to make students familiar with the language of mathematics. Students will learn how to make sound arguments and to detect where and why certain arguments go wrong. For this purpose, we will discuss the basic principles of logic and, closely related, the basic types of mathematical proofs. In doing so, we will encounter numbers such as integers, natural numbers and real numbers and we shall examine what makes these numbers special. After that, we will use basic logic to discuss, among other things, the following mathematical concepts: infinity, sets, relations, functions, permutations and combinations. Our fundamental tool in all of this is plain common sense. You really do not need your toolbox of mathematical formulas learned in previous studies and neither do you need a calculator. Pen and paper are the basic instruments needed. After completing each topic, exercises will be provided to be completed in class or at home, since mathematics is mainly learned by practising repeatedly.

Knowledge and understanding: Students will be able to read, interpret and manipulate basic mathematical terminology (propositional logic, quantifiers, set theory, relations, functions, and combinatorics). Students will also be able to read and interpret several different types of mathematical proofs and identify whether a purported proof is mathematically sound.

Applying knowledge and understanding: Upon completion of the course students will know how to read, interpret, write and manipulate rigorous mathematical statements using propositional logic, quantifiers, set theory, relations, functions and combinatorics. Students will be able to select, from a range of mathematical tools, which is appropriate to prove or disprove a given mathematical statement, and apply the chosen tools, rigorously and clearly in order to achieve the desired goal.

Making judgements: Students will be able to distinguish between mathematically sound and unsound statements and defend the rigour of their own mathematical arguments.

Communication: Students will be able to write clear, rigorous and explicit mathematical arguments using standardized mathematical terminology and such that each step in the argument is a logical consequence of earlier steps.

Learning skills: By the end of the course, students will be able to autonomously and critically reflect upon the mathematical correctness of their own arguments.

Study material: A. Chetwynd & P. Diggle: Discrete Mathematics. Butterworth- Heinemann, Oxford, ISBN 0 340 61047 6. Lecture notes will also be provided.

Recommended literature: None

Exam: Closed book written exam.

ECTS: 4

Course title: Project 1-1 (KEN 1300)

Coordinator: Dr. Martijn Boussé

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

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

Knowledge and understanding: Interpret constraint-satisfaction problems arising in practice and translate this to discrete-mathematical algorithmic models capable of solving the problem. Gain insight into practical use of basic software design and development principles. Recognise and relate user-computer interactions to concepts from graphics and user-interface frameworks. Strengthen knowledge of basic algorithms and methods for efficiently solving constraint-satisfaction problems arising in applied mathematics (especially: discrete mathematics) and artificial intelligence.

Applying knowledge and understanding: Design an answer strategy for scientific questions using analytical thinking and logical reasoning. Translate discrete-mathematical algorithmic models to software code. Implement software to efficiently solve constraint-satisfaction problems arising in applied mathematics (especially: discrete mathematics) and artificial intelligence by finding, designing and applying appropriate algorithms. Formulate computational experiments, and analyse and interpret the results. Apply basic design and development principles in the construction of software systems. Use existing software application frameworks for graphics and user interfaces. Use tools for software project management such as version control systems and issue trackers. Identify project goals, deliverables, and constraints. Plan and chair meetings. Create minutes for meetings. Work in a team such that the workload is balanced. Plan teamwork by setting deadlines and distributing tasks.

Making judgements: Evaluate different mathematical and computational models with respect to their suitability, efficiency and correctness for a specific task. Elicit and evaluate relevant scientific background information. Evaluate the group's progress during the project.

Communication: Give a clear and well-constructed presentation, including a demonstration of the product, and with appropriate use of illustrations and/or videos. Offer and respond to questions on and constructive criticism of presentations. Write a project report according to the structure of an academic article. Submit arguments in exact sciences, with appropriate use of formulae and figures. Cite published sources in the project report according to the academic guidelines. Structurally inform stakeholders on project progress. Effectively communicate with project group members about task division, planning and project deadlines. Effectively communicate with group members by listening to others' ideas; be contactable and include others in the discussion. Cooperate in a group to reach a consensus view. Give constructive feedback to team members. Communicate in the English language.

Learning skills: Reflect on one's own academic abilities and functioning in a team.

Study material: Project manual project 1-1, Maastricht University

Assessment:
Final Grade = 0.9 · (0.15 · gradePhase 1+0.15 · gradePhase 2+0.7 · gradePhase 3)+ skillClassGrade +individualGrade

where gradePhase 3=0.4 gradeProduct+0.4 gradeReport+ 0.1 gradePresentation+ 0.1 gradeProjectManagement.

The *individualGrade* is given due to either outstanding or not enough contribution of a student to the project. By passing skill classes, the students can get a reward called *skillClassGrade*, which is 1 if the students passed all skill classes, 0.5 if the students passed all but one skill classes, and 0 if the students passed all but two skill classes. Failing more than 2 skill classes will lead to an NG in the project. Missing mandatory project events such as project meetings and examination moments will lead to a reduction of the grade or even to receiving an NG for the project.

Skill classes:

Introduction to Project Work (period 1.1)
The students learn how PCL approach is applied in the programme. The structure of the projects is explained. Furthermore, the students will get to know their team members and will be introduced to various aspects of project work such as taking minutes and creating an agenda.

Elementary writing (period 1.1 and 1.2)
In period 1-1, the students have to take part in an English Diagnostic Test that determines their current English level. Those students, which scored low in the diagnostic test, will receive an additional training on elementary writing in form of a skill class. Furthermore, all students have to pass a quiz about writing.

Elementary presenting (period 1.2)
The students learn the basics of a good presentation. They will have the opportunity to revise the first draft of their presentation based on the learned concepts.

Citing and Referencing (period 1.2)
This skill class provides you with basic knowledge about when and how to cite.

Team Dynamics "How to effectively communicate and cooperate in my team?" (period 1.3)
In this skill class, you will learn effective communication strategies that will help you to cooperate in your team.

Advanced Presenting (period 1.3)
In this skill class, you will present within your project group and receive individual feedback on presentation style and content.

ECTS: 6

Period 1.2

Objects in Programming (KEN1220)

Examiner: Dr. Thomas Bitterman, Dr. Evgueni Smirnov, and Dr. Francesco Barile

Desired prior knowledge: Basic Java Programming

Prerequisites: None.

Description: This course is a follow-up to the course Introduction to Computer Science 1. It teaches object-oriented programming in Java. The main topics covered in the course are objects and classes, interfaces and polymorphism, event handling, inheritance, graphic user interfaces, exception handling, and streams.

Knowledge and understanding: After successful completion of the course, students will be able to explain the methodological and theoretical principles of object-oriented programming.

Applying knowledge and understanding: Students will be able to implement basic object-oriented computer programs. They will be able to design and describe simple object-oriented computer systems.

Making judgements: Students will be able to judge the quality and correctness of simple object-oriented programs.

Communication: Students will be able to communicate about object-oriented programming constructs and algorithmic basics.

Learning skills: Students will be able to recognize their own lack of knowledge and understanding and take appropriate action such as consulting additional material or other sources of help.

Study material: Course notes, slides, and other information made available.

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

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

Exam: Written exam (80%) + practical assignments (20%).

ECTS: 4

Calculus (KEN1440)

Examiner: Dr. Otti D'Huys, Dr. Martijn Boussé, and Dr. Gijs Schoenmakers

Prerequisites: None.

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

Knowledge and understanding: Student can define, write and explain key facts and concepts involving limits and continuity, can interpret and solve differential calculus, integral calculus, sequences and series, first-order linear differential equations problems, and understand the basics of multivariable calculus.

Applying knowledge and understanding: Students are able to solve problems applying the concepts learned in the course, using standard problem-solving strategies.

Making judgements: Students are able to analyse a simple problem within the course content and justify the solution methodology they choose. They can summarize this methodology mathematically.

Communication: Students are able to explain their solution strategy in written form and defend their solution strategy in discussion with others

Learning Skills: After successful completion of the course the students will be able both to solve standard problems (constructing graphs of functions, finding extrema of functions, computing limits, summing infinite series etc.) and to apply their knowledge in solving and analysing more complex problems (e.g. in analysis of numerical algorithms).

Study material: Calculus, a complete course, any edition, by R.A. Adams, Addison Wesley Longman and materials provided during the lectures.

Exam: Intermediate bonus assignments and a final written exam.

ECTS: 4

Logic (Code: KEN1530)

Examiner: Dr. Tjitze Rienstra, Dr. Stefan Maubach, and Dr. Nico Roos

Prerequisites: None. The course appears as a prerequisite for the course Logic for AI..

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

Knowledge and understanding:
Students need to get accustomed to the fundamental concepts of mathematical logical systems (propositional logic and predicate logic) to able to describe information in a logical framework and to reason and prove correctly. Students will get accustomed to the basic concepts of some advanced logical systems (epistemic logic and dynamic logic).

Applying knowledge and understanding:
Students will apply the reasoning and proof methods learned to small-scale problems and some more complex situations.

Making judgement:
Students will learn to judge how to reason correctly using mathematical proofs and how to judge which logical system is suitable to solve the problem at hand.

Communication:
The chosen syntax of the logical language used must be easily understandable by peers and others experts the logical proofs given must be correct, concise and easily understandable.

Learning skills:
Having learned basic logical concepts and reasoning techniques the students are able to apply them to larger-scale problems.

Study material: Johan van Benthem, Hans van Ditmarsch, Jan van Eijck, Jan Jaspars, Logic in Action. Edition of February 2014 or later. This is a freely available e-book. Check your Canvas for the link.

Exam: Written exam; during the course the students will receive three assignments, that, if they receive a sufficient grade, may earn them up to a total of one bonus point.

ECTS: 4

Period 1.4

Data Structures and Algorithms (KEN1420)

Examiner: Dr. Francesco Barile, Tom Pepels, M.Sc., and Dr. Daniel Cámpora.

Tutor(s): TBA.

Desired Prior Knowledge: Discrete Mathematics, Introduction to Computer Science 1 and 2. The course is desired prior knowledge for Theoretical Computer Science.

Prerequisites: None. The course itself occurs as part of the pre-requisites of both projects of year 2 and the third year course Parallel Programming.

Description: As a continuation of the courses Computer Science 1 and 2, this course will treat the systematic design and application of data structures and algorithms. Data structures such as lists, trees, graphs, and dictionaries, the associated algorithms and their complexity are explored in this course. Algorithms for applications such as sorting, pattern matching and graph traversal are also part of the course. Furthermore, design principles for algorithms such as recursion, divide-and-conquer and dynamic programming will be treated as well. Furthermore, students will develop skills to analyse the run-time and space complexity of data structures and algorithms.

Knowledge and understanding: Students are able to give examples of data structures and explain do they support program design. Students are able to name what types of standard data structures exist and illustrate their properties. Students are able to describe some standard algorithms and highlight their properties. Students are able to illustrate how to develop and analyze new algorithms.

Applying knowledge and understanding: Students are able to select the appropriate data structure for a given problem and students are able to propose an algorithm for solving a given problem

Making Judgements: Students are able to justify if and determine how data structures are applied. Furthermore, students are able to assess whether algorithms are appropriate and efficient.

Communication: Students are able to explain how data structures and algorithms are to be included in program designs.

Learning Skills: Students are able to reflect on which data structures and/or algorithms are applicable for each problem.

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

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

Exam: 'Closed Book' written exam, during the course the students will receive a number of assignments, which can earn them up to a total of one bonus point.

ECTS: 4

Linear Algebra (KEN1410)

Examiner: Dr. Marieke Musegaas, Dr. Philippe Dreessen, and Dr. Steve Chaplick.

Desired Prior Knowledge: None.

Prerequisites: None.

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

Knowledge and understanding: Students are able to recognize and explain the fundamental concepts of Linear Algebra: systems of linear equations, vectors and vector spaces, basis and coordinates, matrices and matrix-vector computations, linearity and orthogonality, linear independence, rank, fundamental spaces (row space, column space, and null space), determinants and invertibility, eigenvalues and Eigen spaces, diagonalization.

Applying knowledge and understanding: Students are able to analyse a linear algebra problem from both an algebraic and a geometrical point of view. Students can solve systems of linear equations, compute determinants and rank, compute eigenvalues and Eigen spaces, make use of complex numbers, diagonalizable matrices, and perform change of coordinates.

Making judgements: Students are able to look at the same problem from different angles and to switch their point of view (from geometric to algebraic and vice versa).

Communication: Students are able to motivate both from an algebraic and a geometric point of view the solution set of a system of linear equations, the linear independence and orthogonality of a set of vectors, the linear transformation between two coordinate systems, the fundamental spaces associated with a matrix, the invertibility of a matrix, and the diagonalization of a matrix in terms of the properties of its eigenvalues and eigenvectors.

Learning skills: Students have acquired the skills to autonomously recognize elements of practical problems, which can be addressed and solved with linear algebra, and use Matlab to solve larger scale problems.

Study material: David C. Lay, Linear algebra and its applications, 6th ed., Pearson, ISBN: 978-1-292-35121-6.

Recommended literature: None.

Exam: There will be a closed book written exam at the end of the course.

ECTS: 4

Principles of Data Science (KEN1435)

Examiner: Dr. Christof Seiler, and Dr. M. ten Thij.

Desired Prior Knowledge: Introduction to Computer Science 1

Prerequisites: None.

Course description: Nowadays data science is at the core of modern society. We collect large amounts of data with the goal to make better decisions. We need to make sense of the data and leverage it in effective ways.

In this course, we will start with where data comes from—controlled experiments and observational studies. We will look at potential biases that can affect conclusions that we make from data. We will focus on what kind of causal statement one can draw based on data coming from experiments versus observational studies.

We will then summarize and visualize data using histograms and scatter plots. As we will see, there are some interesting recurring patterns when we summarize data. For example, the distribution of the average follows the bell shape curve. We will also consider deviations from the bell shape curve in case of outliers, and how to deal with real world and possible "unclean" data. Scatter plots will help us study the regression line and correlations.

This course will build the foundation for subsequent courses: probability and statistics, simulation and statistical analysis, and machine learning. You will learn how to convert data into tables and use them for subsequent analysis and plotting. We will focus on the principles of modern reproducible science, that is, to build analysis workflows that can easily be understood and re-run by others. We will learn how to keep track of analysis decisions and parameter choices. We will summarize all the uncertainties in an accessible way and see that this is crucial for effective decision making in the modern world.

During the labs, we will learn R—one of the main programming languages used in data science—and how to use it to write analysis reports using literate programming—mixing code, plots, and narrative in the same document. We will analyze and visualize real datasets.

Knowledge and understanding: Students learn to organize, analyze, and visualize data. They understand what type of distribution to expect after summarizing the data.

Applying knowledge and understanding: Students analyze real datasets. They apply their knowledge about summaries of the data—and other tools in data science—and determine where they can be appropriately applied. They translate their understanding into conclusions for domain experts. Students develop skills to generalize data analyses to unseen contexts.

Making Judgement: Students decide the limits on what can be learned from data. They judge the data based on the design of the experiment and the final goal of the analysis. They also predict the consequences of data misuse when making causal claims.

Communication: Students communicate their findings in written analysis reports. They write reports in R markdown and compile them to html reports, so that they are accessible for domain experts.

Learning Skills: Students develop skills to turn an abstract question into an actionable decision to gain insights.

Study material: Statistics (fourth edition, 2007) by Freedman, Pisani, and Purves. Additional selected material from data science textbooks and other resources.

Exam: 20% homework assignments and 80% written final exam.

ECTS: 4

Period 1.5

Computational and Cognitive Neuroscience (KEN1210)

Examiner: Dr. Alard Roebroeck and Dr. Michael Capalbo

Desired Prior Knowledge: None

Prerequisites: None.

Description: The course Computational and Cognitive Neuroscience presents an overview of the core topics in cognitive and biological psychology. These topics include (human) perception, learning, memory, planning, problem solving, reasoning, language, speech, and action. Both the functional and neuroanatomical foundations of cognitive faculties are addressed. Several models of cognition and theories of brain function that are of relevance to knowledge engineering will be outlined. Several skills trainings will be given to train understanding in biological functioning of neuronal communication, and functioning of neural networks and genetic algorithms.

Knowledge and understanding: The student can recount the main points of the domain of cognitive science

- The student can describe the main points of the domain of cognitive science
- The student can explain the following (human) behaviours while using these points: perception, learning, memory, planning, problem solving, reasoning, language, speech, and action.
- The student can identify the computational aspects and computational applications of these fields

Applying knowledge and understanding:

- This knowledge is applied in in two practical assignments in which the students are asked to create a genetic algorithm and a neural network

Making judgements:

- Upon completion of the course, students are able to interpret data and literature about a subject in (or related to) the domain of cognitive and biological psychology.
- Using the data and literature, they can support judgements about the societal, scientific or ethical aspects of the subject.

Communication:

- Students are able to communicate ideas and solutions to an audience of non-experts and experts.

Learning skills:

- Students have acquired the skill to translate theoretical models into computational models.

Study material: Material will be provided during the course.

Recommended literature: Sternberg, R.J. (1999). Cognitive psychology (latest edition). Fort Worth: Harcourt Brace. Kalat, J.W. (2007) 9th edition Biological psychology. Pacific Grove, California; London: Brooks Cole. Gazzaniga, M. (2009). Cognitive Neuroscience (third edition).

Exam: Written exam.

ECTS: 4

Numerical Mathematics (Code KEN1540)

Examiner: Dr. Pieter Collins and Dr. Martijn Boussé

Desired prior knowledge: Calculus, Linear Algebra

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

Knowledge and understanding: By the end of this course, students will have knowledge of the fundamental problems of numerical mathematics and basic techniques for their solution. You will understand issues of efficiency and numerical accuracy, will be able to analyse which numerical methods are likely to perform best on different types of problem, and evaluate whether the results of a given computation are trustworthy. You will be able to write your own code (in MATLAB) implementing basic numerical algorithms. Advanced students will have the skills necessary to adapt existing numerical algorithms and develop new algorithms.

Applying knowledge and understanding: Students will be expected to implement the algorithms covered in the lectures themselves, apply them to practical problems, and explain the performance of different algorithms in terms of theoretical analyses.

Making judgements: Students will learn how to analyse which numerical methods are likely to perform best on different types of problem, and to evaluate whether the results of a given computation are trustworthy.

Communication: Students will learn the terminology required to discuss numerical algorithms and the results of numerical computations with mathematicians, (social) scientists and engineers.

Learning skills: Students will learn to design, analyse, implement and apply numerical methods.

Study material: Slides, pre-recorded lectures, exercise sheets.

Recommended literature: Faires & Burden, “Numerical Methods”.

Exam: Written examination with formula sheet (80%). Computer-based homework exercises (20%) and preparatory exercises (+10% Bonus). Additionally, to achieve a passing grade, a sufficient score must be achieved on the written examination.

ECTS: 4

Software Engineering (KEN1520)

Examiners: Tom Pepels, M.Sc., and Dr. Daniel Cámpora

Desired prior knowledge: Introduction to Computer Science 1 and 2, Data Structures and Algorithms.

Prerequisites: None

Description: This course introduces students to software design and project management concepts. Students get introduced to multiple techniques they require to work on medium and large-scale projects in professional business and research environments. Students learn how to produce professional, reliable, and cost-efficient software that can be developed in a team, reused, maintained, further evolved, and that is tested professionally. Covered concepts include requirement engineering, project planning, risk management, software evaluation and testing, software engineering processes, design principles, software architectures, design patterns and principles, API development, code review, version control, specifications, debugging, refactoring, and abstract data types.

Knowledge and understanding: Students learn how to efficiently design and write professional software that meets specifications made by themselves or by a customer. Students learn what the essential elements of the software engineering process are. These elements include requirement analysis, design methodologies, implementation strategies, and validation techniques.

Applying knowledge and understanding: Students acquire the skills
- to critically analyze software requirements, software designs, software implementations, and software evaluations.
- to efficiently plan, execute, and monitor progress in group projects.
- to cooperate better in a group and to participate more effectively as a professional in academia or business environment.

Software engineering is a core activity of knowledge engineers and data scientists. In their professional career, students often will act as active programmers, software designers, and project managers where they need to create software as part of a team or act as team leader. The project management skills and tools being taught in this course that help students creating professional and cost-effective software are thus crucial for their career and lay the foundation for further studies of in this field. The knowledge obtained will directly help students for the subsequent semester projects and Bachelor thesis.

Making judgements: Students learn to judge the viability of selected software development methodologies and new developments in design concepts during their career. Students learn to compare design choices and judge their consequences.

Communication: Students learn to discuss and document software developments professionally. Knowledge about widely spread standard software development techniques and about standard design patterns are essential for efficient communication between software developers. Standards in software engineering facilitate cost-effective communication and help to avoid misunderstandings between customers and suppliers, between team leaders and team members as well as between team members.

Learning skills: Students learn to successfully reflect on their project management skills, on how they contribute to a software project as part of a team, and on how to adjust their software engineering approaches to different professional scenarios. Students learn to reflect on and to verify own and others software designs and implementations in a professional manner.

Study material: Lecture material provided during the lecture.

Recommended literature:

- Goldman and Miller, MIT 6.031: Software Construction, http://web.mit.edu/6.031/
- Martin, Clean Code: A Handbook of Agile Software Craftsmanship (2008)

Exam: Written "closed-book" exam at the end of the course. During the course, students receive several graded assignments that count for 20% of the final grade.

ECTS: 4

Period 1.6

Course title: Project 1-2 (KEN 1600)

Coordinator: Dr. Otti D'Huys

Prerequisites: In order to participate in this project the student has to have passed two out of four courses from the set: Discrete Mathematics, Calculus, Computer Science I and Computer Science II.

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

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

Knowledge and understanding: Interpret the meaning of mathematical models of real-world processes. Gain insight into practical use of software design and development principles. Recognise and relate user-computer interactions to concepts from graphics and user-interface frameworks. Strengthen knowledge of basic algorithms and methods for specific problems in artificial intelligence and applied mathematics.

Applying knowledge and understanding: Students will be able to design an answer strategy for scientific questions using analytical thinking and logical reasoning and to translate mathematical models to software code. Furthermore, students will be able to implement software to solve problems in applied mathematics by applying numerical methods and artificial intelligence algorithms, formulate computational experiments, and analyse and interpret the results, apply design and development principles in the construction of software systems and use existing software application frameworks for graphics and user interfaces. Even more so, students will learn to use tools for software project management such as version control systems and issue trackers, identify project goals, deliverables, and constraints. Lastly they will learn how to plan and chair meetings, create notes for minutes, work in a team such that the workload is balanced and plan teamwork by setting deadlines and distributing tasks.

Making judgements: Students will learn to evaluate different mathematical and computational models with respect to their suitability, efficiency and correctness for a specific task.

Communication: Students will be able to give a clear and well-constructed presentation, including a demonstration of the product, and with appropriate use of illustrations and/or videos, to offer and respond to questions on and constructive criticism of presentations. Furthermore, they will learn to write a project report according to the structure of an academic article, submit arguments in exact sciences, with appropriate use of formulae and figures. They learn to cite published sources in the project report according to the academic guidelines. Additionally, students will learn to structurally inform stakeholders on project progress and effectively communicate with project group members about task division, planning and project deadlines, effectively communicate with group members by listening to others' ideas; be contactable include others in the discussion. It will be important to cooperate in a group to reach a consensus view, communicate in the English language, elicit and evaluate relevant scientific background information.

Learning skills: Reflect on one's own academic abilities and functioning in a team.

Study material: Project manual project 1-2, Maastricht University.

Assessment:
Final Grade = 0.9 · (0.15 · gradePhase 1+0.15 · gradePhase 2+0.7 · gradePhase 3)+ skillClassGrade +individualGrade

where gradePhase 3=0.4 gradeProduct+0.4 gradeReport+ 0.1 gradePresentation+ 0.1 gradeProjectManagement.

The *individualGrade* is given due to either outstanding or not enough contribution of a student to the project. By passing skill classes, the students can get a reward called *skillClassGrade*, which is 1 if the students passed all skill classes, 0.5 if the students passed all but one skill classes, and 0 if the students passed all but two skill classes. Failing more than 2 skill classes will lead to an NG in the project. Missing mandatory project events such as project meetings and examination moments will lead to a reduction of the grade or even to receiving an NG for the project.

Skill classes:

Information Research: Systematic Literature Search (period 1.4)

This skill class will give the students an introduction to which databases, search strings and settings can be used to systematically search for literature. Furthermore, they will have to come up with a search plan for the current project.

Team Dynamics "Laying the foundation of effective teamwork" (period 1.4)

This team dynamics workshop aims to provide the students with a deeper awareness, insight and practice in effective team collaboration & co-creation. During this introduction workshop, you and your project team will draw up a team charter to initiate effective group collaboration in project 1-2.

Advanced Presenting (period 1.4)

This skill class will focus on presentation skills and techniques. This introduction helps with public speaking and prepares you for project report presentations. Areas of focus include: structure of a presentation, public speaking techniques and enunciating, language aspects to remember while planning a presentation, and the dos and don'ts expected by Maastricht University.

Academic Writing (period 1.5)

In the project skills meetings you will explore the key structure of your report, as well as key points of Academic Writing at Maastricht University. Areas of focus include: structure of paper; linguistic aspects of writing in English, presenting information logically and citation and reference procedures.

Team Dynamics "Evaluating and drawing lessons from the project teamwork" (period 1.6)

In this skill class, students are going to evaluate the team collaboration and communication during project 1-2 by means of interactive exercises.

ECTS: 6

2.2 Curriculum of the Second Year of the Bachelor's Programme

Year 2		ECTS
Period 2.1	Databases (KEN2110) Probability and Statistics (KEN2130) Graph Theory (KEN2220) Project 2-1 (*)	4 4 4
Period 2.2	Reasoning Techniques (KEN2230) Machine Learning (KEN2240) Simulation and Statistical Analysis (KEN2530) Project 2-1 (*)	4 4 4
Period 2.3	Project 2-1 (KEN2300)	6
Period 2.4	Mathematical Modelling (KEN2430) Human Computer Interaction and Affective Computing (KEN2410) Theoretical Computer Science (KEN2420) (Elective **) Project 2-2 (*)	4 4 4
Period 2.5	Philosophy & Artificial Intelligence (KEN2120) Linear Programming (KEN2520) Natural Language Processing (KEN2570) Project 2-2 (*)	4 4 4
Period 2.6	Project 2-2 (KEN2600)	6

(*) Project 2-1 will start in period 2.1 and will run until period 2.3 with weekly meetings; Project 2-2 will start in period 2.4 and will run until period 2.6 with weekly meetings. The credits for the projects will become available at the end of period 2.3 and 2.6 respectively.

*** Electives: In case students have passed both electives of period 2.4, Theoretical Computer Science or Introduction to Image and Video Processing can replace 1 of the third year electives.*

For each period, we will give a short explanation of the various parts. Before the start of each period, the students will receive detailed information about the content, the study material, the teaching form, the schedule, and the examination method.

Period 2.1

Databases (KEN2110)

Examiner: Tom Pepels, M.Sc. and Dr. Ashish Sai

Desired Prior Knowledge: Introduction to Computer Science 1 and 2,Data Structures and Algorithms, Software Engineering.

Prerequisites: Description: This course covers the use of (relational) databases and data modelling with the goal of writing (distributed) data-intensive software applications. Specifically, students will learn to use the Structured Query Language (SQL) to manipulate data to develop data-models that are Atomic, Consistent, Isolated and Durable. Moreover, the course covers alternative (distributed) data-storage methods and object persistence techniques such as NoSQL. During the course, students will learn to use different database management systems and how to use them to build software.

Knowledge and understanding: Students will be able to describe the basic concepts of databases, explain the fundamental concepts of database management systems, query languages, data modelling and database programming.

Applying knowledge and understanding: Students will be able to explain the proper database design based on system requirements, indicate possibilities and limitations of database types. In addition, students will be able to combine software architectures to design and construct a database application.

Making judgements: Student will be able to analyze and justify a practical database problem, examine different approaches, and refine database models based on use cases. Moreover, they can make improvements to existing database designs, reflect on certain solutions of the databases design and implementation, and assess the correctness of the database model.

Communication: Students will be able to summarize the basic entities and relationships involved in persistent data, and communicate with developers, database managers and users on proper database design and interfacing.

Learning skills: Students will be able to identify and understand follow-up literature, beyond the teaching material of the course.

Study material: Alan Beaulieu, 2020. Learning SQL, (3rd ed.). O'Reilly Media, Inc.

Recommended literature: Martin Kleppmann, 2017. Designing Data-Intensive Applications. O'Reilly Media, Inc.

Exam: Written exam (75%) + practical assignment (25%)

ECTS: 4

Probability and Statistics (KEN2130)

Examiner: Dr. Christof Seiler and Dr. Gijs Schoenmakers

Desired Prior Knowledge: Discrete Mathematics and Calculus

Prerequisites: None.

Course Description: This course is a first introduction to probability and statistics. We will start by learning how to count and define a notion of probability. We will then move on to the concept of conditional probability, random variables and their distributions, expectation, continuous random variables, moments, joint distributions, and inequalities and limit theorems. This will provide us with the necessary language to study central topics of importance in statistics, such as the difference between a population and a sample, confidence intervals, parameter estimation, and hypothesis testing.

Knowledge and understanding: In this course, the students obtain tools to define random variables and identify probability distributions in a wide range of probabilistic experiments. Furthermore, they know which procedure is most appropriate to find an answer to a given statistical question.

Applying knowledge and understanding: Students are capable of calculating probabilities, expectations, variances and related quantities in a wide range of probabilistic experiments. Furthermore, they can estimate statistical quantities and perform statistical tests to extract information from data sets.

Making judgements: Students are able to critically analyse probabilistic experiments and statistical inferences and are able to decide whether to accept or reject statistical hypotheses.

Communication: The students will be able to communicate their conclusions and the underlying rationale to expert and non-expert audiences.

Learning Skills: Students are able to use elements from probability theory and statistics in other domains in order to increase one's knowledge.

Study material: Hwang and Blitzstein, Introduction to Probability (2019, second edition)

Exam: 20% homework assignments and 80% written final exam

ECTS: 4

Graph Theory (KEN2220)

Examiner: Dr. Matus Mihalák and Dr. Marieke Musegaas

Desired Prior Knowledge: Discrete Mathematics; Data Structures and Algorithms

Prerequisites: None

Description: A graph is simply a collection of points, some of which are joined by lines. This deceptively simple structure is one of the cornerstones of both theoretical and applied computer science. A great many problems that arise in the real world can be modeled as graph problems. Several classical examples include the problem of finding the shortest route between two cities, of maximizing flow in a network of pipelines, or of finding an optimal pairing between producers and consumers. In this course we will look at both the algorithmic/applied side of graph theory and its more abstract mathematical foundations, because the latter is often important for understanding the former. We will cover topics such as paths, tours, trees, matchings, flows and colorings.

Knowledge and understanding: Students will have a solid overview of the basic concepts and results of (applied) graph theory, including the main mathematical tools to argue about graphs. Students will have the tools to model and analyze various real-world problems using graphs.

Applying knowledge and understanding: Students will be able to recognize when a problem can be modeled with graphs, and whether the problem can be efficiently solved using standard or slightly adjusted graph-theoretic algorithms.

Making judgements: Students will be able to formulate a given (sub)problem as a graph-theoretic problem, argue why the formulation is correct, and they will be able to judge the feasibility of existing algorithmic solutions.

Communication: Students will be able to explain, in the language of graph theory, how a problem at hand can be modelled and solved.

Learning skills: Students will enhance their study skills such as time management, effective reading, critical thinking and reading, exact and unambiguous writing and formulation of ideas and statements, and reflection on marked (graded) work. Along the way, students will improve general learning skills such as self-motivation, careful listening and giving instructions, and openness to new knowledge.

Study material: Appropriate material will be provided during the course.

Recommended literature: None.

Exam: Written exam (80% of the final grade). Weekly graded exercises (20% of the final grade).

ECTS: 4

Period 2.2

Machine Learning (KEN2240)

Examiners: Dr. Evgueni Smirnov and Dr. Enrique Hortal Quesada,

Desired prior knowledge: Introduction to Computer Science 1, Calculus, Linear Algebra, Logic, Probability and Statistics

Prerequisites: None

Description: Machine learning is a major frontier field of artificial intelligence. It deals with developing computer systems that autonomously analyse data and automatically improve their performance with experience. This course presents basic and state-of-the-art techniques of machine learning. Presented techniques for automatic data classification, data clustering, data prediction, and learning include Decision Trees, Bayesian Learning, Linear and Logistic Regression, Recommender Systems, Artificial Neural Networks, Support Vector Machines, Instance-based Learning, Rule Induction, Clustering, and Reinforcement Learning. Lectures and practical assignments emphasize the practical use of the presented techniques and prepare students for developing real-world machine-learning applications.

Knowledge and understanding: After successful completion of the course, students will be able to describe and explain the basic machine learning algorithms. Students will understand the mathematical foundation of machine learning algorithms and how mathematical methods are successfully combined to obtain the variety of machine learning algorithms that are currently available.

Applying knowledge and understanding: Students will acquire the knowledge to apply, formulate, and validate techniques from machine learning and to apply basic machine learning algorithms to real-life problems. Students will be able to implement machine-learning algorithms in software and apply existing machine learning software implementation to datasets. Students will have the necessary knowledge to design, implement, and apply data processing systems that autonomously extract information from data, interpret results, and make decisions.

Making judgements: Students learn how to critically analyse real-world problems, select appropriate machine learning techniques according to the specific problem, and predict the consequences of their choices. After successful completion of the course, students gain the ability to judge which problems can be solved better and to which extend through the application of machine learning techniques. Students obtain an awareness of and responsibility for ethical and social consequences of developments in and application of machine learning.

Communication: The skills acquired during the course will allow students to present the results of different stages of the application of machine-learning techniques to specialists or non-specialists.

Learning skills: After successful completion of the course, students can analyse, adapt, design, implement, and critically reflect on machine-learning algorithms and tools. Students also obtain the critical fundamental skills and knowledge to study further advanced machine learning techniques in the professional literature.

Study material: Lecture material provided during the lecture.

Recommended literature:

- T. Mitchell (1997). Machine Learning, McGraw-Hill, ISBN-13: 978-0071154673.
- H. Blockeel, Machine Learning and Inductive Inference (course text), Uitgeverij ACCO, 2012.
- I.H. Witten and E. Frank (2011). Data Mining: Practical Machine Learning Tools and Techniques (Third Edition), Morgan Kaufmann, January 2011, ISBN-13: 978-0123748560.

Exam: Written "open-book" exam at the end of the course. During the course, students receive several graded assignments that can earn them a maximum bonus grade of 1.0.

ECTS: 4

Simulation and Statistical Analysis (KEN2530)

Examiners: Dr. Joel Karel and Dr. Marijn ten Thij

Prior Knowledge: Knowledge: Probability & Statistics, Calculus, Matlab, and Java.

Prerequisite: None.

Description: Mathematical simulation is concerned with studying processes and systems. Uncertainty can be an important factor and must be modelled properly. After modelling a complex system, various scenarios can be simulated, using Monte Carlo simulation, to gain insight. The results need to be properly interpreted and uncertainty has to be reduced. The modelling, implementation, analysis, and technical aspects will be discussed as an introduction in this field. Emphasis will be on discrete event simulation and the statistical analysis of the output of simulation studies, where topics are: modelling, Poisson processes, random number generators, selecting and testing input distributions, generating random variates, experiment design, statistical analysis of experiments, comparing experimental results and variance reduction. Practical exercises will be used to place the techniques in context.

Knowledge and understanding: Define concepts of simulation, discrete event simulation and statistical inference. Explain techniques underlying mathematical simulation. Explain methods for analyzing experimental results and efficient simulation including their assumptions, justify why they are important, and match them to simulation design.

Applying Knowledge and understanding: Being able to model a system in a structured manner, to design and implement simulators for systems, and to collect data from these simulations. In addition, you will be able to employ techniques underlying mathematical simulation and apply methods from statistics for analyzing experimental results.

Making judgements: Being able to choose and motivate alternative techniques underlying mathematical simulation. Choose, motivate, and contrast methods for statistical analysis.

Communication: Being able to convey the phases of a specific simulation study to non-experts. Being able to explain the assumptions and choices made when analyzing experimental data to experts and non-experts.

Learning Skills: The ability to independently learn to handle large-scale simulation. To identify shortcomings in data analysis.

Study material: Simulation Modeling and Analysis (5th edition) -Averill Law

Recommended literature: Object-Oriented Computer Simulation of discrete-event systems – Jerzy Tyszer, Design and Analysis of Experiments – Douglas C. Montgomery, Introduction to Probability Models – Sheldon M. Ross.

Exam: Written exam and assignments and/or bonus assignments.

ECTS: 4

Reasoning Techniques (KEN2230)

Examiners: Prof. Dr. Mark Winands and Dr. Tjitze Rienstra

Desired Prior Knowledge: Introduction to Data Science and Artificial Intelligence; Logic.

Description: Central in this course is how, based on available data, new knowledge and information can be obtained using reasoning processes. The course will be supported by tutorials, in which the acquired techniques can be put into practice by using Prolog. The following four techniques are discussed:

1. Reasoning using logic: syntax, semantics, and inference in first-order logic, situation calculus, forward and backward reasoning, completeness, logic programming with Prolog.
2. Problem solving using search: problem types, blind-search methods, informed-search methods, comparison of search methods, games as search problems, minimax, alpha-beta pruning, Monte Carlo Tree Search, chance games, constraint satisfaction problems.
3. Planning: planning in situation calculus, representation of states, goals and operators, state space and plan space, algorithms for classic planning.
4. Reasoning with uncertainty: uncertainty and probability theory, conditional probability, the Rule of Bayes, semantics of belief networks, exact and approximate inference in belief networks.

Knowledge and understanding: Students learn to understand how problems can be represented as logical problems, as search problems, as planning problems or as problems involving uncertainty and get accustomed to reasoning methods to solve problems of all four types mentioned above.

Applying knowledge and understanding: Students learn to apply the reasoning methods learned to toy problems and some more complex situations.

Making judgements: Students learn to judge which type of knowledge representation is suitable for the problem at hand, and which reasoning technique is suitable to solve the problem at hand. Communication: students can explain the knowledge representation used and reasoning technique chosen to peers and other experts.

Learning skills: Students are able to critically reflect on their own and other's chosen representations and used reasoning methods.

Study material: Russell, S. and Norvig, P., Artificial Intelligence: A Modern Approach, 4th edition. Pearson, 2020. Bratko, I. (2012). Prolog: *Programming for Artificial Intelligence*, 4th edition. Addison-Wesley

Recommended literature: Luger, G.F., Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 6th edition. Pearson International Edition, 2009.

Exam: Closed-book written exam (80%of final grade) and assignments during the course (20% of final grade).

ECTS: 4

Period 2.3

Course title: Project 2-1 (KEN 2300)

Coordinator: Dr. Katharina Schneider

Prerequisites: Students must have passed Project 1-1. Furthermore, the student has to have passed at least two out of the following three courses: Introduction to Computer Science 1, Introduction to Computer Science 2, and Data Structures and Algorithms. This project is a prerequisite for Project 3-1.

Description: Students work on a project assignment in small groups of about six students. The group composition stays the same for the whole project and is announced at the beginning of period 2.1. The students are guided through the project by a fixed tutor. The project assignment is divided into three subtasks (one per period) and is strongly related to the content of the courses from period 2.1 and 2.2. In periods 2.1 and 2.2, the students work on the project, while also having to attend the courses of these periods. They meet their tutor approximately once a week. In period 2.3, the students work three weeks full-time on the project and meet their tutor twice a week.

At the beginning of each period, the students have to hand in a planning for the current phase. At the end of each period, the students have to give a presentation and the source code, presentation and an overview of who did what need to be uploaded to Canvas. While the presentations at the end of period 2.1 and 2.2 are in front of the examiners and the tutors, the presentations at the end of period 2.3 will additionally be in front of the fellow students. In period 2.3, they furthermore have to hand in a report and attend a product and report examination.

Applying knowledge and understanding: Students will learn to concretize project assignment and construct and maintain a planning Furthermore, they will learn formulating, selecting and validating models for the game chosen and collect and interpret experimental data with evaluation metrics. Lastly they will improve their ability to plan and chair meetings, create notes for minutes, work in a team such that the workload is balanced and plan teamwork by setting deadlines and distributing tasks.

Making judgements: After completing this project, students will be able to compare and criticize results, position them in terms of the literature diagnose limitations and formulate a discussion.

Communication: Students will be able to write a scientific paper that: describes the project, explains the methods, summarizes the outcomes, discusses them and makes the conclusions. Students will be able to present and defend project in English and coordinate project progress in project meetings.

Learning skills: Students will be able to reflect on the progress of the project and study relevant literature to solve problem at hand.

Study material: Project manual project 2-1, Maastricht University.

Assessment:
Final Grade = 0.9·(0.15·gradePhase 1+0.15·gradePhase 2+0.7·gradePhase 3)+ skillClassGrade +individualGrade

where gradePhase 3=0.4 gradeProduct+0.4 gradeReport+ 0.1 gradePresentation+ 0.1 gradeProjectManagement.

The *individualGrade* is given due to either outstanding or not enough contribution of a student to the project. By passing skill classes, the students can get a reward called *skillClassGrade*, which is 1 if the students passed all skill classes, 0.5 if the students passed all but one skill classes, and 0 if the students passed all but two skill classes. Failing more than 2 skill classes will lead to an NG in the project. Missing mandatory project events such as project meetings and examination moments will lead to a reduction of the grade or even to receiving an NG for the project.

Skill Classes:

LaTeX (period 2.1)
In this skill class, students learn the basics of LaTeX such as figures, tables, referencing and formulas. At the end of the course, they will be able to write reports and articles using LaTeX.

Information Research: Referencing Tools (period 2.1):
This skill class will give you an introduction to reference tools that will help you to manage your references.

Legal aspects of Data Science (period 2.2)
This project skills class will consist out of two sessions. In the first session, you will get an introduction into the basic principles of the GDPR. By the end of this session, you will be able to grasp the key actors, concepts and obligations of the GDPR, and develop awareness and understanding of the legal requirements you will encounter in your professional career. The general introduction of the aspects of the GDPR is concretized by a number of examples and the session is ended in an interactive Q&A.

In the second session, you will explore the ethical grey zone that exist next to the legal compliance obligations contained in the GDPR. The concepts of digital ethics and accountability are explored along with their limitations. This is done by investigating different real-life cases and scenarios in a lecture.

ECTS: 6

Mathematical Modelling (KEN2430)

Examiner: Dr. Joel Karel and Prof. Dr. ir. Ralf Peeters.
Desired Prior Knowledge: Linear Algebra, Calculus, Matlab.

Description: Mathematical modelling is of great importance for solving practical problems by casting them into a form suitable for the use of mathematical techniques. In this course, a number of basic topics are discussed. First, attention is paid to a framework for mathematical modelling. Then we focus on some widely used model classes from engineering, in particular on the class of linear time-invariant dynamical models. These are described by linear difference equations (in discrete time) or linear differential equations (in continuous time). Alternative model descriptions that are discussed are transfer functions (in the frequency domain) obtained with the z-transform and the Laplace transform respectively; and state-space models, which may or may not involve canonical forms. Some further topics receiving attention are the concepts of stability, sinusoidal fidelity, Bode diagrams, the interconnection of subsystems, and the technique of pole placement by means of state feedback.

The subject matter is clarified through exercises and examples involving practical applications. Also, relevant functionality in Matlab is introduced, which offers a powerful instrument for analysing linear dynamic models.

Knowledge and understanding: Being able to formulate linear dynamical models, state properties and define representations. Identify frequency domain properties of systems and relate them to applications in signal processing.

Applying knowledge and understanding: Being able to construct elementary mathematical models. Perform model analysis and extract model properties. Employ various model representations and choose the most appropriate one. Compute state-feedback control.

Making judgements: To recognize what are the important aspects to consider when building a mathematical model. Decide on stability of models.

Communication: Being able to convey properties of models to specialists and non-specialists.

Learning skills: Being able to independently find Matlab functionality to solve basic problems in systems theory.

Study material: Lecture notes.

Recommended literature: Richard J. Vaccaro, Digital Control: A State-Space Approach, McGraw-Hill, 1995, ISBN 0-07-066781-0.

Exam: Written exam and assignments and/or bonus assignments.

ECTS: 4

Contents
Education
Bachelor
Master
Facilities
Staff

Human Computer Interaction and Affective Computing (KEN2410)

Examiner: Dr. Konstantia Zarkogianni

Desired Prior knowledge: Machine Learning, Probabilities and Statistics.

Description: Human-Computer Interaction (HCI) is the study of interaction between people (users) and computers. It is often regarded as the intersection of computer science, behavioural sciences, statistics, design, and several other fields of study. It involves multiple aspects, such as business and user requirement extraction, prototyping techniques, design analytics, descriptive and inferential statistics, as well as the right application of research methods in designing interfaces and testing with users. This course also covers Affective Computing, a new branch of HCI that places emphasis on making user emotions and personality part of computational models. Affective Computing attempts to bring emotions into intelligent interfaces that interact with humans and see how they can have a positive and constructive impact in human-machine interactions.

Knowledge and understanding: The course shows guidelines for the design, implementation and evaluation of HCI systems and addresses interaction styles with the user. Students will be able to identify ways to involve user emotion and cognition in the design of an HCI intelligent interface.

Applying Knowledge and understanding: Students will be able to apply stages of successful user interface design. Formulate user interview techniques, rapid prototyping, design interfaces and conclude on user needs in business-like scenarios.

Making judgements: Students will show awareness of how to involve the user in the design procedure and solve HCI problems based on usability studies and data analytics.

Communication: Students will be able to present their prototypes, communicate and defend their results. They will be able to examine existing interfaces and they will ground their findings based on HCI principles and inferential statistics.

Learning skills: Students will be able to describe a course of action for designing human-centric systems, applicable in a variety of social and business practices. They will be able to establish links among Machine Intelligence, Affective Interactions and Statistical analyses in the design of user-centered interfaces. They will be able to use these skills in constructing personalized solutions, quantitatively analyse and apply user needs, design emotion-capturing techniques by using computational models of affect.

Study material: Lecture Slides and other sources that will be made available.

Recommended literature:

- Shneiderman B, Plaisant C, Cohen M, Jacobs S, Elmqvist N, Diakopoulos N. (2016) Designing the user interface: strategies for effective human-computer interaction. Pearson, ISBN: 978-0134380384
- Calvo RA, D'Mello S, Gratch JM, Kappas A, (2015). The Oxford handbook of affective computing. Oxford University Press, ISBN: 978-0199942237
- Coursera video lectures of Scott Klemmer and accompanying slides.

Exam: Individual (90% of final grade) and group (10% of final grade) assignments.

ECTS: 4

Theoretical Computer Science (optional course) (Code: KEN2420)

Examiner: Dr. Georgios Stamoulis

Desired Prior Knowledge: Introduction to Data Science and Artificial Intelligence, Discrete Mathematics, Data Structures and Algorithms.

Description: This course explores the theoretical underpinnings of computing by investigating algorithms and programs casted as language recognition problems. The influence of the theory on modern hardware and software system design is demonstrated. The following subjects will be treated: mathematical foundations, alphabets and languages, finite automata and regular languages, Turing machines, acceptance and decidability, recursive functions and grammars, time complexity classes, NP problems, NP-completeness

Knowledge and understanding: Students will learn to comprehend the inherent complexity of problems and be able to motivate why some problems are inherently more difficult than others are. They will learn to have insight into how complex problems can be solved efficiently and will be able to classify such problems into a language hierarchy and complexity classes. Furthermore, students will be able to apply the tools needed for such classification

Applying knowledge and understanding: students will be able to apply the theory learned to solve small-scale problems

Making judgements: Students will learn to judge which problems are decidable and efficiently solvable and to judge which technique is suitable to solve the problem at hand.

Communication: The knowledge representation used and technique from complexity theory chosen must be easily understandable by peers and others experts

Learning skills: The student will learn to reflect on own one’s and other’s thoughts on complexity and solvability of problems.

Study material: Elaine Rich (2008), Automata, Computability and Complexity, Prentice Hall, New Jersey, ISBN 0-13-228806-0.

Exam: Written exam; during the course the students will receive three assignments, that, if they receive a sufficient grade, may earn them up to a total of one bonus point.

ECTS: 4

Introduction to Image and Video Processing (optional course) (KEN3238)

Examiner: tba

Desired prior knowledge: Calculus, Linear Algebra, Machine Learning.

Prerequisites:None.

Course Description: Image and video processing is everywhere around us, in smartphones, robotics, medicine, security systems, microscopy, remote sensing, video games, travel, shopping, environmental management and many other applications. Image and video processing is based on principles of signal processing, extended to multiple dimensions. In this class students will have a short introduction to basic 2D signals and systems, sampling, convolution. Color domain processing in different spaces and its relevance to our visual perception system will be presented. We will learn about linear and non-linear filtering in the spatial and frequency domains (Fourier, DCT), their relation and applications like enhancement, noise estimation and removal, compression, restoration. Compression standards like JPEG and their links to frequency transformations will be presented. Video analysis will include methods for motion estimation, segmentation and introduction to action recognition, and video standards. Deep learning methods will be connected to the basics of image processing and some methods will be introduced. Solved math and programming will accompany the classes.

Knowledge and understanding: Students will gain an in-depth knowledge of image and video processing methods used all around us by understanding the underlying foundations and applying methods themselves. They will be able to solve more complex image/video processing problems through the provided examples, links to online resources including practical examples, and their projects that the course assessment is based on.

Applying knowledge and understanding: The students will be able to immediately apply basic image and video processing concepts to real world problems. They will also have the opportunity to further demonstrate their understanding and creativity in problem-solving, as well as professional presentation by implementing them in mini projects.

Making judgements: By understanding fundamental principles and seeing them work in practice, the students will be able to understand how and where to implement and develop image and video analysis algorithms, as well as build complex systems using and extending them.

Communication: Assessment will be based on mini projects with topics on image/video processing chosen by the instructor (the students themselves are also free to suggest subjects relevant to the class). The goal will be for them to learn to carry out independent work, solve realistic problems based on the class material, as well as effectively communicate their motivation and results both to a general and expert audience by providing interesting demos, presentations, and clearly structured reports.

Learning Skills: Students will be able to carry out basic image and video analysis algorithms, understanding and implementing the mathematics behind them. They will solve small research questions within mini projects and will become familiarized with the newer toolboxes and libraries, mostly in – but not restricted to - Matlab and Python.

Study material: Lecture slides and provided material. Gonzalez & Richard E. Woods, "Digital Image Processing ".

Recommended literature: Computer Vision: a Modern Approach. D. A. Forsyth, J. Ponce (online).

Exam: Two mini projects (100%).

ECTS: 4

Period 2.5

Philosophy & Artificial Intelligence (KEN2120)

Examiner: Dr. Robert Gianni

Prerequisites: none.

Description: One of the characteristics of scientific knowledge is the translation of natural phenomena into quantitative or mathematical data – the book of nature, Galileo wrote, is written in the language of mathematics. Over the course of the twentieth and twenty-first century, this desire to understand the world through the logic of mathematics has been extended beyond the natural world to include such things as human consciousness, learning, and intelligence. Indeed, the foundation of what is called 'artificial intelligence' is the pursuit of replicating human consciousness and intelligence through mathematical models and formulas. In this course we will examine these issues from a philosophical perspective, beginning with a basic overview of the philosophy of science with an emphasis on quantification and then moving on to study philosophical issues that have developed out of the pursuit of artificial intelligence. We will begin with classic thinkers in the field like Alan Turning, Hubert Dreyfus, and Joseph Weizenbaum and continue through to contemporary philosophical studies of cutting edge attempts to develop types of machine learning that aim to mimic human forms of learning.

Knowledge and understanding: At the conclusion of this course, students should be able to demonstrate knowledge of the following topics through written essays:
- The history of computing and artificial intelligence
- The history and philosophy of scientific knowledge with an emphasis on Kuhn's theories of scientific paradigms
- Historical and philosophical theories of technology and society
- The philosophical presuppositions of artificial intelligence

Applying knowledge and understanding: Students will be able to draw upon both lectures and readings to write an essay that exhibits critical reflections on conventional and naïve notions of instrumentalism, technological determinism, and functionalism by persuasively arguing for a contextual approach that highlights the contingency and flexibility of design and meaning.

Making judgements: Students are asked to select relevant passages from texts that contribute to the argument that they make in the essay. This will be graded. In tutorials, students are asked to make decisions about specific problems (i.e. self-driving cars, Turing tests). This is not graded.

Communication: During tutorials, students present their work orally to their classmates.

Learning Skills: Students will be able to articulate and solve problems in groups. Students will also be expected to engage with a number of theories concerning computation and artificial intelligence through different texts and will be asked to reflect upon and critique these theories.

Study material: Selected texts will be made available.

Exam: Take home exam (essay).

ECTS: 4

Linear Programming (KEN2520)

Examiner: Dr. Steven Kelk

Desired Prior Knowledge: Linear Algebra.

Prerequisites: None

Description: A linear program is very different to, say, a Java program. It simply consists of a linear objective function (of potentially very many variables) and a set of linear inequalities. The goal is to find values of the variables, which maximize or minimize the objective function, subject to all the inequalities being satisfied. Linear programs - even very large linear programs - can be solved extremely quickly, in both theory and practice. The model is also expressive enough to capture a large number of real-world problems. These two factors explain the fundamental role of linear programming in operations research, computer science, economics, management and many other fields. The course consists of an in-depth study of the simplex algorithm (a standard algorithm for solving linear programs), duality theory, and sensitivity analysis. Examples from practice illustrate the power of the model and teach the student the skill of modelling. Practical aspects of linear programming (e.g. use of software packages for solving linear programs, and integration with languages such as Java) are also considered.

Knowledge and understanding: Students will be able to identify which real-world optimization problems can be formulated as linear programs. Students will be able to describe the mathematical foundations of the Simplex algorithm for solving linear programming, and articulate how these foundations impact upon the performance of the Simplex method in practice. Students will recognize the power and importance of duality theory for reasoning about the behaviour of linear programs (in particular with regard to sensitivity analysis). Students will be able to exhibit an awareness of non-Simplex paradigms for solving linear programs (interior-point methods) and be able to recount the importance of the linear programming model in operations research and applied mathematics.

Applying knowledge and understanding: Students will be able to 1) translate mathematical models into linear programs, 2) to apply the Simplex method by hand to solve small linear programs, 3) to show how the Simplex method behaves in normal and exceptional cases, 4) to manipulate the algebra underpinning the Simplex method, 5) to combine insights from this algebra and primal-dual relations to make rigorous statements about the (sub)optimality of solutions to linear programs, 6) to argue how small changes to linear programs impact upon their optima (sensitivity analysis), 7) to explain key differences between the Simplex method and interior-point methods, and to 8) leverage linear-to program arguments when developing simple algorithms for combinatorial optimization problems.

Making judgements: Students will be able to distinguish between mathematical models that can and cannot be cast as a linear program. Students will be able to contrast and compare the behaviour of the Simplex algorithm with interior-point methods. Students will be able to select, out of a large range of algebraic and duality-based instruments, appropriate tools for making rigorous statements about linear programs.

Communication: Students will be able to formulate linear programs and defend their correctness. Students will be able to clearly articulate and defend algebraic and duality-based arguments concerning linear programs.

Learning Skills: By the end of the course, students will be able to autonomously and critically reflect upon the appropriateness of the linear programming paradigm for tackling optimization problems arising in practice and be able to assess the correctness of mathematical arguments pertaining to linear programming. Students will be able to identify follow-up literature, which goes beyond the scope of the material presented in the course.

Study material: Hillier & Lieberman (2010 or 2015): Introduction to Operations Research (9th or 10th edition). McGraw Hill, ISBN 978-007-126767-0 or ISBN 9781259162985. Support for the 11th edition is forthcoming.

Recommended literature: students are beforehand encouraged to refresh their knowledge of: (unique) solutions of systems of linear equations, matrix inversion, and matrix rank.

Exam: Written exam and optional bonus exercises (the results of which are added to your exam score, up to 10%).

ECTS: 4

Natural Language Processing (KEN2570)

Examiners: D): Dr. Jerry Spanakis and Dr. Aki Härmä

Tutors(s): None.

Desired Prior Knowledge: Introduction to Computer Science 1 and 2, Probability and Statistics, Machine Learning

Prerequisites: None

Description: ChatGPT can answer almost any question you have. Siri can tell me when I need an umbrella. But how do they work? Over the past few years, Natural Language Processing (NLP) was revolutionized by statistical, probabilistic and machine learning methods. NLP addresses fundamental questions at the intersection of human language and machine learning. How can computers acquire, understand and produce language? How can computational methods give us insight into observed human language phenomena? How to make sense of the vast amounts of information available online in free, unstructured form? In this course students will learn how computers can learn useful text/language representations and how different tasks (language modelling, text classification, information extraction, sequence labeling, etc.) can be used for solving different complex problems (spelling correction, spam detection, search engine design, opinion analysis, summarization, question-answering, etc.). Open NLP problems (such as evaluation or interactive dialogue systems) and the effect of deep learning on NLP will be discussed.

Knowledge and understanding: By the end of the course, students are able to acquire the basic text and language processing aspects. Furthermore, students are able to describe basic NLP problems, tasks and methods.

Applying knowledge and understanding: Students are able to demonstrate how to tackle a text/language problem and to formulate, design and implement a NLP system. Students are able to suggest when a problem's complexity requires an NLP solution.

Making Judgements: Students are able to pose questions and define problems in different domains (e.g. social sciences) and contexts (e.g. business) that include language/text data. Furthermore, students are able to judge which tools are applicable for solving these problems and to decide a course of action in accordance with ethical and social consequences.

Communication: Students are able to outline an approach in real organizational problems, which require NLP and are able to demonstrate, present and communicate a solution to a NLP problem Learning Skills: Students are able to master and choose the appropriate basic programming tools for NLP and are able to follow up on literature that will allow them to build complete NLP models.

Study material: Handouts

Recommended Literature:
- Daniel Jurafsky and James H. Martin. "Speech and language processing an introduction to natural language processing, computational linguistics, and speech." Pearson, London, 2000
- Chris Manning and Hinrich Schütze, Foundations of Statistical Natural Language Processing, MIT Press. Cambridge, MA. 1999

Exam: Practical individual assignments (20%) + Group Project (30%) + Open-Book Written Exam (50%).

ECTS: 4

Course title: Project 2-2 (KEN 2600)

Coordinator: Dr. Katharina Schneider

Prerequisites: Students must have passed Project 1-2. Furthermore, the student has to have passed at least two out of the following three courses: Introduction to Computer Science 1, Introduction to Computer Science 2, and Data Structures and Algorithms. This project is not a prerequisite for another project / course.

Description: Students work on a project assignment in small groups of about six students. The group composition stays the same for the whole project and is announced at the beginning of period 2.4. The students are guided through the project by a fixed tutor. The project assignment is divided into three subtasks (one per period) and is strongly related to the content of the courses from period 2.4 and 2.5. In periods 2.4 and 2.5, the students work on the project, while also having to attend the courses of these periods. They meet their tutor approximately once a week. In period 2.6, the students work three weeks full-time on the project and meet their tutor twice a week.

At the beginning of each period, the students have to hand in a planning for the current phase. At the end of each period, the students have to give a presentation and the source code, presentation and an overview of who did what need to be uploaded to Canvas. While the presentations at the end of period 2.4 and 2.5 are in front of the examiners and the tutors, the presentations at the end of period 2.6 will additionally be in front of the fellow students. In period 2.6, they furthermore have to hand in a report and attend a product and report examination.

Applying knowledge and understanding: Students will learn to concretize project assignment and construct and maintain a planning. Additionally, they will learn formulating, selecting and validating models for a concrete problem at hand and to collect and interpret data with evaluation metrics. Lastly they will improve their ability to plan and chair meetings, create notes for minutes, work in a team such that the workload is balanced and plan teamwork by setting deadlines and distributing tasks.

Making judgement: After completing this course successfully, students will be able to compare and criticize results, position them in terms of the literature; diagnose limitations and formulate a discussion

Communication: Students will be able to write a scientific paper that: describes the project, explains the methods, summarizes the outcomes, discusses them and makes the conclusions. Furthermore, student will be able to present and defend project in English. Coordinate project progress in project meetings

Learning skills: Students will learn to reflect on the progress of the project and study relevant literature to solve problem at hand

Study material: Project manual project 2-2, Maastricht University.

Assessment:
Final Grade = 0.9 · (0.15 · gradePhase 1+0.15 · gradePhase 2+0.7 · gradePhase 3)+ skillClassGrade +individualGrade

where gradePhase 3=0.4 gradeProduct+0.4 gradeReport+ 0.1 gradePresentation+ 0.1 gradeProjectManagement.

The *individualGrade* is given due to either outstanding or not enough contribution of a student to the project. By passing skill classes, the students can get a reward called *skillClassGrade*, which is 1 if the students passed all skill classes, 0.5 if the students passed all but one skill classes, and 0 if the students passed all but two skill classes. Failing more than 2 skill classes will lead to an NG in the project. Missing mandatory project events such as project meetings and examination moments will lead to a reduction of the grade or even to receiving an NG for the project.

Skill Classes:

Academic Writing (period 2.4)
Students will receive written feedback on their project report draft.

Academic Presentation (period 2.5)
In this skill class, the project groups will practice their presentation and will receive feedback on the presentation style, slide layout, pronunciation, and language.

ECTS: 6

2.3 Curriculum of the Third Year of the Bachelor's Programme

The first semester of the third year allows you to make your own selection of subjects in the field of artificial intelligence, data science, applied mathematics and computer science, the core areas of the study of Data Science & Artificial Intelligence. In each of the periods 1 and 2, you choose 3 out of 6 optional courses. The first semester of year 3 has the same structure in the first and the second year; there are two periods of eight weeks and one period of four weeks. There is also a project in period 3.3. Alternatively, students can choose to study the first semester of the third year at a partner university abroad. Please check the Study abroad guide for more info: https://intranet.maastrichtuniversity.nl/en/dacs-students/going-abroad/study-abroad-guide

Other options will be uploaded by the student counsellors.

Year 3		ECTS
Period 3.1*	Recommender Systems (KEN3160) Semantic Web (KEN3140) Game Theory (KEN3130) Robotics and Embedded Systems (KEN3236) Digital Society (KEN3111) Project 3-1 (**)	4 4 4 4 4
Period 3.2*	Large Scale IT and Cloud Computing (KEN3239) Logic for Artificial Intelligence (KEN3231) Parallel Programming (KEN3235) Computer Security (KEN2560) Introduction to Bio-Informatics (KEN3440) Software and Systems Verification (KEN3150) Introduction to Quantum Computing (KEN3241) Project 3-1 (**)	4 4 4 4 4 4 4
Period 3.3	Project 3-1 (KEN3300)	6
Period 3.4	Data Analysis (KEN3450) Operations Research Case Studies (KEN3410) Intelligent Systems (KEN3430)	4 4 4
Period 3.5-3.6	Bachelor's thesis (KEN3500)	18

(*) Third year students choose three electives per period out of the optional courses during period 1 and 2

*** Project 3-1 will start in period 3.1 and 3.2 with weekly meetings. The credits for the project will become available at the end of period 3.3.*

For each period, we will give a short explanation of the various courses. Before the start of each period, the students will receive detailed information about the content, the study material, the teaching form, the schedule, and the examination method.

Semantic Web (optional course) (KEN3140)

Examiner: Prof. Michel Dumontier

Desired Prior Knowledge: Logic.

Description: Most of the information available on the World Wide Web (WWW) is not directly understandable for computers. For instance, web pages are designed for human readability. Computer programs have difficulty in interpreting the information presented on web pages. The focus on human readable information introduces restrictions on what computer programs can do to support human users in tasks such as:

- finding information
- buying goods
- making travel plans

The Semantic Web should eliminate these restrictions by separating the content of what is presented on a web page from the way it is presented. In recent years, the focus has shifted to providing data, independent of webpages (for example: Linked Open Data (LOD). Ontologies are used to provide a shared conceptualization of information. Ontologies form the basis of the Semantic Web, Knowledge Based System, Databases, etc., and they play an important role in data exchange and interoperability in many domains. Ontologies are applied in the bio-medical domains, in data mining applications, in Linked Open Data (LOD), in websites based on semantic technology, etc. Since ontologies are intended to be shared between different systems, defining an ontology is a challenging task.

This course will focus on the standards the World Wide Web Consortium (W3C) is defining in order to realize the Semantic Web. The course also addresses the underlying knowledge representation formalisms of the current semantic web standards. Moreover, the course will address the engineering principle of crating an ontology. Note that the course does not address standards for making websites.

Knowledge and understanding: Making the student familiar with the developments and standards of the Semantic Web. The student will get insights in semantic web standard, such as RDF, RDFa, SPARQL and OWL2. Moreover, the students will get some basic insight in the semantics of RDF and the Description Logic underlying OWL. Finally, the student will be made familiar with the ontology development process, and criteria for evaluating an ontology. The student should understand the role of upper ontologies and ontology design patterns, as well as the philosophical choices they represent.

Applying knowledge and understanding: The student should be able to build applications using semantic web standards such as RDF, RDFa, SPARQL and OWL2. The student should also be able to develop an ontology for an application domain.

Making judgements: The student should be able to judge whether and how semantic web standards can be applied in applications. The student should also be able to judge the quality of an ontology.

Communication: The student should have sufficient understanding of the Semantic Web and its standards in order to explain why and how an application should be set up using semantic web standards. The student should also be able to explain and defend the choices made in the ontology engineering process.

Contents
Education
Bachelor
Master
Facilities
Staff

Learning skills: The student should be able to study the literature about semantic web developments.

Study material:
- A Semantic Web Primer, Grigoris Antoniou, Paul Groth, Frank van Harmelen and Rinke Hoekstra, MIT Press, ISBN: 9780262018289 (third edition).
- Syllabi and scientific papers about ontology engineering.

Recommended Literature: The documents on the site of the World Wide Web Consortium (W3C).

Examination: Practical exercises and a written exam at the end of the course. The grade is for 70% determined by the written exam and for 30% by practical assignments. Participation in the practical is required for receiving a grade.

ECTS: 4

Game Theory (optional course) (KEN3130)

Examiner: Prof. Dr. Frank Thuijsman

Prerequisites: Discrete Mathematics, Linear Algebra.

Description: We introduce the field of Game Theory. Game Theory is the mathematical study of problems, called games, that involve two or more decision makers, called players, who each have their own individual preferences over the possible outcomes. In a game, each player always aims to maximize his individual payoff and chooses his actions accordingly. These actions may be probabilistic or deterministic, depending on the situation. Meanwhile he reasons logically about actions that might be taken by the other players. A basic difference exists between strategic and non-strategic models. Both types of models and their solution concepts will be discussed. Issues like value, fairness, manipulations, threats, optimality and rationality will be addressed.

Knowledge and understanding: Students can recognize and classify the main types of games, i.e. cooperative games, strategic games, bipartite matching problems, and formulate the main solution concepts value, optimal strategies, Nash- and correlated equilibrium, as well as a number of algorithms to calculate these.

Applying the use of knowledge and understanding: Students can calculate solutions for the different types of games

Making Judgements: Students can explain advantages and disadvantages of different solution concepts. They are able to judge the correctness of solutions presented Communication: Students can explain and defend the correctness of their solutions

Learning Skills: By the end of the course, students will be able to autonomously and critically reflect upon the pros and cons of different types of games for modelling competition and cooperation. This includes considerations on the computational aspects with respect to different solution concepts.

Study material: Lecture notes.

Examination: There will be a closed book written exam at the end of the course.

ECTS: 4

Robotics and Embedded Systems (optional course) (KEN3236)

Examiner: Dr. Rico Möckel.

Desired prior knowledge: Calculus, Linear Algebra, Machine Learning.

Prerequisites: Introduction to Computer Science 1 and 2.

Description: Nowadays, a variety of products require that algorithms from data science and artificial intelligence are adapted to and implemented in robotic and embedded systems. Applications that heavily rely on intelligent robotic and embedded systems include self-driving cars, autonomous drones, intelligent industrial robots in (semi-) autonomous factories, smart phones, intelligent medical devices, and distributed intelligent embedded devices in smart homes.
In this course, students receive an introduction to the fields of robotics, embedded systems, and real-time control. Students obtain an overview of state-of-the-art intelligent robotic and embedded systems in academia and industries. Students gain hands on experience in programming embedded robotic systems using embedded processors and a modular robotic system developed at the Department of Advanced Computing Sciences. Students learn about communication standards for embedded systems, sensors, and actuators. Student practise and strengthen their expertise in data science and knowledge engineering by applying mathematical methods for controlling robotic systems: They study control techniques including PID control, forward and inverse kinematics as well as locomotion control and learning using central pattern generators. The course concludes with a robot competition where students build and program robots using a modular robotic system.

Knowledge and understanding: Students obtain knowledge in designing, building, and programming robotic and embedded systems. Students learn how to apply mathematical concepts like dynamic systems for controlling robotic systems in real-time. Students further obtain knowledge about sensors and motor control and study the application of machine learning and mathematical methods for learning and optimizing control parameters. Students receive training in the programming language C - the most popular languages for programming microcontrollers.

Applying knowledge and understanding: After successful completion of this course, students can analyse, apply, implement, and validate control techniques in embedded and robotic systems with and without real-time constraints. Students can apply techniques from machine learning, search, and optimisation to obtain parameters for embedded control systems as required in many professional academic and industrial applications.

Making judgements: Students learn to judge where real-time systems are required and embedded systems can be beneficial. Students further learn to critically analyse the use of robotic systems in a variety of scenarios and to make design choices for robotic and embedded systems. By introducing students to a variety of state-of-the-art robotic systems, the course lays the foundation so that students can process professional literature in robotics and embedded systems.

Communication: Students will be able to 1) discuss robotic and embedded systems professionally and critically, 2) plan, discuss, implement, and validate projects in robotics and embedded, 3) present the results of project assignments in form of video, and to 4) critically analyse and explain control techniques for robotic systems to a general and professional audience.

Learning Skills: Students are able to autonomously and critically reflect upon the abilities and limitations of robotic and embedded systems in order to keep up with new developments in the field. Students can further assess the capabilities and limitations of their own solutions to a control or machine learning problem in robotics, and to identify follow-up literature, which goes beyond the scope of the material presented in the course.

Study material: Course material will be provided during the lectures.

Exam: The final course grade is 80% of the final written "closed-book" exam grade plus 20% of the assignments grade.

ECTS: 4

Digital Society (optional course) (KEN3111)

Lecturer & Examiner: Dr. T. Frissen

Prerequisites: none.

Desired prior knowledge: none.

Description: Digitalization has a profound impact on our society. We can observe changes in different areas. What digital technologies do, what they look like and how they relate to each other is not identical worldwide, but dependent on local practices as well. Usually new technologies are understood as innovation and progress: and indeed, digital technologies improve a broad range of domains, such as healthcare or education. New possibilities as e.g. participation in our digital cultures arise but also new inequalities, as the access and competences needed for participation are not evenly distributed and the platforms that allow for participation also harbour new mechanisms of control and surveillance. The pace and diversity of these developments ask for continuous investigation and reflection. It requires work to shape and use technologies in ways that contribute to the public good. Moreover, digital technologies have also led to highly problematic developments such as electoral manipulation, fake news and algorithmic discrimination.

Technological developments are often conceived as predefined or given. Does a society's technology drive the development of its social structure and cultural values? Scholars in science and technology studies have shown that technology and society are deeply intertwined. Technology is inherently social. Technologies are shaped by people; they emerge and are embedded in social practices.
The aim of this course is to investigate the consequences of digitalization for our society/societies. These consequences have been differently valuated: participation vs. exploitation of users, innovation as enhancement vs. challenge, ethics and techno-moral change vs./and sustainability.
We will discuss digitalization from

- a social perspective when we read about digital participation and how technology and society are intertwined
- a political perspective when we discuss activism, digital citizenship but also problems of manipulation and verification (as in the case of fake news and deep fakes)
- a cultural perspective when we analyze imaginaries and discourses around innovation of technology and promises being made
- a legal perspective when we discuss privacy and the attempts to adapt privacy laws
- an ethical perspective when we discuss design decisions, privacy but also techno-moral change and questions of environment and sustainability.

The course is structured in the following way:
Transformations
(digital participation, digital citizenship, data-activism)

Imaginaries
(innovation and techno-moral change)

Disruptions
(fake news and deep fakes, sustainability and e-trash)

Knowledge and understanding: Students acquire knowledge on the impact of digitalization on society.

Applying knowledge and understanding: Students learn to understand the interrelation between digital technology and sociality.

Making judgements: Upon completion of the course, students can reflect on ethical challenges related to digitalization.

Communication: Students are able to communicate central topics related to digitalization to an audience of non-IT-experts (e.g. the debate will bring students from the FASoS Bachelor Digital Society and students from the bachelor's Data Science and Artificial Intelligence following this course together to train both groups to communicate topics related to digitalization from a social science and IT perspective).

Learning Skills: Students are able to reflect critically in written form on a topic related to the digital society but also to do so orally in a presentation and debate.
Study material: The literature (provided via the reference list of the library).

Exam: (group) presentation in class (1-3 students) per task (25% of the final grade), 2 short academic papers of 1500 words each (2x25% of the final grade) and participation in a final debate (25% of the final grade).
If a resit is needed for the (group) presentation, the presentation will be given via video (e.g. Zoom or Skype). If a student needs to resit the papers they can be rewritten and improved based on the comments of the tutor. If a resit is needed for the debate (in case a student does not show or participate in the debate), the student can write a 1500-word paper on the content of the debate instead.

ECTS: 4

Recommender systems (KEN3160)

Examiner: Prof. Dr. Nava Tintarev. and Dr. Francesco Barile

Required Knowledge: Machine Learning

Desired Prior Knowledge: Natural Language Processing, Human Computer Interaction & Affective Computing

Recommender systems play an important role in helping to mediate many of our everyday decisions and choices, including the music we listen to, the news that we read, and even the people that we date. They do this by learning from our past interactions, inferring our interests and documenting our preferences. To make the right suggestions at the right time recommender systems must not only understand our preferences but also our current needs and perhaps our immediate intent. Thus, the core focus of most recommender systems is devoted to profiling users and matching items based on these profiles and current context.

Much of the research to date on recommender systems has focussed on the engineering and evaluation of core recommendation algorithms. Researchers have developed a variety of approaches to harness different forms of preference data in the pursuit of more accurate recommendations. For example, researchers have used simple ratings for collaborative, rich meta-data for content-based methods, and even the opinions and sentiment expressed within user-generated reviews.

When evaluating recommender systems, there has been a heavy emphasis on measuring the accuracy of suggestions, or the error of predictions. However, in practice it is important to consider evaluation metrics beyond accuracy, such as diversity, novelty, and serendipity. This in turn has led to increased attention being given to the nature of the interactions between users and recommender systems, and the influence that the user interface and interaction style can have on user behaviour and the overall recommendation experience. This course focuses on:

- Non-personalized and Stereotype-based Recommender Systems
- Classical recommender systems algorithms, e.g., Content-based Filtering, Collaborative-based Filtering
- Offline Evaluation e.g., protocols, criteria, metrics
- User-centered evaluation
- Interfaces and interaction in Recommender systems, e.g., explanations and conversational recommender systems
- Group Recommender Systems
- Ethics, bias, and fairness in recommender systems
- Advanced methods, e.g., Matrix Factorization, Hybrid recommender systems, Contextual Recommender systems

Knowledge and understanding: Students will be able to explain concepts from recommender systems, such as the difference between different recommendation methods and can identify advantages and limitations of these methods. Students will also be able to explain one advanced method or topic suitable for progression to a Master level program in Data Science or Artificial Intelligence.

Applying knowledge and understanding: Students will be able to apply ideas, methods, and tools for recommender systems that are suitable for a given domain. Students will be able to solve problems and design analytically, to comprehend (design) problems and abstract their essentials, to construct and develop logical arguments with clear identification of assumptions and conclusions. Students will develop the ability to transpose academic knowledge and expertise into (inter)national societal, professional and business contexts.

Making judgements: Students will gain acquaintance with the standards of academic criticism. Students will develop an awareness of, and responsibility for ethical, normative and social consequences of developments in science and technology, particularly resulting from Data Science and Artificial Intelligence.

Communication: Students will develop academically and internationally appropriate communicative skills, i.e., the ability to give effective oral presentations, both formally and informally, and understand and offer constructive criticism of the presentations of others.

Learning skills: Students will be able to reflect on their own working methods, and own readiness to take the necessary corrective action.

Study material: Course notes, required reading of scientific articles.

Recommended Literature: Jannach, Dietmar, et al. Recommender systems: an introduction. Cambridge University Press, 2010. Additional research papers and online articles

Exam: The assessment is composed by three components: Individual “Review” of Scientific Papers (15%), Practical group assignment (30%), and Written exam (55%). The three components are mandatory (e.g., A grade of at least 5.5 for each component is necessary in order to pass the exam).

ECTS: 4

Period 3.2

Computer Security (optional course) (KEN2560)

Examiner: Dr. Bastian Küppers

Tutor: Dr. Bastian Küppers

Desired Prior Knowledge: Introduction to Computer Science (1 & 2), Data Structures and Algorithms, Software Engineering, Databases

Description: Computer security is the process of securing information systems against unauthorized access. As information systems have become mandatory in the modern world, coupled with the increased frequency of security incidents, organizations now recognize the need for a comprehensive security strategy. The course will introduce a wide range of topics in computer security and online privacy. The main objective of the course is to cultivate a security mind set by discussing various attack techniques and defenses. The topics we will explore are information security (cryptography, cryptoanalysis), software security and network security, as well as designing secure systems. The class will consist of lectures in which several computer security issues will be discussed. In parallel, there will be bonus assignments where the students will have to solve some of the most important issues we discussed in classroom.

Knowledge and understanding: Students will be able to explain the fundamental concepts of computer security, such as the principles of a secure system and the potential attacks that can compromise it. They will also learn to recognize the various aspects that play a key in applying those principles to real world scenarios, and avoid common mistakes that can introduce vulnerabilities to their systems.

Applying Knowledge and understanding: Students will be able to apply computer security fundamentals to real world scenarios.

Making judgements: By understanding the fundamentals of computer security and by realizing their assignments, the students will be able to understand and avoid mistakes when designing a system. In principle, after completing this course, they will be able to design and develop secure systems on their own.

Communication: Students will be able to explain the principles of computer security to specialists and non-specialists. They will be able to explain why the design of a system is secure or not and how the system can be improved.

Learning skills: Students will be able to read and interpret scientific literature on computer security that goes beyond the scope of the course, and independently design and implement secure systems for real-world applications.

Study material: material provided electronically on Canvas/digital learning environment.

Recommended literature:
- J. Buchmann. Introduction to Cryptography. Springer.
- Tanenbaum & Bos. Modern Operating Systems (4th edition). Pearson.

Exam: Assignments and Project.

ECTS: 4

Large Scale IT and Cloud Computing (KEN3239)

Examiner: Dr. T. Eifert, Dr. Bastian Küppers

Desired Prior Knowledge: Introduction to Computer Science 1, Databases

Prerequisites: none

Description: The course offers a comprehensive introduction to the field of scalable IT systems, so-called "Big IT", and cloud computing. After a technical introduction to the available methodologies of setting up and running scalable systems, use cases are presented. These use cases emphasize the correlation of the processes and requirements of large institutions and possible technical solutions. A special focus is put upon the question which technological platform is best used for which use case as well as process aspects of scaling. Security aspects specific to cloud computing are discussed along the use cases. Cloud computing, as a special case of scalable IT, is discussed in detail. Different cloud providers are presented and evaluated in the context of university requirements, i.e. requirements posed by research and teaching processes.

Knowledge and understanding: students acquire an overview of existing technologies for scalable systems, and specific security requirements for the different use cases.

Applying knowledge and understanding: Students are able to understand scalability and are able to set up and use a scalable IT system. In addition, students are able to evaluate high scalable IT solutions in terms of benefits and security risks.

Making judgements: Students are able to analyze the requirements of a specific use case and can decide which technology is best used for that case of application.

Communication: students are able to communicate about scalable IT systems and specific security requirements.

Learning skills: Additionally, students are able to analyse the interdependencies between large organizations, processes and IT solutions - taking into account security-related aspects - and to design suitable solutions using cloud offerings.

Study material: Lecture notes

Exam: Assignments and Project

ECTS: 4

Logic for Artificial Intelligence (optional course) (KEN3231)

Examiner: Prof. Dr. ir. Nico Roos.

Desired prior knowledge: Knowledge of propositional and predicate logic.

Prerequisites: The first year bachelor course: Logic.

Description: Logics form the formal foundation of knowledge representation and reasoning, which is a fundamental topic in Artificial Intelligence. Logics play a role as an analysis aid and as a knowledge-representation formalism. Moreover, the semantics of logics enables us to evaluate the intended meanings of knowledge representation formalisms, and the correctness and completeness of reasoning processes.

Humans make assumptions in their day-to-day reasoning. Examples of reasoning with assumptions are: common sense reasoning, model-based diagnosis, legal argumentation, agent communication and negotiation, and so on and so forth. The assumptions humans use in their reasoning may be incorrect in the light of new information. This implies that conclusions may have to be withdrawn in the light of new information. Therefore this form of reasoning is called non-monotonic reasoning and the underlying logics are called non-monotonic logics.

The course will cover model-based diagnosis as an application of reasoning with assumption, standard logics extended with defeasible rules, argumentation systems, the semantics of reasoning with assumptions and defeasible rules, and closure properties of the reasoning systems.

Knowledge and understanding:
- The student should be able to describe non-monotonic logics and argumentation systems.
- The student should be able to identify the logic underlying specific forms of knowledge representation.
- The student should be able to describe and discuss the semantic of non-monotonic logics.

Applying knowledge and understanding:
- The student should be able analyze important properties of practical formalisms for knowledge representation and reasoning.
- The student be able to apply non-monotonic logics and argumentation systems to practical problems

Making judgements:
- The student should be able to judge whether specific knowledge representation formalisms are able to represent the intended meaning of the knowledge to be represented.
- The student should be able to analyze whether conclusions derived from a knowledge representation are correct and complete.

Communication:
- The student should be able to explain how logic can be used as a tool for analyzing a knowledge representation problem.
- The student should be able to explain issues involved in the handling assumptions in a knowledge-representation.

Learning skills:
- The student should be able to study autonomously the literature describing the applications of logics for knowledge representation and reasoning.

Study material: Syllabi.

Recommended Literature: A syllabus and scientific literature.

Examination: Written exam at the end of the course. A bonus of 1.0 point can be earned by a series of bonus assignments.

ECTS: 4

Parallel Programming (optional course) (KEN3235)

Examiners: Prof. Dr. Hans Pflug, Dr. Bastian Küppers

Prerequisites: Introduction to Computer Science 1 and 2, Data Structures and Algorithms.

Description: Parallel programming introduces the students to the paradigm of parallel computing on a computer. Nowadays almost all computer systems include so-called multi-core chips. Hence, in order to exploit the full performance of such systems one needs to employ parallel programming.

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

After completing this course students will be able to write parallel programs with MPI and OpenMP on a basic level, and deal with any difficulties they may encounter

Knowledge and understanding: Students recall the basic concepts for parallel programming and recognize important parallelization patterns.

Applying knowledge and understanding: Students are able to write parallel software code using MPI, OpenMP, and Java Threads.

Communication: Students are able to explain why a specific pattern is adequate for a given problem.

Learning Skills: Students are able to study autonomously the literature describing parallel programming in order to comprehend important details and problems of the field.

Study material: Course notes and several codes will be provided online.
Recommended literature: Parallel programming with MPI; Peter Pacheco; Morgan Kaufmann (1996); (a very early revision is available online)

Exam: Written exam.

ECTS: 4

Software and Systems Verification (optional course) (KEN3150)

Examiner: Dr. Pieter Collins

Desired prior knowledge: Reasoning Techniques, Theoretical Computer Science

Description: Have you ever written a program with a bug in it? Then this course is for you! Software verification tools can check whether your program works by showing that it correctly satisfies its specification, or finds a case in which it can go wrong. Unlike unit testing and other software validation methods, verification tools use formal methods to rigorously prove correctness. Similar techniques can be used to show that (mathematical models of) cyber-physical robotic systems work as designed.
In this course, we will start by and introducing the main notions of object-oriented program verification, including pre- and post-conditions for methods, and class invariants. We shall use Hoare logic to convert programs and their specifications into logical statements to be proved. We shall apply these techniques to the verification of simple programs written in Java.
In the second part of the course, we consider formal models of software and systems as labelled transition systems (automata), using temporal logics for specification, and consider the fundamental algorithms for verification. We shall apply these algorithms to simple discrete verification problems, such as vending machines and communications systems, modelled using a formal system specification language. Finally, we will look at simple continuous systems, such as robots and electronic systems, and show how to verify these using rigorous numerical methods based on interval arithmetic.

Knowledge and understanding: Students are able to recognise the difference between formal verification and validation, and distinguish rigorous numerical methods, notably how interval arithmetic differs from floating-point. They can explain the various kinds of annotations used in program specification, state the deduction and precondition rules of Hoare logic, and interpret linear temporal logic formulae.

Applying knowledge and understanding: Students are able to write formal specifications for simple programs. Furthermore, students can use Hoare logic to reduce program specification to first-order logic statements, and justify these. Students are able to construct Büchi automata accepting temporal logic formulae and can apply interval and affine arithmetic for verifying properties of continuous systems. Moreover, students are able to write annotations for object-oriented software, use software for model-checking discrete systems and use software for rigorous numerics to verify safety of simple continuous systems.

Making judgements: Students are able to determine the most appropriate modeling framework and verification tools for a given problem.

Communication: Students can write and read formal specifications and can discuss informal design goals and their translation into formal specifications.

Learning skills: Students will critically reflect on their own human reasoning and the potential of digital computers to perform provably-correct automated reasoning.

Study material: Course notes.

Recommended literature:
- J.B. Almeida, M.J. Frade, J.S. Pinto & S. Melo de Sousa, "Rigorous Software Development: an Introduction to Program Verification", Springer, 2011.
- C. Baier & J.P. Katoen, "Principles of Model Checking", MIT Press, 2008.
- L. Jaulin, M. Kieffer, O. Didrit & E. Walter, "Applied Interval Analysis", Springer, 2001.

Exam: Written exam (100%)

ECTS: 4

Introduction to Bio-Informatics (KEN3440)

Examiner: Dr. Rachel Cavill.

Desired Prior Knowledge: Introduction to Computer Science 1, MatLab.

Prerequisites: None.

Description: This course presents a general introduction to the fundamental methods and techniques of bioinformatics in biomedical and biological research. The objective is that the students will acquire a general understanding of bioinformatics methods at the algorithmic level and will therefore be able to read and understand publications in this field, and – to some extent – apply their knowledge to concrete biological problems. This relates to the major areas of bioinformatics like sequence alignment, phylogenetic analysis, gene finding, and omics data analysis. This course consists of a series of closely related lectures and computer classes, based on relevant case-studies using real data. In the lectures the main theoretical aspects are presented. In the computer practicals, the students work to analyse real data using the techniques they have encountered. By extensively exploring the case study, the students acquire a thorough understanding of the subject.

Knowledge and understanding: Students should be able to perform common analyses on both sequence data and numeric data from omics experiments. This includes sequence alignment, building phylogenetic trees, applying hidden Markov models, detecting differentially expression and performing pathway analysis.

Applying knowledge and understanding: For all the above topics students should be able to demonstrate the algorithms on paper with simple examples and apply the algorithms appropriately on realistic datasets using a computer.

Making judgements: After successful completion of the course, students will be able to judge the use, quality, and correctness of different bioinformatics algorithms and results.

Communication: After this course students will be able to explain the algorithmic basis of bioinformatics problems and interact with biologists to provide recommendations of analysis approaches in the situations studied.

Learning Skills: After successful completion of the course students will be able to independently read bioinformatics literature to further their knowledge.
Study material: Introduction to Computational Genomics, A Case Studies Approach, Nello Cristianini, Matthew W. Hahn, Cambridge University Press, 2006, Hardback and Paperback (ISBN-13: 9780521856034 | ISBN-10: 0521856035).

Exam: Written exam (50%) + assignments (50%).

ECTS: 4

Introduction to Quantum Computing (optional course) (KEN3241)

Lecturer & Examiner: Dr. Menica Dibenedetto

Prerequisites: Linear Algebra.

Desired prior knowledge: Theoretical Computer Science, Data Structures & Algorithms.

Description: This course offers an introduction to the interdisciplinary field of quantum computation. The focus will lie on an accessible introduction to the elementary concepts of quantum mechanics, followed by introducing the mathematical formalism and a comparison between computer science and information science in the quantum domain. The theoretical capability of quantum computers will be illustrated by analysing fundamental algorithms of quantum computation and its potential applications.

Quantum technology has become one of the most prominent interdisciplinary fields of recent research. This course will focus on introducing the mathematical concepts underpinning quantum computation, and on explaining how this new computational paradigm might potentially offer possibilities beyond the scope of conventional computers. Topics that will be introduced and discussed include: (i) most common models of quantum computation (e.g., quantum circuits and measurement-based quantum computing). (ii) An exposition of the machinery borrowed from quantum mechanics, such as superposition of states, quantum entanglement, (de)coherence etc., which gives rise to the potential speed-up of quantum algorithms over their classical analogs. (iii) Some of the most common quantum algorithms (searching, factoring etc.) and protocols (quantum teleportation, EPR paradox). The course will finish with an exposition of potential applications of quantum computation and algorithms in other fields (such as security/cryptography, AI, optimization etc.)

Important: no prior knowledge in quantum mechanics is assumed or required, and all necessary concepts will be introduced and motivated from a mathematical and theoretical computer science point of view. Possible quantum architectures and/or related hardware issues will not be discussed.

Knowledge and understanding: By the end of this course, students are able to understand the differences between classical and quantum computation: Where is the computational power of quantum machines coming from? What are the limits of this new computational paradigm? What does the term “quantum supremacy” mean and why it is important? How likely is it ever to be achieved and what would it mean for our current understanding of the computational landscape?

Applying knowledge and understanding: Students are able to understand some of the most famous quantum algorithms, and to demonstrate where their power comes from. They will be able to judge how this potential computational power can be leveraged, and how it can be applied to other fields in a beneficial way.

After successful completion of this course, students are able to understand and use the mathematical framework of quantum computing to solve computational problems.

Making judgements: Students are able to judge and identify the settings where the potential quantum power might be beneficial and how they can leverage this. Students will further be able to analyse simple quantum algorithms for different computational problems.

Communication: Students are able to discuss quantum computation critically and judge not only its benefits but, equally important, its shortcomings. During lectures and practical assignments, students will be exposed to a different way of thinking about computation that will also enhance their understanding on classical computation.

Learning Skills: Students are able to critically read and understand scientific papers on quantum computing. To explain and analyse quantum algorithms described in quantum circuit or measurement-based quantum computing models. Finally, to relate quantum complexity classes to the classical ones.

Recommended Study material:
- Isaac Chuang, Michael Nielsen, "Quantum Computation and Quantum Information", 10th Anniversary Edition, Cambridge University Press, 2011.
- N. David Mermin, "Quantum Computer Science: An Introduction", 1st Edition, Cambridge University Press, 2007

Course material will be also provided during the lectures.

Exam: The final course grade is 100% of the final written "closed-book" exam grade.

ECTS: 4

Period 3.3

Project 3-1 (KEN3300)

Examiners: Dr. Katharina Schneider and Dr. Rico Möckel

Prerequisites: Project 2-1.

Description: Project 3-1 consists of two distinct paths: projects at the Department of Advanced Computing Sciences with focus on university research and DSAI/BSSC/BISS projects with focus on applied research proposed by companies that are affiliated with BSSC (Brightlands Smart Service Campus). The DSAI/BSSC/BISS projects are facilitated in cooperation with BISS (Brightlands Institute for Smart Society). In the first week of period 1, students indicate their preference by ranking these projects. Groups are created by means of an algorithm that minimizes regret and allocates students to their most preferred options.

About the projects at the Department of Advanced Computing Sciences: Students work in small groups, guided by teachers of the subjects concerned and by the tutors. During the project, students apply their knowledge in data science, and artificial intelligence to robotic and other intelligent and autonomous systems. Depending on their chosen specialization within their project group, students study and search for solutions in at least one, typically in multiple of the following fields: control, machine learning, computer vision, signal processing, human-computer/robot interaction, multi-agent and distributed systems, optimization, data visualization as well as modelling and simulation.

About the DSAI/BSSC/BISS projects: Students participate in small groups and receive guidance from a tutor, a teacher with knowledge of the subjects concerned, and a content expert from the company. Furthermore, the students receive business related skills such as creating business presentations from a teacher at BISS. Students learn how to apply their knowledge in data science, and artificial intelligence to solve real-world problem that arise in a professional environment, and how to interact with a client from the industry.

Project Skills period 3.1 & 3.2:

Group CV Check (online or on-site): during this class, you will receive tips and feedback on how to write a professional résumé (i.e. Curriculum Vitae).

Networking skills (online or on-site): during this class, you will learn hands-on tips to build an interesting network to support you in your search for a job or internship.
Both of the above classes are provided by instructors of the UM Career Services.

Study material: Period book 3.1-3.3. Maastricht University.

Exam: The project will be assessed based on report, product, presentation and project management.

ECTS: 6

Period 3.4

Data Analysis (KEN3450)

Examiners: Dr. Jerry Spanakis

Desired Prior Knowledge: Calculus, Linear Algebra, Mathematical Modelling & Simulation, Machine Learning, Introduction to Computer Science 1 and 2.

Prerequisites: None

Description: This course aims at preparing students on how to be a successful "data scientist". The crucial processes of inspecting, cleaning, transforming, restoring and preparing data for modelling are tackled. Different types of data are going to be explored through case studies ("clinics") that a modern "data scientist" has to deal with. Furthermore, several techniques from machine learning and mathematical modelling (multiple regression, classification, tree-based models, dimensionality reduction, etc.) are presented from the data analysis perspective and students learn how to apply these techniques to different types of data. Finally, the cornerstone of data analysis is presented: correct communication of the analysis outcome (storytelling, visualization, etc.).

Knowledge and understanding: Students are able to illustrate and explain data analysis and machine learning techniques with emphasis on modelling, and to give examples of different domains where data analysis can be applied

Applying Knowledge and understanding: Students are able to examine datasets using techniques learned in course, and to experiment with different techniques for data modelling

Making Judgements: After successful completion of the course, students are able to 1) judge the quality of data (of any kind), 2) to justify and rank which techniques should be applied in each problem and 3) to assess results of data analysis process

Communication: Students are able to present the results of different stages of data analysis to specialists and non-specialists and are able to decide on the correct communication medium (scientific, verbal and visual) of the analysis outcome

Learning Skills: After successful completion of the course, students are able to suggest options for tackling different datasets combining verbal, numerical/scientific and visual descriptions, also taking into account the context cases (e.g. business, academic) or the domain of application. Furthermore, students are able to formulate data descriptions based on their characteristics and can suggest options for modelling data and perform basic temporal analysis and dimensionality reduction

Study material: Jupyter notebooks (and limited slides)

Recommended literature: Selected chapters from the following textbooks:
- A. Downey, Think Stats: Exploratory Data Analysis
- James, G., Witten, D., Hastie, T., Tibshirani: An Introduction to Statistical Learning (with Applications in R)
- J. Vanderplans, Data Science Handbook
- S. Skiena, The Data Science Design Manual
- J W. McKinney, Python for Data Analysis
- Chris Albron, Machine Learning with Python Cookbook

Exam: Open-book Written Exam 50%, Data clinics 50% (20% individual assignments, 30% group assignments)

ECTS: 4

Operations Research Case Studies (KEN3410)

Examiner: Dr. Steven Kelk and Dr. Barbara Franci

Desired Prior Knowledge: Linear Programming.

Prerequisites: None.

Description: Operations Research (OR) is concerned with the best way to assign scarce resources to competing activities. It is for this reason an important branch of mathematics that is widely used in industry to support economically efficient decision making, but also in other application areas where discrete or stochastic optimization has a central role. In this course we will explore a number of themes both within deterministic OR (where all the problem data is known at the beginning) and stochastic OR (decision problems involving uncertainty and randomness). Themes within deterministic OR include the network simplex method (used for solving minimum-cost flow problems), integer linear programming and non-linear programming. Stochastic themes include queuing systems, Markov chains and Markov decision problems. As background students will be introduced to the methodological similarities and differences between OR and data science.

Knowledge and understanding: Students can recognize, classify and distinguish some of the major types of OR models, i.e. transportation and network optimization models, integer and non-linear programming, Markov chains and Markov decision problems, queueing models.

Applying knowledge and understanding: Students can apply a wide variety of algorithms to calculate solutions for problems of the types mentioned above. Students will be able to translate simple real-world/industrial optimization problems into a format suitable for (variously) the transportation simplex, network simplex and integer linear programming.

Making judgements: Students can explain advantages and disadvantages of different models and algorithms. They are able to judge the correctness of solutions presented.

Communication: Students can explain and defend their solution methods.

Learning skills: Students will be able to critically reflect upon the scope and limitations of the learned models, and be able to identify follow-up literature describing paradigms, models and algorithms that go beyond the scope of the course.

Study material: Hillier & Lieberman (2010 or 2015): Introduction to Operations Research (9th or 10th edition). McGraw Hill, ISBN 978-007-126767-0 or ISBN 9781259162985. Support for the 11th edition is forthcoming.

Recommended literature: None.

Exam: Written exam, worth 100% of the credit.

ECTS: 4

Intelligent Systems (KEN3430)

Examiner: Dr. ir. Kurt Driessens

Desired Prior Knowledge: Machine Learning

Prerequisites: None.

Description: The course offers an introduction to intelligent systems, their components, design issues and possible development paths. Based on the metaphor of a computational agent (that is, a software program or a robot which acts and interacts flexibly and autonomously in order to achieve some goal), basic concepts and methods from agent technology are discussed. Topics covered are the concept of artificial intelligence, expert systems, characteristics of an agent and agent architectures, agent cooperation and competition among agents, behaviour-generation and -learning with the added complexity of a multi-agent environment, agent oriented world views and possible future paths to general artificial intelligence. An emphasis is made on the complexity of interacting systems, both between different agents, but also between the subsystems of a single agent. In the practical part of the course, the students build up their experience with the implementation of a number of different types of agents.

Knowledge and understanding: Students are able to compare and discuss benefits and drawbacks of a number of different agent technologies. They can also explain the complexities arising from interactions between multiple techniques within a single agent, and the interactions between agents and systems.

Applying knowledge and understanding: Students will be able to implement of a number of different types of agents architectures and agent-subsystems and agent behavior generation techniques.

Making judgements: The student will be able to judge whether it is beneficial to use intelligent systems technology over other approaches for handling a given problem, and which agent architectures might fit best.

Communication: The student will gain a working knowledge of intelligent system terminology and will learn to motivate his/her choices concerning the application of intelligent technology.

Learning Skills: Students have to reflect upon their knowledge and recognize the need for continued learning as they are confronted with the complexities involved with applying the knowledge gained in their bachelor studies and linking individual techniques into a working system.

Study material: Course slides to support the lectures; supplementary material consisting of research papers and book chapters.

Examination: Written exam (80%) + assignments (20%).

ECTS: 4

Period 3.4 to 3.6

Bachelor’s Thesis (KEN3500)

ECTS: 18

Bachelor’s thesis Data Science and Artificial Intelligence
At the end of the Bachelor’s study in Data Science and Artificial Intelligence each individual student has to write a thesis manuscript. This thesis manuscript must be designed as a scientific article of 8 pages using a standard (LaTeX) design. Students are expected to conduct a pro-active and independent research on their topics. This includes the search and reading of related work. The topics must be discussed with the potential thesis supervisor(s) and a research plan must be submitted to and approved by the Board of Examiners as an initial step. The thesis has to be accompanied by relevant attachments and software. Students will present the thesis in a public on-site conference.

This means that a strict submission form will be used. In order to start working on the thesis, a student is required to have obtained at least 140 ECTS (among which are 60 credits of the first year, and 40 ECTS of the second year).

General procedure

Below is an indication for these phases. A bachelor's thesis coordinator will supervise the entire procedure and schedule. *Please note there is also an option to start the trajectory in September. For more information ask the thesis coordinator.*

November
Phase 0: Thesis Topic meeting
Potential topics and research fields will be presented.

January
Phase 1: Topic selection
During the skills class, each student selects a topic (and problem statement) and finds two appropriate prospective thesis examiners. Every student hands in a signed bachelor's project plan to the bachelor's thesis coordinator. If the Board of Examiners approves the thesis plan, the examiners are appointed.

Periods 3.4-5: February - May

Phase 2: Research
In this period every student conducts his/her own research. This will preferably be guided in groups by the thesis supervisor. Further two seminars will be organised during which the students present their progress.

Phase 3: Writing
Parallel to the research, a scientific article is written.

Period 3.6: June

In Period 6, the research is finished and the first versions of the thesis manuscript is expected (first submission). The first thesis examiner will evaluate the thesis manuscript and gives a first reaction within around a week. The second examiner will also evaluate the paper during this week. The second and final submission will take place at the end of the second week of period 3.6 (concrete dates will be announced).

Phase 4: Preparation for presentation
In the second week of period 3.6, the preparation for the final presentation will start for every student individually. The presentations will be created with PowerPoint and have a maximum length of 10 minutes

Phase 5: Presentation
The bachelor's theses will be presented in the third week of period 3.6 in a scientific conference setting at the university. The presentations have a maximum length of 15 minutes per student (including questions). The conference is open to all students and teaching staff from Data Science and Artificial Intelligence and anyone else who might be interested. The final decision on the grade for the bachelor's thesis will be made shortly after the presentations. A special bachelor's thesis coordinator will supervise the phase.

Re-sit: In case the student fails to present his/her work at the Bachelor conference, the student gets one opportunity to defend his/her work at the next bachelor conference. If the student does pass at any of those two conferences, the student has to select a new topic and submit a new thesis plan. For students not finishing at the June Conference there is one re-sit possibility in a Conference at the end of August.

Requirements for the bachelor's thesis project
For the bachelor's thesis, every student has to conduct a short scientific research project. This can be an empirical as well as a theoretical research. The topic for the research project is open, as long as it fits into the Data Science and Artificial Intelligence program.
The department will offer a list of potential research topics. The topic and the research questions have to be approved by the examiners and the Board of Examiners. To this end, the student will create a bachelor's project plan using the form provided by the Board of Examiners. This plan will be signed by the student, the prospective thesis examiners and then handed in to the bachelor's thesis coordinator. It is possible to execute the bachelor thesis project as an external training period. This should be well defined in the bachelor's thesis plan. In this case, the plan should also include the name of the external institute or company, the name of the external supervisor, the size of the project and any agreements about compensation. The plan should also be signed by the external supervisor. In principle, there should be no confidentiality agreements for a thesis, and staff members cannot be expected to commit to these. The external research cannot start before period 3.5 due to courses in period 3.4. When not selecting a topic offered by the department, or when wanting to do a thesis with the involvement of an external party, it is advisable to start the preparations and requesting permission three months in advance. The research needs to be original in such a way that the thesis supervisor is convinced that this research has not been done before. The research also needs enough depth and still it must be possible to finish it in the set amount of time. Every thesis is an individual work.

Requirements for the bachelor's thesis manuscript

Content aspects
The thesis manuscript describes the cause, research question, approach and results of the research. This has to be done in a clear, structured and scientific manner which includes:
- a clear introduction in which the context and research questions are presented;
- a clear conclusion, based solely on the already used thought out principles and derived results;
- a clear line is shown between problem statement, methods, and the derived results;
- a motivation of the methods followed;
- an adequate description of the methods followed;
- an honest, clear, and concise description of the derived results, if necessary using tables;
- a discussion of the results;
- the usage of relevant and recent literature;
- the correct usage of references;
- the adequate usage of the literature for the reasoning in the thesis manuscript.

Design aspects
The number of pages of the thesis is 8, in the designated LaTeX format, including images and references. This thesis should at least contain:
- title;
- author;
- abstract;
- one or two keywords;
- list of references;
- page numbers.

It goes without saying that the correct scientific references are used for used resources (by using the designated BiBTeX reference style). Images and tables are accompanied by an index and caption. Mathematical formulas, definitions, etc. have to be properly designed and numbered. The start and end of mathematical formulas have to be properly defined.

Language aspects
The thesis manuscript has to be written in Dutch or English, considering correct spelling, syntactical structure of sentences and structure of content in paragraphs. The target audience is fellow Data Science and Artificial Intelligence students. Any jargon and/or abbreviations have to be explained unless they are common knowledge for this audience (e.g. CPU).

Citations
It is allowed to use several short citations with a maximum length of two sentences.
These citations have to be clearly referenced and have to be typographically distinguishable (that is, citations are placed in quotes). Non-allowed citations or missing references will result in an unsuccessful result.

Assessment
The assessment will take place based on the contents and design of the thesis, the presentation of this thesis and the process. The weighing of the various aspects is up to the examiners.

2.4 Curriculum of the First Year of the Bachelor Programme Computer Science

Course year 1:		ECTS
Period 1.1	Introduction to Computer Science (BCS1110) Procedural Programming (BCS1120) Discrete Mathematics (BCS1130) Project 1-1 (BCS1300)	4 4 4
Period 1.2	Logic (BCS1530) Objects in Programming (BCS1220) Calculus (BCS1440) Project 1-1 (BCS1300)	4 4 4
Period 1.3	Project 1-1 (BCS1300)	6
Period 1.4	Linear Algebra (BCS1410) Data Structures and Algorithms (BCS1420) Object-Oriented Modelling (BCS1430) Project 1-2 (BCS1600)	4 4 4
Period 1.5	Databases (BCS1510) Statistics BCS1520) Algorithmic Design (BCS1540) Project 1-2 (BCS1600)	4 4 4
Period 1.6	Project 1-2 (BCS1600)	6

Period 1

Introduction to Computer Science (BCS1110)

Examiner: Dr. Ashish Sai and Dr. Thomas Bitterman

Desired Prior Knowledge: None

Prerequisites: None

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

Knowledge and insight: Students will be able to explain the fundamental concepts and how they relate to the broader field of Computer Science. Students will demonstrate awareness of the current trends and developments in Computer Science.

Applying knowledge and insight: Students will analyze and decompose a given computational problem and identify appropriate methods and tools to design a solution. They will apply the fundamental concepts and critical skills to a practical problem requiring a solution that touches on various subfields of Computer Science.

Judgement: The students will be able to recognize and compare different approaches and techniques available within computer science to solve a given problem.

Communication: Students will communicate about different aspects of computer science using appropriate terminology and format such as pseudo-code and UML diagrams. They will describe solutions to abstract computational problems verbally and in writing. They will also be able to justify their design choices and document them in a traceable manner.

Learning skills: Students will work independently and collaboratively to understand and decompose a computing problem. They will identify and apply techniques and tools in order to solve the problem.

Study material: "An Invitation to Computer Science" by G. Michael Schneider, 8th Edition

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

Exam: Written exam (75%) + group project (25%)

ECTS: 4

Period 1

Procedural Programming (BCS1120)

Examiner: Dr. Enrique Hortal, Dr. Tom Bitterman and Dr. Daniel Cámpora

Desired Prior Knowledge: None. The course appears as desired prior knowledge for the courses Introduction to Computer Science 2, Data Structures and Algorithms, Software Engineering, Databases and Machine Learning.

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

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

Knowledge and understanding: The course offers preliminary methodological and theoretical bases for studying and applying computers and computer programming on which the rest of the curriculum builds.

Applying knowledge and understanding: Whenever a computer system or a programming system has to be designed and implemented the knowledge and insights acquired during the course can be used and applied.

Making judgements: After successful completion of the course, students will be able to judge the quality and correctness of simple non-object-oriented programs.

Communication: The skills acquired during the course will enable students to communicate about standard programming constructs and algorithmic basics.

Learning Skills: After successful completion of the course, students will be able to formalize, analyse and program solutions to simple software problems.

Study material: Lecture slides, example code and multimedia material that are made available before and after each lecture.

Recommended literature: H. Schildt, Java: A Beginner's Guide, Eighth Edition, ISBN: 1260440214, McGraw-Hill Education
Additional literature: C. Horstmann (2016). Java Concepts (8th Edition). John Wiley & Sons, New York, ISBN: 978-1-1190-5645-4 or C.Horstmann (2012). Big Java Late Objects. John Wiley & Sons, New York, ISBN 978-1-1180-8788-6

Exam: Closed-book written exam (80%) + Assignments (20%)

ECTS: 4

Discrete Mathematics (BCS1130)

Examiner: Dr. Marieke Musegaas, dr. Otti D'Huys and dr. Stefan Maubach

Desired Prior Knowledge: None.

Prerequisites: None.

Description: In this course, we build a mathematical framework that is based on logic and reason. The main objective of the course is to make students familiar with the language of mathematics. Students will learn how to make sound arguments and to detect where and why certain arguments go wrong. For this purpose, we will discuss the basic principles of logic and, closely related, the basic types of mathematical proofs. In doing so, we will encounter numbers such as integers, natural numbers and real numbers and we shall examine what makes these numbers special. After that, we will use basic logic to discuss, among other things, the following mathematical concepts: infinity, sets, relations, functions, permutations and combinations. Our fundamental tool in all of this is plain common sense. You really do not need your toolbox of mathematical formulas learned in previous studies and neither do you need a calculator. Pen and paper are the basic instruments needed. After completing each topic, exercises will be provided to be completed in class or at home, since mathematics is mainly learned by practising repeatedly.

Knowledge and understanding: Students will be able to read, interpret and manipulate basic mathematical terminology (propositional logic, quantifiers, set theory, relations, functions, and combinatorics). Students will also be able to read and interpret several different types of mathematical proofs and identify whether a purported proof is mathematically sound.

Applying knowledge and understanding: Upon completion of the course students will know how to read, interpret, write and manipulate rigorous mathematical statements using propositional logic, quantifiers, set theory, relations, functions and combinatorics. Students will be able to select, from a range of mathematical tools, which is appropriate to prove or disprove a given mathematical statement, and apply the chosen tools, rigorously and clearly in order to achieve the desired goal.

Making judgements: Students will be able to distinguish between mathematically sound and unsound statements and defend the rigour of their own mathematical arguments.

Communication: Students will be able to write clear, rigorous and explicit mathematical arguments using standardized mathematical terminology and such that each step in the argument is a logical consequence of earlier steps.

Learning skills: By the end of the course, students will be able to autonomously and critically reflect upon the mathematical correctness of their own arguments.

Study material: A. Chetwynd & P. Diggle: Discrete Mathematics. Butterworth- Heinemann, Oxford, ISBN 9780340610473. Lecture notes will also be provided.

Recommended literature: None

Exam: Closed book written exam

ECTS: 4

Period 2

Logic (BCS1530)

Examiner: Dr. Tjitze Rienstra and Dr. Stefan Maubach

Prerequisites: None. The course appears as a prerequisite for the course Logic for AI.

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

Knowledge and understanding: Students need to get accustomed to the fundamental concepts of mathematical logical systems (propositional logic and predicate logic) to able to describe information in a logical framework and to reason and prove correctly. Students will get accustomed to the basic concepts of some advanced logical systems (epistemic logic and dynamic logic).

Applying knowledge and understanding: Student will apply the reasoning and proof methods learned to small-scale problems and some more complex situations.
Making judgements: Students will learn to judge how to reason correctly using mathematical proofs and how to judge which logical system is suitable to solve the problem at hand.

Communication: The chosen syntax of the logical language used must be easily understandable by peers and others experts the logical proofs given must be correct, concise and easily understandable

Learning skills: having learned basic logical concepts and reasoning techniques the students are able to apply them to larger-scale problems

Study material:
- Johan van Benthem, Hans van Ditmarsch, Jan van Eijck, Jan Jaspars, Logic in Action. Edition of February 2014 or later. This is a freely available e-book. Check your Canvas for the link.

Exam: Written exam; during the course the students will receive three assignments, that, if they receive a sufficient grade, may earn them up to a total of one bonus point.

ECTS: 4

Objects in Programming (BCS1220)

Examiner: Dr. Evgueni Smirnov, dr. Tom Bitterman and dr. Francesco Barile

Desired prior knowledge: Basic Java Programming

Prerequisites: None.

Description: This course is a follow-up to the course Introduction to Computer Science 1. It teaches object-oriented programming in Java. The main topics covered in the course are objects and classes, interfaces and polymorphism, event handling, inheritance, graphic user interfaces, exception handling, and streams.

Knowledge and understanding: After successful completion of the course, students will be able to explain the methodological and theoretical principles of object-oriented programming.

Applying knowledge and understanding: Students will be able to implement basic object-oriented computer programs. They will be able to design and describe simple object-oriented computer systems.

Making judgements: Students will be able to judge the quality and correctness of simple object-oriented programs.

Communication: Students will be able to communicate about object-oriented programming constructs and algorithmic basics.

Learning skills: Students will be able to recognize their own lack of knowledge and understanding and take appropriate action such as consulting additional material or other sources of help.

Study material: Course notes, slides, and other information made available.

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

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

Exam: Written exam (80%) + practical assignments (20%).

ECTS: 4

Calculus (BCS1440)

Examiner: Dr. Otti D'Huys, Dr. Martijn Boussé, Dr. Gijs Schoenmakers

Prerequisites: None.

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

Knowledge and understanding: Student can define, write and explain key facts and concepts involving limits and continuity, can interpret and solve differential calculus, integral calculus, sequences and series, first-order linear differential equations problems, and understand the basics of multivariable calculus.

Applying knowledge and understanding: Students are able to solve problems applying the concepts learned in the course, using standard problem-solving strategies.

Making judgements: Students are able to analyse a simple problem within the course content and justify the solution methodology they choose. They can summarize this methodology mathematically.

Communication: Students are able to explain their solution strategy in written form and defend their solution strategy in discussion with others

Learning Skills: After successful completion of the course the students will be able both to solve standard problems (constructing graphs of functions, finding extrema of functions, computing limits, summing infinite series etc.) and to apply their knowledge in solving and analysing more complex problems (e.g. in analysis of numerical algorithms).

Study material: Calculus, a complete course, any edition, by R.A. Adams, Addison Wesley Longman and materials provided during the lectures.

Exam: Intermediate bonus assignments and a final written exam.

ECTS: 4

Course title: Project 1-1 (BCS1300)

Coordinator: Dr. Martijn Boussé

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

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

Knowledge and understanding: Interpret constraint-satisfaction problems arising in practice and translate this to discrete-mathematical algorithmic models capable of solving the problem. Gain insight into practical use of basic software design and development principles. Recognise and relate user-computer interactions to concepts from graphics and user-interface frameworks. Strengthen knowledge of basic algorithms and methods for efficiently solving constraint-satisfaction problems arising in applied mathematics (especially: discrete mathematics) and artificial intelligence.

Applying knowledge and understanding: Design an answer strategy for scientific questions using analytical thinking and logical reasoning. Translate discrete-mathematical algorithmic models to software code. Implement software to efficiently solve constraint-satisfaction problems arising in applied mathematics (especially: discrete mathematics) and artificial intelligence by finding, designing and applying appropriate algorithms. Formulate computational experiments, and analyse and interpret the results. Apply basic design and development principles in the construction of software systems. Use existing software application frameworks for graphics and user interfaces. Use tools for software project management such as version control systems and issue trackers. Identify project goals, deliverables, and constraints. Plan and chair meetings. Create minutes for meetings. Work in a team such that the workload is balanced. Plan teamwork by setting deadlines and distributing tasks.

Making judgements: Evaluate different mathematical and computational models with respect to their suitability, efficiency and correctness for a specific task. Elicit and evaluate relevant scientific background information. Evaluate the group's progress during the project.

Communication: Give a clear and well-constructed presentation, including a demonstration of the product, and with appropriate use of illustrations and/or videos. Offer and respond to questions on and constructive criticism of presentations. Write a project report according to the structure of an academic article. Submit arguments in exact sciences, with appropriate use of formulae and figures. Cite published sources in the project report according to the academic guidelines. Structurally inform stakeholders on project progress. Effectively communicate with project group members about task division, planning and project deadlines. Effectively communicate with group members by listening to others' ideas; be contactable and include others in the discussion. Cooperate in a group to reach a consensus view. Give constructive feedback to team members. Communicate in the English language.

Learning skills: Reflect on one's own academic abilities and functioning in a team.

Study material: Project manual project 1-1, Maastricht University

Assessment:
Final Grade = 0.9·(0.15·gradePhase 1+0.15·gradePhase 2+0.7·gradePhase 3)+ skillClassGrade +individualGrade

where gradePhase 3=0.4 gradeProduct+0.4 gradeReport+ 0.1 gradePresentation+ 0.1 gradeProjectManagement.

The *individualGrade* is given due to either outstanding or not enough contribution of a student to the project. By passing skill classes, the students can get a reward called *skillClassGrade*, which is 1 if the students passed all skill classes, 0.5 if the students passed all but one skill classes, and 0 if the students passed all but two skill classes. Failing more than 2 skill classes will lead to an NG in the project. Missing mandatory project events such as project meetings and examination moments will lead to a reduction of the grade or even to receiving an NG for the project.

Skill classes:

Introduction to Project Work (period 1.1)

The students learn how PCL approach is applied in the programme. The structure of the projects is explained. Furthermore, the students will get to know their team members and will be introduced to various aspects of project work such as taking minutes and creating an agenda.

Elementary writing (period 1.1 and 1.2)

In period 1-1, the students have to take part in an English Diagnostic Test that determines their current English level. Those students, which scored low in the diagnostic test, will receive an additional training on elementary writing in form of a skill class. Furthermore, all students have to pass a quiz about writing.

Elementary presenting (period 1.2)

The students learn the basics of a good presentation. They will have the opportunity to revise the first draft of their presentation based on the learned concepts.

Citing and Referencing (period 1.2)

This skill class provides you with basic knowledge about when and how to cite.
Team Dynamics "How to effectively communicate and cooperate in my team?" (period 1.3)
In this skill class, you will learn effective communication strategies that will help you to cooperate in your team.

Advanced Presenting (period 1.3)

In this skill class, you will present within your project group and receive individual feedback on presentation style and content.
ECTS: 6

Period 4

Data Structures and Algorithms (BCS1420)

Examiner: Dr. Francesco Barile and Tom Pepels, M.Sc. and Dr. Daniel Cámpora

Tutor(s): TBA.

Desired Prior Knowledge: Discrete Mathematics, Introduction to Computer Science 1 and 2. The course is desired prior knowledge for Theoretical Computer Science.

Prerequisites: None. The course itself occurs as part of the pre-requisites of both projects of year 2 and the third year course Parallel Programming.

Description: As a continuation of the courses Computer Science 1 and 2, this course will treat the systematic design and application of data structures and algorithms. Data structures such as lists, trees, graphs, and dictionaries, the associated algorithms and their complexity are explored in this course. Algorithms for applications such as sorting, pattern matching and graph traversal are also part of the course. Furthermore, design principles for algorithms such as recursion, divide-and-conquer and dynamic programming will be treated as well. Furthermore, students will develop skills to analyse the run-time and space complexity of data structures and algorithms.

Knowledge and understanding: Students are able to give examples of data structures and explain do they support program design. Students are able to name what types of standard data structures exist and illustrate their properties. Students are able to describe some standard algorithms and highlight their properties. Students are able to illustrate how to develop and analyze new algorithms.

Applying knowledge and understanding: Students are able to select the appropriate data structure for a given problem and students are able to propose an algorithm for solving a given problem

Making Judgements: Students are able to justify if and determine how data structures are applied. Furthermore, students are able to assess whether algorithms are appropriate and efficient.

Communication: Students are able to explain how data structures and algorithms are to be included in program designs.

Learning Skills: Students are able to reflect on which data structures and/or algorithms are applicable for each problem.

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

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

Exam: 'Closed Book' written exam, during the course the students will receive a number of assignments, which can earn them up to a total of one bonus point.

ECTS: 4

Linear Algebra (BCS1410)

Examiner: Dr. Marieke Musegaas, dr. Philip Dreesen and Dr. Steve Chaplick

Desired Prior Knowledge: None.

Prerequisites: None.

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

Knowledge and understanding: Students are able to recognize and explain the fundamental concepts of Linear Algebra: systems of linear equations, vectors and vector spaces, basis and coordinates, matrices and matrix-vector computations, linearity and orthogonality, linear independence, rank, fundamental spaces (row space, column space, and null space), determinants and invertibility, eigenvalues and Eigen spaces, diagonalization.

Applying knowledge and understanding: Students are able to analyse a linear algebra problem from both an algebraic and a geometrical point of view. Students can solve systems of linear equations, compute determinants and rank, compute eigenvalues and Eigen spaces, make use of complex numbers, diagonalizable matrices, and perform change of coordinates.

Making judgements: Students are able to look at the same problem from different angles and to switch their point of view (from geometric to algebraic and vice versa).

Communication: Students are able to motivate both from an algebraic and a geometric point of view the solution set of a system of linear equations, the linear independence and orthogonality of a set of vectors, the linear transformation between two coordinate systems, the fundamental spaces associated with a matrix, the invertibility of a matrix, and the diagonalization of a matrix in terms of the properties of its eigenvalues and eigenvectors.

Learning skills: Students have acquired the skills to autonomously recognize elements of practical problems, which can be addressed and solved with linear algebra, and use Matlab to solve larger scale problems.

Study material: David C. Lay, Linear algebra and its applications, 6th ed., Pearson, ISBN: 978-1-292-35121-6.

Recommended literature: None

Exam: Closed book written exam

ECTS: 4

Object Oriented Modelling (BCS1430)

Examiner: Dr. Ashish Sai, Dr. Yuquan Wang

Desired Prior Knowledge: Procedural Programming, Objects in Programming.

Prerequisites: None.

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

Knowledge and insight: Students understand the principles and practices of object-oriented programming with a focus on design and analysis. Students can explain the use of tools such as UML for modelling software systems. Students also understand the use of design patterns and GRASP principles in improving software quality.

Applying knowledge and insight: After the course, students can critically analyze software models, designs and implementations. Students will be able to use UML diagrams to represent software requirements, system architecture, class structure, system behavior and interactions. Students will also be able to implement software solutions using appropriate design patterns and GRASP principles in an object-oriented language such as Java.

Making Judgements: The students can judge the suitability of different object-oriented modelling techniques, design patterns and GRASP principles for different software development problems. Students can also evaluate the trade-offs between various design alternatives in terms of complexity, modularity, flexibility and reusability. Students can critically assess the quality of their own and others' software designs and implementations using appropriate criteria and metrics.

Communication: Through the course, students become able to communicate effectively with different stakeholders involved in software development using appropriate techniques. Students can present their software designs and implementations using UML diagrams and documentation.

Learning skills: Students develop independent learning skills to keep up with the evolving trends in object-oriented design and analysis. Students also practice how to reflect on software design and implementation in a professional manner.

Study material: "Applying UML and Patterns" by Craig Larman

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

Exam: Written exam (70%) + practical assignments (30%)

ECTS: 4

Period 5

Databases (BCS1510)

Examiner: Tom Pepels, M.Sc. and Dr. Ashish Sai

Desired Prior Knowledge: Introduction to Computer Science 1 and 2, Data Structures and Algorithms, Software Engineering.

Prerequisites: Description: This course covers the use of (relational) databases and data modelling with the goal of writing (distributed) data-intensive software applications. Specifically, students will learn to use the Structured Query Language (SQL) to manipulate data to develop data-models that are Atomic, Consistent, Isolated and Durable. Moreover, the course covers alternative (distributed) data-storage methods and object persistence techniques such as NoSQL. During the course, students will learn to use different database management systems and how to use them to build software.

Knowledge and understanding: Students will be able to describe the basic concepts of databases, explain the fundamental concepts of database management systems, query languages, data modelling and database programming.

Applying knowledge and understanding: Students will be able to explain the proper database design based on system requirements, indicate possibilities and limitations of database types. In addition, students will be able to combine software architectures to design and construct a database application.

Making judgements: Student will be able to analyze and justify a practical database problem, examine different approaches, and refine database models based on use cases. Moreover, they can make improvements to existing database designs, reflect on certain solutions of the databases design and implementation, and assess the correctness of the database model.

Communication: Students will be able to summarize the basic entities and relationships involved in persistent data, and communicate with developers, database managers and users on proper database design and interfacing.

Learning skills: Students will be able to identify and understand follow-up literature, beyond the teaching material of the course.

Study material: Alan Beaulieu, 2020. Learning SQL, (3rd ed.). O'Reilly Media, Inc.

Recommended literature: Martin Kleppmann, 2017. Designing Data-Intensive Applications. O'Reilly Media, Inc.

Exam: Written exam (75%) + practical assignment (25%)

ECTS: 4

Statistics (BCS1520)

Examiner: Dr.ir. Marijn ten Thij

Prior Knowledge: Calculus, Discrete Mathematics.

Prerequisite: None.

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

Knowledge and understanding: Students will obtain an overview of the most relevant and frequently used probability distributions as well as several statistical procedures

Applying Knowledge and understanding: Students can calculate probabilities, expectations, variances and related quantities in a wide variety of probabilistic experiments; estimate statistical quantities; perform several statistical tests to extract information.

Making judgements: Students can analyse probabilistic experiments, critically analyse statistical inferences and decide whether to accept or reject statistical hypotheses. Choose, motivate, and contrast methods for statistical analysis.

Communication: Students can explain how to solve problems involving probabilities and/or statistical procedures.

Learning Skills: Students can reflect on the use of probability theory and statistics in other domains in order to increase one's knowledge.

Study material: Probability & Statistics for Engineers & Scientists - Walpole, Myers, Myers & Ye

Recommended literature: None

Exam: Written exam (80%) and an individual (take home) assignment (20%).

ECTS: 4

Algorithmic Design (BCS1540)

Examiner: Dr. Thomas Bitterman, dr. Yuquan Wang

Desired prior knowledge: Data Structures and Algorithm

Prerequisites: None

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

Knowledge and understanding: Students can give examples of run-time complexity classes for well-known algorithms. Students can differentiate between different algorithm designs from examples. Students can describe some advanced algorithms and highlight their properties. Students know the complexity classes P and NP.

Applying knowledge and understanding: Students will be able to derive the run-time complexity of select algorithms. Amortization can be applied when deriving algorithm complexity.

Making judgements: Students will be able to determine the best of several potential algorithms to apply to a problem. This judgement can be backed up by rigorous analysis.

Communication: Students will be able to carry out a written formal derivation of run-time complexity. Algorithms will be expressed in code.

Learning Skills: The algorithms and analysis techniques learned will be applicable to further work in the field.

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

Assessment: Closed book written exam plus assignments.

ECTS: 4

1.6 Course title: Project 1-2 (BCS1600)

Coordinator: Dr. Otti D'Huys

Prerequisites: In order to participate in this project the student has to have passed two out of four courses from the set: Discrete Mathematics, Calculus, Computer Science I and Computer Science II.

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

At the beginning of period 1.5 and 1.6, the students have to hand in a planning for the current phase. At the end of each period, the students have to give a presentation and the source code, presentation and an overview of who did what need to be uploaded to Canvas. While the presentations at the end of period 1.4 and 1.5 are in front of the examiners and the tutors, the presentations at the end of period 1.6 will additionally be in front of the fellow students. In period 1.6, they furthermore have to hand in a report and attend a product and report examination. Knowledge and understanding: Interpret the meaning of mathematical models of real-world processes. Gain insight into practical use of software design and development principles. Recognise and relate user-computer interactions to concepts from graphics and user-interface frameworks. Strengthen knowledge of basic algorithms and methods for specific problems in artificial intelligence and applied mathematics.

Applying knowledge and understanding: Students will be able to design an answer strategy for scientific questions using analytical thinking and logical reasoning and to translate mathematical models to software code. Furthermore, students will be able to implement software to solve problems in applied mathematics by applying numerical methods and artificial intelligence algorithms, formulate computational experiments, and analyse and interpret the results, apply design and development principles in the construction of software systems and use existing software application frameworks for graphics and user interfaces. Even more so, students will learn to use tools for software project management such as version control systems and issue trackers, identify project goals, deliverables, and constraints. Lastly they will learn how to plan and chair meetings, create notes for minutes, work in a team such that the workload is balanced and plan teamwork by setting deadlines and distributing tasks.

Making judgements: Students will learn to evaluate different mathematical and computational models with respect to their suitability, efficiency and correctness for a specific task.

Communication: Students will be able to give a clear and well-constructed presentation, including a demonstration of the product, and with appropriate use of illustrations and/or videos, to offer and respond to questions on and constructive criticism of presentations. Furthermore, they will learn to write a project report according to the structure of an academic article, submit arguments in exact sciences, with appropriate use of formulae and figures. They learn to cite published sources in the project report according to the academic guidelines. Additionally, students will learn to structurally inform stakeholders on project progress and effectively communicate with project group members about task division, planning and project deadlines, effectively communicate with group members by listening to others' ideas; be contactable include others in the discussion. It will be important to cooperate in a group to reach a consensus view, communicate in the English language, elicit and evaluate relevant scientific background information.

Learning skills: Reflect on one's own academic abilities and functioning in a team.

Study material: Project manual project 1-2, Maastricht University.

Assessment:
Final Grade = 0.9 · (0.15 · gradePhase 1+0.15 · gradePhase 2+0.7 · gradePhase 3)+ skillClassGrade +individualGrade

where gradePhase 3=0.4 gradeProduct+0.4 gradeReport+ 0.1 gradePresentation+ 0.1 gradeProjectManagement.

The *individualGrade* is given due to either outstanding or not enough contribution of a student to the project. By passing skill classes, the students can get a reward called *skillClassGrade*, which is 1 if the students passed all skill classes, 0.5 if the students passed all but one skill classes, and 0 if the students passed all but two skill classes. Failing more than 2 skill classes will lead to an NG in the project. Missing mandatory project events such as project meetings and examination moments will lead to a reduction of the grade or even to receiving an NG for the project.

Skill classes:

Information Research: Systematic Literature Search (period 1.4)

This skill class will give the students an introduction to which databases, search strings and settings can be used to systematically search for literature. Furthermore, they will have to come up with a search plan for the current project.

Team Dynamics "Laying the foundation of effective teamwork" (period 1.4)

This team dynamics workshop aims to provide the students with a deeper awareness, insight and practice in effective team collaboration & co-creation. During this introduction workshop, you and your project team will draw up a team charter to initiate effective group collaboration in project 1-2.

Advanced Presenting (period 1.4)

This skill class will focus on presentation skills and techniques. This introduction helps with public speaking and prepares you for project report presentations. Areas of focus include: structure of a presentation, public speaking techniques and enunciating, language aspects to remember while planning a presentation, and the dos and don'ts expected by Maastricht University.

Academic Writing (period 1.5)

In the project skills meetings you will explore the key structure of your report, as well as key points of Academic Writing at Maastricht University. Areas of focus include: structure of paper; linguistic aspects of writing in English, presenting information logically and citation and reference procedures.

Team Dynamics "Evaluating and drawing lessons from the project teamwork" (period 1.6)

In this skill class, students are going to evaluate the team collaboration and communication during project 1-2 by means of interactive exercises.

ECTS: 6

3 Master

3.1 Curriculum of the First Year of the Master's Programmes

General introduction Master AI
The Master in Artificial Intelligence (AI) is a two-year advanced programme organised by the Department of Advanced Computing Sciences . The focus of this programme is on the understanding, design and creation of intelligent systems, such as those used in robotic systems, games or digital personal assistants. Artificial Intelligence has become a very active domain in both academia and industry. It has given rise to computer programmes and robots that learn from experience, recognise and adapt to patterns in their environment, and reason strategically in complex decision-making situations.

The impact of the field of Artificial Intelligence is pertinent due to the key role it plays in technological applications that have become indispensable in society, such as simple personal assistants that adapt the settings of your smart phone to automatically recognised activities (e.g. driving or attending a meeting, automated trading software used in real markets to respond to rapid price changes, interactive computer games that include human like opponents, robotic assistance in the exploration of dangerous environments, etc.). In this Master's programme, you are trained to become an expert and capable of dealing with todays and future challenges in the field of Artificial Intelligence and its applications.

The master's programme Artificial Intelligence covers a range of subjects emphasizing the following research topics as its core:
1. Intelligent techniques for playing and solving (board) games and controlling virtual characters in video games;
2. Situated agents to study the control and coordination of embodied agents, i.e. robots (e.g. autonomous flying robot swarms);
3. Multi-agent systems of collaborating autonomous intelligent systems;
4. Formal techniques for reasoning in agents and representing and communicating knowledge;
5. Machine learning to extract useful patterns and knowledge from experience and make predictions about the future;

The members of the teaching staff are actively involved in one or more of these research topics. As a result, the educational contents of the courses relate directly to the research performed.

General introduction Master DSDM
Data Science for Decision Making is the science of making informed decisions. It has widespread applications in business and engineering. In today's world, many companies and organisations collect all sorts of data in large amounts. They aim to extract useful information from it, to recognize patterns and anomalies. Data Science for Decision Making provides the mathematical tools to analyze and model the underlying real-world processes and decision questions, and to process and analyze the (big) data that come with the processes. It also provides and uses the computational software that is the key to data science.

The two-year master's programme in Data Science for Decision Making teaches the use of applied mathematics to analyse and optimize processes, problems and operations. Examples of applications are: discovering patterns in data such as images and time series, predicting future values such as demand for products or traffic on the roads, scheduling customer service agents, optimising supply chains, controlling dynamical systems, modelling biological processes, finding optimal strategies in negotiation, and extracting meaningful components from brain signals.

The master's programme Data Science for Decision Making covers a wide range of research topics, focusing on the following ones in its core:

1. Data mining to extract useful patterns and knowledge from large data repositories;
2. Mathematical modelling and parameter estimation from data, system identification, model approximation and reduction of model complexity;
3. Mathematics and algorithm design and analysis to efficiently deal with the challenges that the ever-growing amount of data pose;
4. Statistical analysis, in the computational sense, of data.

The members of the teaching staff are actively involved in one or more of the research topics. As a result, the educational contents of the courses relate directly to the research performed.

Programme master's AI

Year 1		ECTS
Period 1	Intelligent Search & Games (KEN4123) *1 elective of the following courses:* Data Mining (KEN4113) Foundations of Agents (KEN4115) **** Restricted access Signal and Image Processing (KEN4222) Stochastic Decision-Making (KEN4221) Research Project 1 (**)	6 6 6 6 6
Period 2	Advanced Concepts in Machine Learning (KEN4154) *1 elective course from the following courses:* Deep Learning for Image and Video Processing (KEN4244) Advanced Natural Language Processing (KEN4259) Research Project 1 (**)	6 6 6
Period 3	Research Project AI (KEN4230)	6
Period 4	Agents and Multi-Agent Systems (Code: KEN4111) *1 elective course from the following courses:* Building and Mining Knowledge Graphs (KEN4256) Computational Statistics (KEN4258) Dynamic Game Theory (KEN4251) Planning and Scheduling (KEN4253) Explainable AI (KEN4246) Research Project 2 (**)	6 6 6 6 6 6
Period 5	Autonomous Robotic Systems (KEN4114) *1 elective course from the following courses:* Computer Vision (KEN4255) Information Retrieval and Text Mining (KEN4153) Introduction to Quantum Computing for AI and DS (KEN4155) Reinforcement Learning (KEN4157) Research Project 2 (**)	6 6 6 6 6
Period 6	Research Project 2 (KEN4131)	6

Year 2		ECTS
Period 1, 2, 3	Electives *	30
Period 4, 5, 6	Master’s thesis AI (KEN4160)	30

Programme master’s DSDM

Year 1		ECTS
Period 1	1 Data Mining (KEN4113) *1 elective course from the following courses:* Signal and Image Processing (KEN4222) Mathematical Optimization (KEN4211) Stochastic Decision-Making (KEN4221) Research Project 1 (**)	6 6 6 6
Period 2	Model Identification and Data Fitting (KEN4242) *1 elective course from the following courses:* Advanced Concepts in Machine Learning (KEN4154) Deep Learning for Image and Video Processing (KEN4244) Advanced Natural Language Processing (KEN4259) Research Project 1 (**)	6 6 6 6
Period 3	Research Project 1 (KEN4230)	6
Period 4	Computational Statistics (KEN4258) *1 elective course from the following courses:* Building and Mining Knowledge Graphs (KEN4256) Dynamic Game Theory (KEN4251) Planning and Scheduling (KEN4253) Data Fusion (KEN4223) Explainable AI (KEN4246) Research Project 2 (**)	6 6 6 6 6 6
Period 5	Algorithms for Big Data (KEN4254) *1 elective course from the following courses* Symbolic Computation and Control (KEN4252) Information Retrieval and Text Mining (KEN4153) Computer Vision (KEN4255) Introduction to Quantum Computing for AI and DS (KEN4155) Research Project 2 (**)	6 6 6 6 6
Period 6	Research Project 2 (KEN4231)	6

Year 2		ECTS
Period 1, 2, 3	Electives *	30
Period 4, 5, 6	Master’s thesis DSDM (KEN4260)	30

** Note: during the elective semester (first semester of year 2) of the master’s programme it is possible to take electives from our other master’s programme or relevant master’s programmes at Maastricht University (maximum of 13 ECTS outside the Department of Advanced Computing Sciences) or to participate in a research project, a business internship or a study abroad semester at one of our partner universities. Please contact exchange officer and/or the Study Adviser for more information.*

*** The Research Project 1 will start in period 1.1 and 1.2 with weekly meetings.*

The credits for the project will become available at the end of period 1.3. The Research Project 2 will start in period 1.4 and 1.5 with weekly meetings. The credits for the project will become available at the end of period 1.6.

Part of the Research project is a Project skill programme, offered in September and February (all elements are mandatory):			
Period 1.1:	Session 1 Session 2 Session 3 Session 4 Session 5 Session 6	(1 hour) (3 hours) (2 hours) (2 hours) (2 hours) (2hours)	Introduction to Project-Centred Learning and LaTeX Team Building afternoon Discover your Competences Part 1 (Career Services) Discover your Competences Part 2 (Career Services) Presentation skills Presentation skills
Period 1.2	Session 1 Session 2	(3 hours) (3 hours)	Academic Writing Skills A workshop of choice at Career Services

Note: Master's students who have finished their bachelor's degree at the the Department of Advanced Computing Sciences and already participated in the introduction to PCL and LaTeX do not need to partake in session 1, period 1.1. Moreover, if these students can show that they already took the workshop "discover your competences" they may request for an exemption with the study advisor. Second year students that have not participated in this project skill programme are free to follow a selection of sessions.

Period 1.1

Intelligent Search & Games (KEN4123)

Examiners: Prof. dr. Mark Winands and dr. Cameron Browne.

Desired Prior Knowledge: Data Structures & Algorithms

Course description: In this course, the students learn how to apply advanced techniques in the framework of game-playing programs. Depending on the nature of the game, these techniques can be of a more or less algorithmic nature. The following subjects will be discussed:
1. Basic search techniques. Alpha-beta; A*.
2. Advanced search techniques. IDA*; B*, transposition tables; retrograde analysis and endgame databases; proof-number search and variants; multi-player search methods; Expectimax and *-minimax variants.
3. Heuristics. World representations; killer moves; history heuristic, PVS; windowing techniques; null-moves; forward-pruning techniques; selective search, GOAP.
4. Monte Carlo methods. Monte Carlo tree search (MCTS) techniques, enhancements, and applications; AlphaGo and AlphaZero approaches.
5. Game design. Evolutionary game design; game quality metrics; self-play evaluation; puzzle design.

Knowledge and understanding: The student can explain basic and advanced search techniques and can identify which of them to use either in a game context, or in problems with a similar structure.

Applying knowledge and understanding: Students have obtained the knowledge to develop, program, analyse, and apply advanced techniques autonomously to a wide variety of problems. They will also learn that adapting known techniques to fit a given problem can achieve a better performance.

Making judgements: Students will be able to judge the quality of approaches (systems or scientific publications) based on the techniques taught.

Communication: Students will be able to present the results of their game programs and search algorithms to specialists or non-specialists.

Learning skills: Students will be able to familiarize themselves with Game AI techniques beyond the scope of the course in order to solve a problem.

Study material: Course notes and other information made available.

Recommended Literature:
- Millington, I. (2019). Artificial Intelligence for Games, 3rd Edition, CRC Press, ISBN: 978-1138483972
- Russell, S.J. and Norvig, P. (2020). Artificial Intelligence: A Modern Approach, 4th edition.
- Pearson. ISBN 0-13-461099-7.
- Yannakakis, G.N. and Togelius, J. (2018) Artificial Intelligence and Games, Springer, Berlin. ISBN 978-3-319-63519-4 (eBook) 978-3-319-63518-7 (hardcover)

Exam: Written exam (50%) + a large practical task (50%).

ECTS: 6

Data Mining (KEN4113)

Examiners: Dr. Evgueni Smirnov

Desired Prior Knowledge: Statistics and Basic Machine Learning

Prerequisites: None.

Course description: Data mining is a major frontier field of computer science. It allows extracting useful and interesting patterns and knowledge from large data repositories such as databases and the Web. Data mining integrates techniques from the fields of databases, machine learning, statistics, and artificial intelligence. This course will present the state-of-the-art techniques of data mining. The lectures and labs will emphasize the practical use of the presented techniques and the problems of developing real data-mining applications. A step-by-step introduction to data-mining environments will enable the students to achieve specific skills, autonomy, and hands-on experience. A number of real data sets will be analysed and discussed.

Knowledge and understanding: Students will acquire knowledge on data preparation, data preprocessing, feature selection/generation, data mining, and model validation.

Applying knowledge and understanding: When confronted with real-life problems, students will be able to identify data-analysis tasks. Then, they will be able to apply data-mining techniques for supervised and unsupervised data-analysis. If necessary, students will be able to design data-mining algorithms specific for the tasks they have.

Making judgements: Students will be able to assess the quality of data-mining models, processes, results, and tools.

Communication: Students will be able to present the results of different stages of data-mining processes to specialists or non-specialists.

Learning skills: Students will be able to recognize their own lack of knowledge and understanding and take appropriate action such as consulting additional material or other sources of help.

Study material: Course notes, slides, and other information made available.

Recommended Literature: Pang-Ning, T., Steinbach, M., Karpatne, A., and Kumar, V. (2018). Introduction to Data Mining, 2nd Edition, Pearson, ISBN-10: 0133128903, ISBN-13: 978-0133128901

Exam: Written exam (80%) + practical assignments (20%).

ECTS: 6

Foundations of Agents (KEN4115) **** Restricted access

Examiner: Prof. Dr. ir. Nico Roos.

Desired Prior Knowledge: Logic, Calculus, and Probability Theory.

Prerequisites: A basic course in logic and in probability theory.

Description: Agents are autonomous computer programs, robots, humans, etc. Agents operate in some environment, which they can observe, and in which they can realize objectives through the execution of actions. Examples of environment in which agents can operate, are computer game environments, the internet, and also the physical world is case of robots and humans.

In this course we address the problem of how an agent can act optimally in order to realize its objectives. We will answer this question by investigating how we can formally specify the agent's environment, the agent's objectives, the observations the agent can make and the actions it can execute. We use the formal model to investigate how the agent can determine an (optimal) behaviour realizing its objectives.

The following formal models will be investigated:
- Markov Decision Processes,
- Partially Observable Markov Decision Process,
- logic-based models such as Epistemic Logic, Doxastic Logic, Dynamic Logic, and BDI logics, and
- Game Theory.

Some examples of methods for determining the agents optimal behaviour addressed in the course are: Value and Policy Iteration, Q-Learning, Planning, etc.

Knowledge and understanding:
- The student will be able to explain formal models for describing agents.
- The student will be able to verify whether the formal models are correct.
- The student will be able to explain the underlying assumptions of each formal model.
- The student will be able to compare and discuss the differences between the formal models.
- The student will be able to analyze important properties of formal models for describing agent.

Applying knowledge and understanding:
- The student will be able to apply formal models for describing agents to solve practical problems.

- The student will be able to assess which formal model is most suited for addressing a practical problem.
- The student will be able to make a practical implementation of a formal model.

Making judgments:
- The student will be able to analyze which formal model of an agent is adequate for specific problem domains.
- The student will be able to select the formal model of an agent that is adequate for specific problem domains.

Communication:
- The student will be able to explain to its peers why a formal model of an agent is adequate for specific problem domains.
- The student will be able to explain to its peers how a formal model of an agent should be applied for a specific problem.

Learning skills:
- The student will be able to explain to its peers why a formal model of an agent is adequate for specific problem domains.
- The student will be able to explain to its peers how a formal model of an agent should be applied for a specific problem.

Study material: Syllabi, scientific papers.

Recommended Literature: none.

Examination: Written exam at the end of the course. A bonus of 1.0 point can be earned by a series of bonus assignments.

ECTS: 6

Mathematical Optimization (Code KEN4211)

Examiner: Dr. Pieter Collins

Desired prior knowledge: Calculus, Linear Algebra, Linear Programming

Description: Optimisation is the subject of finding the best or optimal solution to a problem from a set of potential or feasible solutions. Optimisation problems are fundamental in all forms of decision-making, since one wishes to make the best decision in any context, and in the analysis of data, where one wishes to find the best model describing experimental data. This course treats two different areas of optimisation: nonlinear optimisation and combinatorial optimisation, building on knowledge of linear programming using the simplex algorithm. Together, nonlinear and combinatorial optimisation cover a wide range of real life optimisation problems. Nonlinear optimisation deals with the situation that there is a continuum of available solutions. A best solution is then usually approximated with one of several available general purpose algorithms, such as Brent's method for one-dimensional problems, (quasi-)Newton and conjugate gradient methods for unconstrained problems, and Lagrangian methods, including active-set methods, sequential quadratic programming and interior-point methods for general constrained problems. Combinatorial optimisation deals with situations that a best solution from a discrete set of available choices must be found. A variety of techniques, such as linear programming, branch and cut, Lagrange relaxation and approximation algorithms are employed to tackle this type of problem. Throughout the course, we aim to provide a coherent framework for the subject, with a focus on optimality conditions (notably the Karush-Kuhn-Tucker conditions), Lagrange multipliers and duality, relaxation and approximate problems, and on convergence rates and computational complexity. The methods will be illustrated by in-class computer demonstrations, exercises illustrating the main concepts and algorithms, modelling and computational work on case studies of practical interest, and a discussion of advanced stochastic and batch optimization methods for machine-learning.

Knowledge and understanding: By the end of this course, students will have a strong foundation in nonlinear and combinatorial optimisation. You will be able to formulate real-life problems as optimisation problems. You will understand optimality conditions, including the Karush-Kuhn-Tucker conditions and be able to test for optimality. You will know how to solve a variety of general optimisation problems, including constrained nonlinear problems, and (mixed-)integer linear problems. You will understand notions of duality and Lagrange multipliers, and be able to apply techniques based on relaxation and approximation.

Applying knowledge and understanding: Students will know the advantages and disadvantages of different methods, and be able to choose an appropriate method for a given problem. You will be able to implement and test optimisation algorithms on a computer. You will be able to apply your knowledge to the solution of practical problems and in developing new efficient algorithms.

Making judgements: Students will be able to select an appropriate solution method for a given optimization problem, and judge the quality of the solution obtained.

Communication: Students will be able to discuss the development and use of optimization algorithms.

Learning skills: Students will learn how to develop and implement mathematical methods for optimisation, select and evaluate algorithms, and formulate mathematical model of real-world problems.

Study material: Lecture notes, handouts.

Recommended literature:

1. Nocedal and Wright, *"Numerical Optimization"* (Springer).
2a. Papadimitriou and Steiglitz, *"Combinatorial Optimization: Algorithms and Complexity"* (Dover Publications).
2b. Cook, Cunningham, Pulleyblank and Schrijver, *"Combinatorial Optimization"*, (Wiley-Interscience).

Exam: Written exam, closed book (100%)

ECTS: 6

Signal and Image Processing (KEN4222)

Examiner: Dr Joel Karel and dr. Pietro Bonizzi

Desired Prior Knowledge: Linear algebra, Calculus, basic knowledge of Matlab. Some familiarity with linear systems theory and transforms (such as Fourier and Laplace) is helpful.

Prerequisites: None.

Course description: This course offers the student a hands-on introduction into the area of digital signal and image processing. We start with the fundamental concepts and mathematical foundation. This includes a brief review of Fourier analysis, z-transforms and digital filters. Classical filtering from a linear systems perspective is discussed. Nextwavelet transforms and principal component analysis are introduced. Wavelets are used to deal with morphological structures in signals. Principal component analysis is used to extract information from high-dimensional datasets. We then discuss Hilbert-Huang Transform to perform detailed time-frequency analysis of signals. Attention is given to a variety of objectives, such as detection, noise removal, compression, prediction, reconstruction and feature extraction. We discuss a few cases from biomedical engineering, for instance involving ECG and EEG signals. The techniques are explained for both 1D and 2D (images) signal processing. The subject matter is clarified through exercises and examples involving various applications. In the practical classes, students will apply the techniques discussed in the lectures using the software package Matlab.

Knowledge and understanding: Students are able to explain fundamental concepts of signal and image processing and their mathematical foundation. They are able to 1) describe various types of filters and their properties, 2) explain orthogonal wavelet filter banks and describe their properties, 3) explain a construction scheme and elicit a wavelet-based noise-filtering scheme, 4) explain principal component analysis and empirical signal processing techniques and how they complement the other techniques discussed.

Applying knowledge and understanding: Students are able to use the various techniques discussed during the lectures to solve real-world problems, such as being able to apply wavelet filtering and principal component analysis on various signals. They are also able to analyse a signal by using Matlab, and independently interpret the outcome of an analysis.

Making judgements: Students are able to assess what technique is suited for a signal processing problem at hand, and to independently and critically look at a signal or image, and understand if and what type of pre-processing is required.

Communication: Students are able to communicate signal and image processing techniques and strategies, and the results of their analyses to experts and non-experts.

Learning skills: Students are able to independently master signal and image processing techniques, from classical signal processing techniques to more empirical techniques.

Study material: Discrete Wavelet Transformations: An Elementary Approach with Applications, Patrick J. Van Fleet, Wiley, ISBN: 978-0-470-18311-3.
Additional material provided electronically on Student Portal.

Recommended literature: Principal Component Analysis, Ian T. Jolliffe, Springer, ISBN13: 978-0387954424.

Exam: Written exam/Computer exam.

ECTS: 6

Stochastic Decision Making (KEN4221)

Examiner: Dr. Gijs Schoenmakers

Prerequisites: Probability & Statistics.

Course Description: Any realistic model of a real-world phenomenon must take into account the possibility of randomness. That is, more often than not, the quantities we are interested in will not be predictable in advance but, rather, will exhibit an inherent variation that should be taken into account by the model. Mathematically, this is usually accomplished by allowing the model to be probabilistic in nature. In this course, the following topics will be discussed:

1. Basic concepts of probability theory: Probabilities, conditional probabilities, random variables, probability distribution functions, density functions, expectations and variances.
2. Finding probabilities, expectations and variances of random variables in complex probabilistic experiments.
3. Discrete and continuous time Markov chains and related stochastic processes like random walks, branching processes, Poisson processes, birth and death processes, queueing theory.
4. Markov decision problems.
5. Multi-armed bandit problems, bandit algorithms, contextual bandits, cumulative regret, and simple regret

Knowledge and understanding: In this course, the students acquire tools for modelling complex processes involving randomness, providing a basis for originality in developing and/or applying ideas in a research context.

Applying knowledge and understanding: When confronted with complex problems that involve probabilistic experiments, students have the tools to create and analyse appropriate models.

Making judgements: The students are able to analyse complex problems as stochastic processes and solve them. Furthermore, students can find optimal solutions in decision problems that are based on these stochastic processes.

Communication: The students will be able to communicate their conclusions and the underlying rationale to expert and non-expert audiences.

Learning Skills: The students have obtained the skills to study related material in a largely autonomous manner.

Study material: Introduction to Probability Models by Sheldon M. Ross (9 th or 10th ed.) + Lecture notes that are provided via Student Portal.

Recommended Literature: Probability: A Lively Introduction by Henk Tijms. Reinforcement Learning by Richard S. Sutton and Andrew G. Barto (2nd ed.) (chapter 2); Bandit Algorithms by Tor Lattimore and Csaba Szepesvári

Exam: Written exam.

ECTS: 6

Period 1.2

Advanced Concepts in Machine Learning (KEN4154)

Examiner: Dr. ir. Kurt Driessens & Dr. Enrique Hortal

Desired Prior Knowledge: Machine Learning

Prerequisites: None.

Description: This course will introduce a number of advanced concepts in the field of machine learning such as Support Vector Machines, Gaussian Processes, Deep Neural Networks, Quantum ML, etc. All of these are approached from the view that the right data representation is imperative for machine learning solutions. Additionally, different knowledge representation formats used in machine learning are introduced. This course counts on the fact that basics of machine learning were introduced in other courses so that it can focus on more recent developments and state of the art in machine learning research. Labs and assignments will give the students the opportunity to implement or work with these techniques and will require them to read and understand published scientific papers from recent Machine Learning conferences.

Knowledge and understanding: Students can explain, construct and adapt powerful machine learning techniques, most with a statistical background. Students recognise the need for non-standard techniques and representations that can be used for complex/structured data. They can explain the strengths and weaknesses of different machine learning approaches.

Applying knowledge and understanding: Students will be able to select, adapt and apply a number of advanced machine learning approaches. They will be able to select the correct representation for a machine-learning problem and to translate a machine learning problem into a suited representational format.

Making judgements: Students will be able to judge which machine learning approach and data-representation is best suited. They will also be able to comprehend and judge machine-learning research.

Communication: Students will be able to relate different machine learning techniques to each other and explain their working, benefits and disadvantages to non-experts. They will also be able to discuss the need and use of structured representation with both experts and non-experts.

Learning Skills: Students will be able to relate information from different sources, and readm, process and evaluate recent research developments in the field of machine learning.

Study material: Slides that support the lectures and collected notes and chapters from freely available books and course notes.

Recommended literature: Pattern Recognition and Machine Learning - C.M. Bishop; Bayesian Reasoning and Machine Learning - D. Barber; Gaussian Processes for Machine Learning - C.E. Rasmussen & C. Williams; The Elements of Statistical Learning - T. Hastie et al.; Machine Learning with Quantum Computers by Francesco Petruccione and Maria Schuld

Exam: Students are graded using a number of assignments (20%) and a written exam (80%).

ECTS: 6

Deep Learning for Image and Video Processing (KEN4244)

Examiner: tba

Desired prior knowledge: Image and Video Processing, Calculus, Linear Algebra, Machine Learning.

Prerequisites: None.

Description: Applications of image and video processing will be presented, and connections to basic algorithms will be demonstrated. We will examine some of the most popular and widespread applications, namely security, surveillance, medical, traffic monitoring, astronomy, farming, culture. The methods used in these applications will be analysed in class and common characteristics between them will be explained. Deep learning, as one of the most successful method, will be covered in detail. Students will be able to suggest further applications of interest to them and bring relevant literature to the class.

Knowledge and understanding: Students will acquire a wide-ranging understanding of the latest trends in image and video processing methods, and, in particular, of deep learning, and how these are applied in real world applications. They will obtain insights on common problems encountered in these applications, and how they can be tackled through advanced image and video processing algorithms.

Applying knowledge and understanding: The knowledge and understanding obtained in this class will be demonstrated in mini projects based on State of the Art research.

Making judgements: Through the presentation of various applications of image and video processing, students will be able to analyse problems in the real world, and understand how to best address them.

Communication: Part of the class will include homework where students will carry out a short literature review and implementation of mini projects on applications that interest them. They will be taught how to communicate them succinctly and effectively, maintaining a balance between overall understanding and technical depth.

Learning Skills: Students will obtain a spherical comprehension of connections between machine learning and image/video/signal processing, as well as their practical in a wide range of applications in our daily life. They will be able to identify the methods needed in different applications of image and video processing, propose a plan for solving the corresponding problems, and justify it.

Study material: Lecture slides, selected papers. A Bovik, Handbook of Image and Video processing

Recommended literature:
Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing (3rd Edition), Prentice Hall.
A. Bovik (Ed.), The Essential Guide to Video Processing. Academic Press, 2009.

Exam: Mini projects (50%) and final exam (50%).

ECTS: 6

Model Identification and Data Fitting (KEN4242)

Examiners: Prof dr. ir. Ralf Peeters and dr. Philippe Dreesen

Tutor(s): None.

Desired Prior Knowledge: Basic knowledge of Matlab and some familiarity with mathematical modelling and systems, and transforms (such as Fourier and Laplace) is helpful. This course offers a useful prior knowledge for the course Symbolic Computation and Control.

Prerequisites: Linear Algebra, Mathematical Modelling, Probability and Statistics.

Course description: This course is devoted to the various practical and theoretical aspects which involve the estimation (the identification) of a mathematical model within a given model class, starting from a record of observed measurement data (input-output data). First, we address distance measures, norms, and criterion functions. Then we discuss the prediction error identification of linear regression models, with special emphasis on the various interpretations of such models (deterministic, stochastic with Gaussian white noise and maximum likelihood estimation, stochastic in a Bayesian estimation context) and on numerical implementation aspects (recursion, numerical complexity, numerical conditioning and square root filtering). Next, we study identification within the important class of auto-regressive dynamical models, to which the Levinson algorithm applies. Other related topics receiving attention are identifiability, model reduction, and model approximation. Some techniques for the estimation of linear dynamical i/o-systems are illustrated with the system identification toolbox in Matlab.

Knowledge and understanding: Students learn to recognize the various aspects that play a key role in building a mathematical model from measurement data: the choice of model class (and order), the choice of parameterization, the criterion of fit, the model estimation method, the quality of the measurement data, and the validity of the estimated model.

Applying knowledge and understanding: Students are able to:
1. estimate models from measurement data, particularly linear regression models and auto-regressive models,
2. to assess the quality of a (linear regression) model, and
3. assess whether a model is identifiable.

Making judgements: Students are able to predict and judge the quality of models that can be obtained from a record of measurement data.

Communication: Students learn to motivate the choice of a model class, the model order and an estimation method to identify a model from measurement data, to interpret the identification outcomes and to explain all this to specialists and non-specialists.

Learning skills: Students are able to read and interpret scientific literature on model estimation and system identification, and to use Matlab and work out ideas computationally.

Study material: Syllabus, provided electronically on the digital learning environment.

Recommended literature:
- L. Ljung, System Identification: Theory for the User (2nd ed.), Prentice-Hall, 1999.
- T. Soderstrom and P. Stoica, System Identification, Prentice-Hall, 1989.

Exam: Written exam.

ECTS: 6

Advanced Natural Language Processing (KEN4259)

Examiner: prof. dr. Jan Scholtes and dr. Aki Härmä

Desired prior knowledge: none

Prerequisites: none

Description: For decades, teaching a computer to deal with natural language processing (NLP) was a long-time dream of humankind. Task such as machine translation, summarization, question-answering, speech recognition or chatting remained a challenge for computer program. Around 2020, major improvements were made. Starting with machine translation and ultimately in late 2022 with ChatGPT. Why were these large-language models suddenly so good? How did we get here? What can we do with these new algorithms to improve them even more?

This course will provide the skills and knowledge to understand and develop state-of-the-art (SOTA) solutions for these natural language processing (NLP) tasks. After a short introduction to traditional generative grammars and statistical approaches to NLP, the course will focus on deep learning techniques. We will discuss Transformers, variations on their architecture (including BERT and GPT) in depth, which models works best for which tasks, their capacities, limitations and how to optimize these.

Although that we have algorithms that can deal with Natural Language Processing in ways that can no longer be distinguished from humans, we still have some major problems to address: (i) we do not fully understand what these algorithms know and what they do not know. So, there is a strong need for eXplainable AI (XAI) in NLP. (ii) Training the deep-learning large language-models costs too much energy. We need to develop models that are less computationally (and thus energy) intensive. (iii) Now that these algorithms operate at human-level quality, several ethical problems arise related to computer generated fake-news, fake profiles, bias, and other abuse. But there are also ethical, legal, regulatory and privacy challenges. In this courses, these important topics will also be discussed.

This course is closely related with the course Information Retrieval and Text-Mining (IRTM). In this course the focus is more on advanced methods and architectures to deal with complex natural language tasks. The IRTM course focusses more on building search engines and text-analytics, but also uses a number of the architectures which are discussed in more depth in this course. The overlap between the two courses is kept to a minimum. There is no need to follow the courses in a specific order.

Knowledge and understanding: Students will be taught state-of-the-art deep learning techniques for natural language processing. Students will understand why deep-learning models are so successful dealing with natural language. They will learn techniques to address the major challenges when building a natural language processing tool.

Appling knowledge and understanding: The achievements in deep learning have significantly improved the quality of state-of-the-art methods for natural language processing. With the knowledge acquired in the course, students will be able to build SOTA solutions.

Making Judgements: Students will be able to analyze the specific challenges of a task in NLP. Based on the gathered knowledge on different ways to model tasks they are able to select and implement a fitting model to solve the task.

Communication: Through small research projects, students will be enabled to communicate their findings and explain the rationale behind their choices in deep learning techniques for natural language processing.

Learning Skills: After successful completion of the course, students will be able to develop natural language processing tools and perform research on new ideas in the field.

Study material: Mostly based on the lecture notes and the provided material including recent papers published in this field.

Recommended literature: Papers published in top international conferences and journals in machine learning field.

Exam: The mandatory completion of a number of tutorials and a written exam.

ECTS: 6

Period 1.3

Research Project AI 1 & DSDM 1 (Code KEN4130 & KEN4230)

Examiner: T.b.a.

Coordinator: Dr. Linda Rieswijk

Desired Prior Knowledge: None.

Prerequisites: None.

Description: The research project takes place during the three periods of the semester. Project topics are presented at the start of the semester and assigned to students based on their preferences and availability. The emphasis in the first phase is on initial study of the context set out for the project and the development of a project plan. In the second period, the goal is to start modelling, prototyping and developing. In period 3, the implementation, model and/or experiments set out in the project plan has to be finished and reported on. At the end of period 1 and 2, a progress presentation takes place. The project results in a project presentation, a project report and possibly a public website and/or product.

Knowledge and understanding: Students get to know and possibly contribute to state of the art methods within the fields of Artificial Intelligence and/or Data Science for Decision Making to answer an open question.

Applying knowledge and understanding: Students write their own research plan in coordination with a staff member (plus possibly outsiders) who act as clients with an open question. Students with different backgrounds and from both masters work together in teams to build and evaluate an answer to an open question. Students find, judge the suitability, apply, and evaluate state of the art techniques to answer questions and construct applications in the field of Artificial Intelligence and Data Science. Students apply the accumulated knowledge from other educational activities in application specific areas

Making judgements: Students judge feasibility of tasks, attainability of goals, and the amount of work involved. Students think about the possible consequences of their work. Students evaluate state of the art and the applicability and scope of research results.

Communication: Students will learn to:
1. orally communicate and cooperate with peers
2. orally report on progress and intermediate results to superiors
3. orally negotiate and communicate with clients
4. communicate their ideas in written form, both for an academic and a general audience
5. give effective presentations

Learning Skills: Students increase their own level of knowledge in a specialised sub-discipline of the field of Artificial Intelligence and/or Data Science. Students perform research into recent state of the art techniques. Students learn that the field of Artificial Intelligence and Data Science are constantly evolving beyond what is taught in class

Study material: Slides provided at the end of joint information sessions. Literature provided by the project supervisors.

Recommended literature: Justin Zobel (2004), Writing for Computer Science, Springer, ISBN:1852338024

Exam:
Phase 1: project plan + presentation (15%);
Phase 2: layman's website + presentation (15%);
Phase 3: Project report + presentation (70%)

ECTS: 6

Period 1.4

Agents and Multi-Agent Systems (Code: KEN4111)

Examiner: Prof. dr. Gerard Weiss.

Desired Prior Knowledge: Basic knowledge and skills in programming..

Description: The notion of an (intelligent) agent is fundamental to the field of artificial intelligence. Thereby an agent is viewed as a computational entity such as a software program or a robot that is situated in some environment and that to some extent is able to act autonomously in order to achieve its design objectives. The course covers important conceptual, theoretical and practical foundations of single-agent systems (where the focus is on agent-environment interaction) and multi-agent systems (where the focus is on agent-agent interaction). Both types of agent-based systems have found their way to real-world applications in a variety of domains such as e-commerce, logistics, supply chain management, telecommunication, health care, and manufacturing. Examples of topics treated in the course are agent architectures, computational autonomy, game-theoretic principles of agent-based systems, coordination mechanisms (including auctions and voting), and automated negotiation and argumentation. Other topics such as ethical or legal aspects raised by computational agency may also be covered. In the exercises and in the practical part of the course students have the opportunity to apply the covered concepts and methods.

Knowledge and understanding: The student is able to describe and explain single- and multi-agent concepts and methods, and to analyse their strengths and shortcomings.

Applying Knowledge and Understanding: The student is able to apply the gained knowledge in concrete application scenarios and practical applications.

Making Judgements: The student is able to judge for a given problem whether and in how far it is beneficial to use a multi-agent approach for its solution.

Communication: The student is able to motivate and explain benefits and shortcomings of their usage in a given application, and thereby showing sufficient understanding of multi-agent concepts.

Learning Skills: The student is able to study independently and critically literature on single- and multi-agent technology, including, in particular, literature describing new developments in the methods and techniques covered in this course.

Study material: Course slides; supplementary material to be announced.

Recommended literature:
- Stuart Russell and Peter Norvig (2010). Artificial Intelligence. A Modern Approach. 3rd edition. Prentice Hall.
- Gerhard Weiss (Ed.) (2013, 2nd edition): Multi-agent Systems. MIT Press.
- Mike Wooldridge (2009, 2nd edition): An Introduction to Multi Agent Systems, John Wiley & Sons Ltd.
- Yoav Shoham and Kevin Leyton-Brown (2009): Multi-agent Systems. Algorithmic, Game-Theoretic, and Logical Foundations, Cambridge University Press.

Examination: Practical and reading assignments (30%) and written exam (70%)

ECTS: 6

Building and Mining Knowledge Graphs (KEN4256)

Examiner: Prof. dr. Christopher Brewster and dr. Kody Moodley

Tutor(s): None.

Desired Prior Knowledge: Introduction to Computer Science

Prerequisites: None.

Description: Knowledge graphs are large-scale, machine-processable representations of entities, their attributes, and their relationships. Knowledge graphs enable both people and machines to explore, understand, and reuse information in a wide variety of applications such as answering questions, finding relevant content, understanding social structures, and making scientific discoveries. However, the sheer size and complexity of these graphs present a formidable challenge particularly when mining across different topic areas.
In this course, we will examine approaches to construct and use knowledge graphs across a diverse set of applications using cutting-edge technologies such as machine learning and deep learning, graph databases, ontologies and automated reasoning, and other relevant techniques in the area of data mining and knowledge representation.

Knowledge and understanding: Students will be able to describe:
- The nature and attributes of a Knowledge Graph
- Examples of Knowledge Graphs;

- Representations for Knowledge Graphs
- Applications of Knowledge Graphs
- Advantages and disadvantages of Knowledge Graphs as compared to other formalisms
- Approaches and challenges in constructing and maintaining Knowledge Graphs
- Approaches and challenges in finding, using and mining Knowledge graphs
- What FAIR is and how it relates to Knowledge Graphs
- Ethical, legal & social issues around Knowledge Graphs

Applying knowledge and understanding: Students will be able to identify requirements and steps to convert knowledge in traditional data formats to Knowledge Graph formats. Students will also be able to implement such strategies. Students will be able to query Knowledge Graphs (for instance using SPARQL query language) to answer basic to intermediately advanced questions. Students will be able to implement basic reasoning strategies on Knowledge Graphs to answer intermediately advanced questions, which cannot be answered by SPARQL queries alone. Students will be able to implement popular methods to integrate different data sources by transferring them into a Knowledge Graph. Students will be able to enrich existing Knowledge Graphs with missing information using basic predictive algorithms. Students will be able to perform basic data quality assessment on Knowledge Graphs. Students will be able to assess the degree of compliance that Knowledge Graphs have with FAIR principles

Making judgements: Students will be able to select which tools are most suitable for constructing, querying, visualising & reasoning with Knowledge Graphs. Students will be able to differentiate between different types of Knowledge Graphs, according to their representation, coverage and content. Students will be able to select which Knowledge Graph is appropriate for answering a particular question. Students will be able to diagnose incompleteness in a Knowledge Graph with respect to answering a particular question. Students will be able to evaluate the data quality and FAIRness of a Knowledge Graph.

Communication: Students will be able to explain the advantages of representing information on the web in Knowledge Graphs. Students will be able to communicate the steps required to convert information to a Knowledge Graph format. Students will be able to communicate to non-experts the main content and representational components of a Knowledge Graph. Students will be able to outline to non-experts the steps required to answer a question by querying a Knowledge Graph.

Learning Skills: Students will be able to reflect critically on the challenges and open problems remaining in Knowledge Graphs research. Students will be able to formulate and propose strategies to answer complex questions using Knowledge Graphs. Students will be able to assess the feasibility of different combinations of methods for answering questions using Knowledge Graphs.

Study material: Material will be provided during the course in the form of handouts.

Recommended literature: Aggarwal, C.C. and Wang, H. eds., (2010) Managing and mining graph data (Vol. 40). New York: Springer. ISBN 978-1-4419-6045-0

Exam: Individual project for application of knowledge and two assignments to demonstrate understanding of core concepts.

ECTS: 6

Computational Statistics (KEN4258)

Examiner: Dr. Christof Seiler

Desired prior knowledge: Probability and Statistics

Prerequisites: None

Description: In this course, we will review basic concepts in statistical inference (confidence intervals, parameter estimation, and hypothesis testing). We will then study computer-intensive methods that work without imposing unrealistic or unverifiable assumptions about the data generating mechanism (randomization tests, the bootstrap, and Markov chain Monte Carlo). This will provide us with the foundations to study modern inference problems in statistics and machine learning (false discovery rates, Benjamini-Hochberg procedure, and causal inference).

Knowledge and understanding: Knowing a wide range of modern statistical models and computational tools to draw inferences will provide the foundations for analyzing complex data in academia and industry.

Applying knowledge and understanding: Students will be able to build statistical models and choose computational tools to perform inference.

Making judgements: In this course, we will discuss one of the most important aspects of analyzing data: being skeptical of results and avoiding wishful thinking.

Communication: Students will present their results using literate programming and reproducible workflows.

Learning skills: Students will be able to understand, apply, and extend papers from statistics journals and machine learning conferences.

Study material: Lecture slides, selected chapters from textbooks, and papers.

Exam: 20% homework assignments and 80% written final exam.

ECTS: 6

Dynamic Game Theory (KEN4251)

Examiner: Prof. dr. Frank Thuijsman, dr. Monica Savioli

Desired Prior Knowledge: Students are expected to be familiar with basic concepts from linear algebra, calculus, Markov chains and differential equations.

Prerequisites: None

Description: The course will focus on non-cooperative games and on dynamic games in the following order: matrix and bimatrix games, repeated games, Stackelberg games, differential games, specific models of stochastic games, evolutionary games. These are games in which the players are acting as strategic decision makers, who cannot make binding agreements to achieve their goals. Instead, threats may be applied to establish stable outcomes. Besides, relations with population dynamics and with "learning" will be examined. Several examples will be taken from biological settings.

Knowledge and understanding: SStudents are able to recognize and classify the main types of dynamic games, i.e. repeated games, stochastic games, Stackelberg games, differential games, and evolutionary games and formulate the main solution concepts value, optimal strategies, Nash- and Stackelberg equilibrium

Data Fusion (KEN4223)

Examiners: prof. Anna Wilbik

Desired prior knowledge: statistics and basic machine learning

Prerequisites: none

Course description: ICT development, e.g., remote sensing, IoT, lead to an enormous growth of available data for analysis. To integrate this heterogeneous or multimodal data, data fusion approaches are used. Data fusion can be understood as a framework for the joint analysis of data from multiple sources (modalities) that allows achieving information/knowledge not recoverable by the individual ones.

During this course, several approaches to data fusion will be discussed, such as:
1. Low level data fusion, where data fusion methods are directly applied to raw data sets for exploratory or predictive purposes. A main advantage is the possibility to interpret the results directly in terms of the original variables. An example of a low level data fusion is measuring the same signal or phenomena with different sensors, in order to discover the original one. Traditionally, PCA based methods are used for this type of data fusion.
2. Mid level data fusion, where data fusion operates on features extracted from each data set. The obtained features are then fused in a "new" data set, which is modeled to produce the desired outcome. A main advantage is that the variance can be removed in the features extraction step, and thus the final models may show better performance. An example of a mid level data fusion is extracting numerical features from an image, and building a decision model based on those features.
3. High level data fusion, also known as decision fusion, where decisions (models outcome) from processing of each data set are fused. It is used when the main objective is to improve the performance of the final model and reach an automatic decision. Several methods can be used for high-level DF, such as weighted decision methods, Bayesian inference, Dempstere Shafer's

theory of evidence, and fuzzy set theory. There is a link between high-level data fusion and ensemble methods.

4. Federated learning. Federated learning enables multiple parties jointly train a machine learning model without exchanging the local data. In case of federated learning, we can talk about model fusion.

Moreover, we will discuss the outcome economy model, to show the possibilities where data fusion could be beneficial in a business setting.

Knowledge and understanding: The student can explain fusion on the different levels: low level, mid level and high level as well as federated learning. They can identify which approach is appropriate for a problem in hand.

Applying knowledge and understanding: Students are able to describe the advantages and disadvantages of different methods. Students have obtained the knowledge to develop, program, analyse, and apply fusion methods to a wide variety of problems in the context of data-driven decision making.

Making judgements: Students will be able to judge the quality of models, results and approaches (e.g., scientific publications).

Communication: Students will be able to present the results the fusion models to specialists or non-specialists.

Learning skills: Students will be able to familiarize themselves with fusion techniques beyond the scope of the course in order to solve a problem.

Study material: Course notes and other information made available.

Exam: Written exam (70%) + assignment (30%)

ECTS: 6

Planning and Scheduling (Code KEN4253)

Examiner: Dr. Steven Kelk

Desired prior knowledge: Data Structures & Algorithms. Discrete Mathematics. Graph Theory

Prerequisites: None

Description: In many real-world processes, particularly in industrial processes and logistics, decisions need to be taken about the time of the completion of (sub)tasks, and the decision about what production machines complete the tasks. There are often constraints on the order in which tasks, or 'jobs' can be performed, and there are usually capacity constraints of the machines. This leads to natural, industrially critical optimization problems. For example, a company might choose to buy many machines to process jobs, but then there is a risk that the machines will be underused, which is economically inefficient. On the other hand, too few machines, or an inappropriate ordering of tasks, may lead to machines spending a significant amount of time standing idle, waiting for the output of other machines, which are overcrowded with tasks. In this course, we look at various mathematical models and techniques for optimizing planning and scheduling problems, subject to different optimality criteria. We will discuss, among others, single-machine models, parallel-machine models, job-shop models, and algorithms for planning and scheduling (exact, approximate, heuristic) and we also touch upon the computational complexity (distinguishing between 'easy' and 'difficult' problems) of the underlying problems. Last but not least, we will also introduce integer linear programming as a uniform and generic tool to model and solve planning and scheduling problems.

Knowledge and understanding: Students will possess the mathematical and algorithmic tools to model and solve planning/scheduling problems. Students will be able to recognize real-world problems in the unified theory and established language of planning and scheduling.

Applying knowledge and understanding: Students will be able to apply the new techniques to various problems arising in real-world applications. Students will be able to deploy the standard algorithmic techniques, and be able to design new algorithmic solutions, and to argue about their performance properties.

Making judgements: Students will understand under which circumstances different planning/scheduling problems are computationally tractable, and will judge algorithmic technique can be used to exactly or approximately solve these problems.

Communication: Students will be able to analytically argue about correctness of the used algorithmic approaches. Students will be able to explain modelling approaches to planning and scheduling problems in the language of the theory of planning and scheduling.

Learning skills: Students will enhance their study skills such as time management, effective reading, critical thinking and reading, exact and unambiguous writing and formulating of ideas and statements, and reflection on marked work. Along the way, students will improve general learning skills such as self-motivation, careful listening and giving instructions, and openness to new knowledge. Students will also be exposed to autonomous self-study.

Study material: Appropriate study material will be provided throughout the course.

Recommended literature: None

Exam: Written exam (75%) at the end of the course, and graded exercises (25%) throughout the course.

ECTS: 6

Explainable Artificial Intelligence (KEN4246)

Examiner: DProf. Dr. Nava Tintarev and Dr. Tjitze Rienstra

Prerequisites: Data Mining or ACML

Desired Prior Knowledge: Data Analysis

Description: A key component of an artificially intelligent system is the ability to explain to a human agent the decisions, recommendations, predictions, or actions made by it and the process through which they are made. Such explainable artificial intelligence (XAI) can be required in a wide range of applications. For example, a regulator of waterways may use a decision support system to decide which boats to check for legal infringements, a concerned citizen might use a system to find reliable information about a new disease, or an employer might use an artificial advice-giver to choose between potential candidates fairly. For explanations from intelligent systems to be useful, they need to be able to justify the advice they give in a human-understandable way. This creates a necessity for techniques for automatic generation of satisfactory explanations that are intelligible for users interacting with the system. This interpretation goes beyond a literal explanation. Further, understanding is rarely an end-goal in itself. Pragmatically, it is more useful to operationalize the effectiveness of explanations in terms of a specific notion of usefulness or explanatory goals such as improved decision support or user trust. One aspect of intelligibility of an explainable system (often cited for domains such as health) is the ability

for users to accurately identify, or correct, an error made by the system. In that case it may be preferable to generate explanations that induce appropriate levels of reliance (in contrast to over- or under-reliance), supporting the user in discarding advice when the system is incorrect, but also accepting correct advice.

The following subjects will be discussed:
1. Intrinsically interpretable models, e.g., decision trees, decision rules, linear regression.
2. Identification of violations of assumptions; such as distribution of features, feature interaction, non-linear relationships between features; and what to do about them.
3. Model agnostic explanations, e.g., LIME, scoped Rules (Anchors), SHAP (and Shapley values)
4. Ethics for explanations, e.g., fairness and bias in data, models, and outputs.
5. Computational Argumentation
6. (Adaptive) User Interfaces for explainable AI
7. Evaluation of explanation understandability

Knowledge and understanding: Students can explain the difference between different explanation approaches (e.g., global versus local models) and can identify which are suitable to use based on underlying assumptions and relative advantages and limitations.

Applying knowledge and understanding: Students can critically choose and apply XAI methods. Students can formulate evaluation protocols to validate the understandability of explanations, demonstrating awareness of the ethical, normative, and social consequences of their applications.

Making judgements: Students will be able to critically evaluate the quality (rigor of methodology), and ethical consequences, of approaches (systems or scientific publications) based on the XAI techniques taught.

Communication: Students will be able to communicate their ideas effectively in written form. They will be able to actively contribute to group-wise communication, and in both oral and written form present their models and outputs to specialists.

Learning skills: Students will be able to familiarize themselves, and critically assess XAI techniques beyond the scope of the course in order to solve a problem.

Study material: Course notes, required reading of scientific articles.

Recommended Literature: Molnar, Christoph. Interpretable Machine Learning. Lulu. com, 2020. Rothman, Denis. Hands-On Explainable AI (XAI) with Python: Interpret, visualize, explain, and integrate reliable AI for fair, secure, and trustworthy AI apps, Packt, 2020

Exam: Active reading (20%) + one individual written assignment (50%) + group project (30%)

ECTS: 6

Autonomous Robotics Systems (KEN4114)

Examiner: Dr. Rico Möckel.

Desired Prior Knowledge: Discrete Mathematics, Linear Algebra, Probabilities and Statistics, Data Structures and Algorithms, Machine Learning, Search Techniques.

Prerequisites: None

Description: Operating autonomously in unknown and dynamically changing environments is a core challenge that all robotic systems must solve to work successfully in industrial, public, and private areas. Currently popular systems that must demonstrate such capabilities include self-driving cars, autonomously operating drones, and personal robotic assistants. In this course, students obtain deep knowledge in creating autonomous robotic systems that can operate in and manipulate unknown and dynamically changing environments by autonomously planning, analysing, mapping, and modelling of such environments. Students learn to approach these challenging tasks through three main techniques: swarm intelligence, model-based probabilistic frameworks, and (mostly) model-free techniques from artificial evolution and machine learning.

Knowledge and understanding: Students gain a deep understanding of the challenges in autonomous robotic systems and how these challenges are addressed in state-of-the-art systems. Students learn about and practice techniques for autonomous mapping, localization, navigation, sensing, modelling robot motion, planning, and decision-making. Through the course, students obtain in-depth knowledge and hands-on experience in a variety of algorithms and techniques from machine learning, agent technology, and search techniques including Bayesian filters (like Kalman Filters, Extended Kalman Filters, Histogram Filters, and Particle Filters), artificial neural networks, evolutionary algorithms, and swarm intelligence.

Applying knowledge and understanding: After successful completion of the course, students will have obtained in-depth knowledge to understand, adapt, apply, and autonomous robotics systems. Students obtain the ability to select from a variety of available tools feasible solutions for the complex and rather ill defined problem domains of autonomous robotic systems and to predict the resulting consequences of their choices. Furthermore, students learn how to choose, apply, formulate, and validate models of autonomous robotic systems and of appropriate control techniques from artificial intelligence for these systems.

Making judgements: Students will be able to comprehend and to critically judge scientific publications on autonomous systems, artificial evolution, and swarm intelligence. From this literature, students are able to search for and to critically process information to solve given ill-defined but in practice highly relevant problems in autonomous systems. Students are able to critically discuss social, economic, and ethical consequences of artificial intelligence and autonomous decision-making.

Communication: Students learn to critically discuss challenges and professional solutions in autonomous robotic applications with both experts and non-experts.

Learning Skills: The course prepares students to work on robotic applications in professional research and business environments. Students will be able to autonomously acquire new skills and knowledge to develop, program, analyse, and apply advanced techniques to a wide variety of problems.

Study material: Thrun et al. (2005), Probabilistic Robotics, The MIT press, ISBN-13: 978-0262201629.

Lecture material and publications: provided during the lecture.

Recommended literature:
- Floreano and Nolfi (2000), Evolutionary Robotics, The MIT press. ISBN-13: 978-0262640565.
- Dario Floreano und Claudio Mattiussi (2008), Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies, ISBN-13: 978-0262062718

Examination: The final course grade is 80% of the final written “closed-book” exam grade plus 20% of the practical group assignments grades

ECTS: 6

Algorithms for Big Data (KEN4254)

Examiner: Dr. Matus Mihalák

Desired prior knowledge: Discrete mathematics, algorithm design and analysis, elementary discrete probability

Prerequisites: None

Description: The emergence of very large datasets poses new challenges for the algorithm designer. For example, the data may not fit into the main memory anymore, and caching from a hard-drive becomes a new bottleneck that needs to be addressed. Similarly, algorithms with larger than linear running time take simply too long on very large datasets. Moreover, simple sensory devices can observe large amount of data over time, but cannot store all the observed information due to insufficient storage, and an immediate decision of what to store and compute needs to be made. Classical algorithmic techniques do not address these challenges, and a new algorithmic toolkit needs to be developed. In this course, we will look at a number of algorithmic responses to these problems, such as: algorithms with (sub-)linear running times, algorithms where the data arrive as a stream, computational models where memory is organized hierarchically (with larger storage units, such as hard-drives, being slower to access than smaller, faster storage such as CPU cache memory). New programming paradigms and models such as MapReduce/Hadoop will be discussed. We will also look at a number of topics from classical algorithm design that have undiminished relevance in the era of big data such as approximation algorithms and multivariate algorithmic analysis.

Knowledge and understanding: Students will know, exemplified on selected topics, what can be provably achieved when designing and analysing algorithms for very large datasets, and will know some of the most successful state-of-the-art algorithmic techniques for dealing with algorithmic challenges posed by large data sets.

Applying knowledge and understanding: Students will be able to adjust and apply the gained knowledge about algorithmic techniques to various algorithmic challenges of handling large datasets.

Making judgements: Students will be able to categorize large-scale problems according to their computational feasibility, and select the appropriate algorithmic response.

Communication: Students will be able to reason about computational problems and algorithms addressing the problems in a clear, exact, and unambiguous way.

Learning skills: Additionally to the guiding material provided by the lecture, the students will autonomously search, read, and study the details from various sources.

Study material: Will be provided throughout the lecture.

Recommended literature: None.

Exam: Written exam (80%) at the end of the course and graded exercises (20%) throughout the course.

ECTS: 6

Computer Vision (KEN4255)

Examiner: Dr. Mirela Popa

Desired prior knowledge: Basic knowledge of Matlab, linear algebra and machine learning. This course offers the basics on image processing although prior knowledge is also a plus.

Description: Can we make machines look, understand and interpret the world around them? Can we make cars that can autonomously navigate in the world, robots that can recognize and grasp objects and, ultimately, recognize humans and communicate with them? How do search engines index and retrieve billions of images? This course will provide the knowledge and skills that are fundamental to core vision tasks of one of the fastest growing fields in academia and industry: visual computing. Topics include introduction to fundamental problems of computer vision, mathematical models and computational methodologies for their solution, implementation of real-life applications and experimentation with various techniques in the field of scene analysis and understanding. In particular, after a recap of basic image analysis tools (enhancement, restoration, color spaces, edge detection), students will learn about feature detectors and trackers, fitting, image geometric transformation and mosaicing techniques, texture analysis and classification using unsupervised techniques, object classification and face recognition, camera models, epipolar geometry and 3D reconstruction from 2D views.

Knowledge and understanding: Students will be able to apply the most suitable techniques for image pre-processing (e.g. enhancement, restoration), feature extraction, texture analysis, perspective geometry, camera models and topics on object recognition. Students will be able to identify the most suitable techniques in a series of visual computing problems

Applying knowledge and understanding: The students will be able to choose and/or construct solutions in a variety of professional/vocational contexts requiring image processing and computer vision (robotics, manufacturing, AI, web applications, surveillance). They will be able to build and assess methodologies for handling real-world complex problems in computer vision, making use of pre-existing data for training their models.

Making judgements: The students will be able to choose and combine the right methods to tackle real-world computer vision problems, captured in real-life settings and having no obvious solutions. They will be able to propose and build techniques combining computer vision methods along with machine learning instruments for scene understanding and object recognition.

Communication: Through small research projects, students will be able to communicate their findings and explain the rationale behind their choices in computer vision techniques for image/video analysis.

Learning skills: After successful completion of the course, students will be able to analyse images and videos and retrieve or process content in order to derive useful information, applicable in a variety of domains (e.g. satellite imagery, surveillance, robotics, medical imaging)

Study material:
- Lecture slides and provided notes
- Computer vision: algorithms and applications". Szeliski, Richard. Springer Science & Business Media, 2010 (available online)
- Computer Vision: A Modern Approach, 2nd Edition. David A. Forsyth.

Recommended literature: "Digital Image Processing", Rafael C. Gonzalez & Richard E. Woods, Addison-Wesley, "Computer Vision: Models, Learning and Inference", Simon J.D. Prince 2012.

Exam: Projects (33% individual assignments – 17% group-based assignments) and final exam (50%).

ECTS: 6

Information Retrieval and Text Mining (KEN4153)

Examiner: Prof dr. ir. Jan Scholtes

Tutor(s): None.

Desired Prior Knowledge: None.

Prerequisites: None.

Description: Using today's search engines allows us to find the needle in the haystack much easier than before. But how do you find out what the needle looks like and where the haystack is? That is exactly the problem we will discuss in this course. An important difference with standard information retrieval (search) techniques is that they require a user to know what he or she is looking for, while text mining attempts to discover information that is not known beforehand. This is very relevant, for example, in criminal investigations, legal discovery, (business) intelligence, sentiment- & emotion mining or clinical research. Text mining refers generally to the process of extracting interesting and non-trivial information and knowledge from unstructured text. Text mining encompasses several computer science disciplines with a strong orientation towards artificial intelligence in general, including but not limited to information retrieval (building a search engine), statistical pattern recognition, natural language processing, information extraction and different methods of machine learning (including deep learning), clustering and ultimately integrating it all using advanced data visualization and chatbots to make the search experience easier and better.

In this course we will also discuss ethical aspect of using Artificial Intelligence for the above tasks, including the need for eXplainable AI (XAI), training deep-learning large language-models more energy efficient, and several ethical problems that may arise related to bias, legal, regulatory and privacy challenges.

This course is closely related with the course Advanced Natural Language Processing (ANLP). In the ANLP course, the focus is more on advanced methods and architectures to deal with complex natural language tasks such as machine translation, and Q&A systems. IRTM focusses more on building search engines and using text-analytics to improve the search experience. In the IRTM course, we will use a number of the architectures that are discussed in more detail in ANLP. The overlap between the two courses is kept to a minimum. There is no need to follow the courses in a specific order.

Knowledge and understanding: The student will be able to select, understand and apply different phases and methods used to create successful Information Retrieval and Text Mining applications. In addition, the student learns to evaluate the quality of such methods according to best-practice standards as used in the field.

Applying knowledge and understanding: Students will be able to recognize applications of text mining and information retrieval in different domains such as legal services, medical research, regulatory oversight, compliance, humanities, and customer services. After the course, the student can formulate an opinion or course of action when dealing with text-based KE-problems based on incomplete, limited and in part unreliable information. After the course, students can apply their knowledge and understanding in a manner that shows a scientific approach to their work or vocation. They are able to handle complex and ill-defined text-based problems for which it is not a priori known if there is an appropriate solution, they know how to acquire the necessary information to solve the sub-problems involved, and they know how to proceed with problems for which there is no standard or reliable route to the solution.

Making judgements: Upon completion of the course, students are able to recommend the most appropriate methods from the fields of text mining and information retrieval when confronted with KE-problems involving textual and other forms of unstructured data.

Communication: Students are able to communicate the (dis)advantages of several methods from the field of text mining and information retrieval to both an audience of non-experts.

Learning skills: After the course, the student has developed those learning skills that are necessary for a successful further career in text mining or information retrieval at the highest professional level. The student will be able to continue to develop their text-mining and information retrieval skills. The student is able to detect missing knowledge and abilities and to deal with them appropriately by finding and consulting resources that can help them to fill the gaps and new developments.

Study material: A syllabus and copies of the course slides will be used.

Recommended literature: Introduction to Information Retrieval. Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. Cambridge University Press, 2008. In bookstore and online: http://informationretrieval.org and Feldman, R., and Sanger, J. (2006). The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data. Cambridge University Press

Exam: The result of the practical exercises contributes 30% to the final examination of the course. The other 70% is determined by the theoretical exam. The theoretical exam is open book. For the practical exercise, students can select a research topic and a text corpus from the provided list (or another relevant open source collection) and implement a number of relevant text-mining operations by using open source text-mining tools. A number of relevant pre-processing operations, text mining operations, and visualizations have to be implemented. Proposals of work will have to be within one week after the start of the course, after which they will be reviewed. After approval, the students can start the implementation of their proposals. At the end of the course, each student or group shall write a report on the research and the results and the results shall be presented to the rest of the class.

ECTS: 6

Symbolic Computation and Control (KEN4252)

Examiner: Prof dr. ir. Ralf Peeters.

Desired Prior Knowledge: Linear Algebra, Calculus, Mathematical Modelling.

Course description: This course consists of two interrelated parts. In the first part, we focus on basic techniques for the digital control of linear dynamical systems using feedback. We start by addressing system stability and we discuss the technique of pole placement by state feedback to solve the regulation problem. Then we introduce state observers to solve the regulation problem by output feedback. Next, we extend our scope to tracking problems. This involves the design of additional dynamics to characterize the relevant class of reference signals, which are then integrated with the earlier set-up for output feedback. Finally, we discuss the classical topic of optimal control, which can be employed to avoid using prototype systems for pole placement, and which allows the user to design a feedback law by trading off the cost involved in generating large inputs against the achieved tracking accuracy. In the second part, we address computational issues, related to the field of systems and control. Classically, computers have been designed primarily to perform approximate numerical arithmetic. Modern software packages for mathematical computation, such as Maple and Mathematica, allow one to perform exact and symbolic computation too. We shall explore this new area. It is demonstrated how speed, efficiency and memory usage considerations often lead to surprising and fundamentally different algorithmic solutions in a symbolic or exact context. Applications and examples involve stability of linear systems, model approximation, and linear matrix equations with free parameters. Practical classes serve to demonstrate the techniques and to make the student familiar with exact and symbolic computation.

Knowledge and understanding: Students familiarize themselves with state and output feedback to achieve control of dynamical systems. Concretely, they learn to (mathematically) build a basic stabilizing feedback controller for a linear input-output dynamical system, using a combination of different design techniques. Students learn methods for exact numerical and symbolic computation, as used in algebraic computation with unspecified parameters. They also learn in which ways these are different from the more commonly used approximate numerical (floating-point) methods: in terms of accuracy, speed (complexity), and memory usage.

Applying knowledge and understanding: Students will be able to construct and implement, for a given linear dynamical input-output system: (a) stabilizing state feedback, (b) full state observer, and (c) additional dynamics to perform tracking of a specified output trajectory. They will also be able to assess the quality of a controller, regarding an optimal control LQ criterion, and in view of the desired settling time and the trajectory approximation. Students will be able to determine the stability of a given linear dynamical system in an exact and/or symbolic algebraic way. They will also be able to efficiently solve linear systems of (matrix) equations involving symbolic parameters, avoiding pitfalls, which arise from techniques from approximate numerical computation.

Making Judgements: Students will be able to judge the quality of a feedback design for stabilization (regulation) or tracking. Students will be able to indicate which exact and symbolic computation methods will and will not be useful for a given parameterized problem, regarding speed and memory usage.

Communication: Students will be able to motivate the design of a feedback controller, the construction of a trajectory approximation, the design of a full state observer, and the implementation choices of the weights in LQ-design. They will be able to explain the concept of feedback in the area of control. Students can adequately discuss speed and efficiency properties of an algorithm (approximate numerical, exact numerical, symbolic) to specialists and non-specialists.

Learning Skills: Students will be able to read and interpret basic scientific literature on control theory and on numerical and symbolic computation. They can use Matlab and the Control Toolbox and work out ideas computationally. Students can use some of the exact and symbolic functionality of Mathematica and work out ideas computationally.

Study material: Syllabus, provided on the study portal. Handouts.

Recommended Literature: Richard J. Vaccaro, Digital Control - A State-Space Approach, McGraw-Hill International Editions, 1995. ISBN 0-07-066781-0.

Exam: Written exam by computer in two parts, each having a weight of 50% on the final grade: one midterm take-home exam with Matlab on part 1 (control), one final classroom exam with Mathematica on part 2 (symbolic computation). The resit exam is on both parts of the course in a classroom setting.

ECTS: 6

Introduction to Quantum Computing for AI and Data Science (KEN4155)

Examiners: Dr. Menica Dibenedetto and Dr. Georgios Stamoulis

Desired prior knowledge: probability theory, linear algebra, design and analysis of algorithms

Prerequisites: None

Description: In this course we lay down the foundations and basic concepts of quantum computing. We will use the mathematical formalism borrowed from quantum mechanics to describe quantum systems and their interactions. We introduce the concept of a quantum bit and discuss different physical realizations of it. We then introduce the basic building blocks of quantum computing: quantum measurements and quantum circuits, single and multi-qubit gates, the difference between correlated (entangled) and uncorrelated states and their representation, quantum communication, and basic quantum protocols and quantum algorithms. Finally, we discuss the different types of noise involved in real quantum computers (coherent and incoherent errors, state preparation, projection and measurement) and their effect on performance, and outline current efforts for mitigating the issues.

Knowledge and understanding: Students will learn the fundamental principles and concepts behind quantum computing, protocols, and algorithms. Students will understand the differences between classical and quantum computation, and where the (theoretical) computational power of quantum machines comes from. Students will also get to understand the current challenges in building and using quantum computers.

Applying knowledge and understanding: Students will be able to apply existing quantum algorithms as black-box to various simple computational problem. Students will further be able to analyse simple quantum algorithms for different computational problems.

Making Judgements: Students will be able to judge how the potential computational power of quantum computing can be leveraged, and how it can be applied to other fields in a beneficial way.

Communication: Students will be able to discuss quantum computation critically and judge not only its benefits but, equally important, its shortcomings. Students will especially be able to communicate potential benefits of quantum computation to the fields of artificial intelligence and data science.

Learning skills: Students will practice learning entirely new computational concepts, and how to relate existing concepts (classical computation) to new concepts (quantum computation). Students will learn to critically reflect on both the scientific literature and the societal expectations. Students will learn to self-study from state-of-the-art research articles, when classical text-books are not available.

Study material: To be announced.

Exam: Written exam (100%)

ECTS: 6

Reinforcement Learning (Code: KEN4157)

Examiner: Dr. ir. Kurt Driessens

Desired Prior Knowledge: Machine Learning

Prerequisites: none

Description: Reinforcement learning is a type of machine learning problem in which the learner gets a (delayed) numerical feedback signal about its demonstrated performance. It is the toughest type of machine learning problem to solve, but also the one that best encompasses the idea of artificial intelligence as a whole. In this course we will define the components that make up a reinforcement learning problem and will see what the important concepts are when trying to solve such a problem, such as state and action values, policies and performance feedback. We will look at the different properties a reinforcement learning problem can have and what the consequences of these properties are with respect to solvability. We will discuss value based techniques as well as direct policy learning and learn how to implement these techniques. We will study the influence of generalisation on learning performance and see how supervised learning (and specifically deep learning) can be used to help reinforcement learning techniques tackle larger problems. We will also look at the evaluation of learned policies and the development of performance over time.
Knowledge and understanding: Students will be able to explain the setup of a reinforcement learning problem and list its formal components, explain the difficulties faced when adding function approximation to reinforcement learning, explain the origins of the learning signal for policy gradient methods for reinforcement learning.

Applying knowledge and understanding: Students can implement and apply online and offline tabular techniques of value based reinforcement learning algorithms, apply the use of function approximation in value based reinforcement learning algorithms, implement and apply policy gradient methods for discrete and continuous action tasks and deep learning methods to reinforcement problems

Making judgements: Students will be able to judge the suitability of reinforcement learning techniques as a solution for an AI problem, choose/select between exploration and exploitation tradeoff methods suited to the problem faced, interpret and judge the results of a reinforcement learning agent

Communication: Students will gain a working knowledge of reinforcement learning as a problem, and of the state of the art in solution techniques and will be able to motivate his/her choices concerning the application of these techniques.

Learning Skills: Students will learn that the state of the art in reinforcement learning continues to develop at a rapid pace and that becoming and staying an expert in the domain will require continued learning.

Study material: Course slides to support the lectures; supplementary material consisting of research papers and book chapters.

Recommended literature: Reinforcement Learning: An Introduction by R. Sutton and A. Barton

Exam: Assignment based.

ECTS: 6

Period 1.6

Research Project AI 2 & DSDM 2 (KEN4131 & KEN4231) .

Examiner: T.b.a.

Coordinator: Dr. Linda Rieswijk

Desired Prior Knowledge: None.

Prerequisites: None.

Description: The research project takes place during the three periods of the semester. Project topics are presented at the start of the semester and assigned to students based on their preferences and availability. The emphasis in the first phase is on initial study of the context set out for the project and the development of a project plan. In the second period, the goal is to start modelling, prototyping and developing. In period 3, the implementation, model and/or experiments set out in the project plan has to be finished and reported on. At the end of period 1 and 2, a progress presentation takes place. The project results in a project presentation, a project report and possibly a public website and/or product.

Knowledge and understanding: Students get to know and possibly contribute to state of the art methods within the fields of Artificial Intelligence and/or Data Science for Decision Making to answer an open question.

Applying knowledge and understanding: Student write their own research plan in coordination with a staff member (plus possibly outsiders) who act as clients with an open question. Students with different backgrounds and from both masters work together in teams to build and evaluate an answer to an open question. Students find, judge the suitability, apply, and evaluate state of the art techniques to answer questions and construct applications in the field of Artificial Intelligence and Data Science. Students apply the accumulated knowledge from other educational activities in application specific areas

Making judgements: Students judge feasibility of tasks, attainability of goals, and the amount of work involved. Students think about the possible consequences of their work. Students evaluate state of the art and the applicability and scope of research results.

Communication: Students will learn to:
1. orally communicate and cooperate with peers
2. orally report on progress and intermediate results to superiors
3. orally negotiate and communicate with clients
4. communicate their ideas in written form, both for an academic and a general audience
5. give effective presentations

Learning Skills: Students increase their own level of knowledge in a specialised sub-discipline of the field of Artificial Intelligence and/or Data Science. Students perform research into recent state of the art techniques. Students learn that the field of Artificial Intelligence and Data Science are constantly evolving beyond what is taught in class

Study material: Slides provided at the end of joint information sessions. Literature provided by the project supervisors.

Recommended literature: Justin Zobel (2004), Writing for Computer Science, Springer, ISBN:1852338024

Exam:
Phase 1: project plan + presentation (15%);
Phase 2: layman's website + presentation (15%);
Phase 3: Project report + presentation (70%)

ECTS: 6

2.2 Year 2 of the Master Programme's AI and DSDM

Period 1, 2 and 3 of year two of the master's program consist of electives to be chosen by the student. This optional program can be assembled at your own choice from the options provided, but within academic significance, level and relevance to your master's track. The choice of electives is subject to approval by the Board of Examiners. The electives consist of the following options to choose from: master courses to be followed at the Department of Advanced Computing Sciences, at other UM Master programmes, at another research university, a research project, an internship, a semester abroad at a foreign university. Note that you must have obtained at least 40 ECTS of course year 1 in order to enter the second year of the programme.

Electives at Maastricht University outside the Department of Advanced Computing Sciences

It is possible to take electives at other relevant master's programmes at Maastricht University for at most 13 ECTS in the second year of the programme. The following courses below will be automatically approved by the Board of Examiners of the master's programmes AI and DSDM. You should apply through the Special Course Approval procedure via the My UM Portal. Note that they may have limited capacity.

Period School of Business and Economics	
Social Choice Theory (ECB4005)	6.5 ECTS
Supply Chain Operations (EBC4016)	6.5 ECTS
Negotiation and Allocations (EBC4193)	6.5 ECTS
Intellectual Property Rights in a Digital Economy (EBC4026)	6.5 ECTS
High-Dimensional Econometric Methods for Big Data (EBC4218)	6.5 ECTS
Faculty of Psychology and Neuroscience	

Besides complying that you have passed 40 ECTS, for taking these electives at FPN you should have passed "Advanced Concepts in Machine Learning" and "Autonomous Robotic Systems" at the Department of Advanced Computing Sciences.

Auditory and Higher Order Language Processing (PSY4051)	4 ECTS
Perception and Attention (PSY4052)	4 ECTS
Sensorimotor Processing (PSY4055)	4 ECTS

Exam: Depends on content of the elective program.

ECTS: 30

Internships:
Another option for the elective semester in the Master Programme is to conduct a Business or Research internship. The students can choose the company or research organisation themselves. Together with a supervisor from the Department of Advanced Computing Sciences and a representative of the host organisation, the student fills out an internship proposal (which can be found here on Canvas) and this requires approval of the Board of Examiners prior to its start. For this reason, it is important to start this process early. The university uses a standard internship agreement that students must use. For more information about doing an internship, look at the FAQ sheet here.

Period 2.4, 2.5, 2.6

3.3 Master's thesis AI & DSDM (KEN4160 & KEN4260)

The Master's Artificial Intelligence and Data Science for Decision Making will be completed by writing a master's thesis. The thesis is produced individually and is the result of a master's research project that runs during the second semester of year 2 of the master's programme. In the preliminary phase, the emphasis is on self-study, subject determination, planning and some preliminary research. Then the actual research is started. The final phase is used to finalize the master's thesis. The master's project is completed by a public on-site presentation and discussion of the results (also known as a defence of the thesis). The master's thesis is supervised by one of the senior researchers of the Department of Advanced Computing Sciences. In principle, there should be no confidentiality agreements for a thesis, and staff members cannot be expected to commit to these.

Exam: Master's thesis and presentation.

ECTS: 30

** Note that when you enrol in February, you follow your electives in period 2.4, 2.5 and 2.6 and work on your master's thesis in period 2.1, 2.2, 2.3.*

Master's thesis Artificial Intelligence and Data Science for Decision Making

Master thesis is an individual work. The research is done by the guidance of the supervisor(s), the thesis is written individually, taking a feedback of the supervisor into account, and the thesis is concluded by a thesis defense. If relevant, the thesis is handed in together with the source code, data, or any other supporting material. In order to start working on the thesis, a student needs to have obtained at least 70 ECTS (among which are 40 credits of the first year).

General procedure
The process of writing a master's thesis consists of 6 phases. With the exception of the first phase, it is scheduled in the last semester of the master's study. The time frame given below is an indication for these phases.

Phase 1: Topic selection
At the end of the first semester of an academic year, the students are informed of the main directions of research in the research areas at the Department of Advanced Computing Sciences. Next to this, the master's programmes maintains a website of concrete thesis topics, which is updated annually. Based on this information, students acquire more information about specific possibilities in the areas by means of individual discussions with relevant researchers available. These discussions take place upon the initiative of the student.

Phase 2: Thesis Research Plan
Before the actual work on the thesis, each student must have chosen a thesis topic and a principal thesis supervisor. The student creates a thesis research plan, which is to be signed by the student and the two prospective thesis examiners, and then handed over to the master's thesis coordinator. The plan is sent to the Board of Examiners for approval.

The students will be invited to present their thesis topic, and the related work in a Master-thesis seminar, in the presence of fellow students and thesis supervisors.

Phase 3: Research
After the thesis plan has been approved, the student carries out his/her own research. This research process will be guided by the thesis supervisor through a series of regular appointments, preferably on a weekly basis.

The students will be invited to present the first research achievements, with audience staff members of the Department of Advanced Computing Sciences and fellow students.

Phase 4: Writing
At the end of the main research phase, the focus is on writing of the thesis. The student can expect at least one feedback on the text from the supervisor, before handing-in the final version of the thesis. The two examiners will evaluate the final version of the thesis shortly before the thesis defence.

Phase 5: Preparation for presentation
In the last week, , the student prepares a final presentation of the thesis research. This individual presentation will have a maximum length of 30 minutes, followed by 15 minutes of discussion between the student and the two examiners.

Phase 6: Presentation
The master's thesis defense takes place upon agreement of the supervision team. The defence is public and takes place on the premises of the department.

Requirements and assessment
For the master's thesis research, every student has to conduct a short scientific research. This can be an empirical or a theoretical research. The topic is open, as long as it fits into the field of the master's programme. Staff of the Department of Advanced Computing Sciences will briefly introduce their main areas of research, but students are encouraged to propose a research topic themselves. The topic and the research question have first to be approved by the prospective examiner. This plan will be signed by the student and the prospective examiners and then handed in to the Board of Examiners for the formal approval. It is possible to execute the master's thesis research as an external training period. This should be well defined in the master's research plan. In this case, the plan should also include the name of the company, the name of the external supervisor, the size of the project and any agreements about payment and confidentiality. The plan should also be signed by the external supervisor.

The research needs to be original in such a way that the thesis supervisor is convinced that this research has not been done before. The research also needs enough depth and still it must be possible to finish it in the set amount of time. Every thesis is an individual work.

The thesis is graded by the examiners using a standard assessment form, available on Canvas. The weighing of these aspects is up to the examiners.

Content aspects
The thesis describes the problem statement, research questions, approach and results of the research. This has to be done in a clear, structured and scientific manner. This includes:
- a clear introduction in which the problem statement and research questions are presented;
- the master's student shows proper analysis of complex issues in a new context and is able to formulate a proper problem statement;
- a clear conclusion, based solely on the already used thought out principles and derived results;
- a clear line is shown between problem statements, approach, methods and the derived results;
- a motivation of the followed approach, reflecting on standard methods and their presuppositions,
- an adequate description of the followed approach;

- a purposeful and systematic way of collecting data;
- an honest, clear and concise description of the derived results, if necessary using tables;
- an analysis and discussion of the results;
- the usage of relevant and recent literature for the reasoning in the thesis.
- the correct usage of references.

Design aspects
Correct scientific references have to be used. Images and tables are accompanied by an index and caption. Mathematical formula, definitions, etc. have to be properly designed and numbered. The start and end of mathematical formulae have to be properly defined.

Language aspects
The thesis has to be written in English, considering correct spelling, syntactical structure of sentences and structure of content in paragraphs. The target audience consists of fellow master's students and lecturers. Any jargon and/or abbreviations have to be explained unless they are common knowledge for this audience.

Citations
It is allowed to use several short citations. These citations have to be clearly referenced and have to be typographically distinguishable (that is, citations are placed in quotes). Non-allowed citations or missing references will result in a non-pass.

4 Facilities for Students

In this chapter, you will get an overview of the facilities that Maastricht University offers its students.

4.1 Student Affairs Office

The Student Affairs Office, among other things, takes care of the organization and administration of the education.

Visiting address: Paul-Henri Spaaklaan 1, 6229 GT Maastricht
Postal address: P.O. Box 616, 6200 MD Maastricht, the Netherlands.

Office hours:
PHS, C.1006 daily between 10h00 - 11h00 and 15h00 - 16h00.

Contact:
Admissions: dacs-admissions@maastrichtuniversity.nl
Tel.: +31(0)43 388 26 77

Exam Administration: dacs-examination@maastrichtuniversity.nl
Tel.: +31(0)43 388 35 25

Scheduling: dacs-scheduling@maastrichtuniversity.nl
Tel.: +31(0)43 388 35 25

International Relations (exchange): dacs-iro@maastrichtuniversity.nl

4.2 Administrative structure of the Faculty

The administrative structure of the Faculty is laid down in the faculty regulations.

The dean is responsible for the faculty's administration. More information is to be found on the website: https://www.maastrichtuniversity.nl/nl/over-de-um/faculteiten/faculty-science-and-engineering.

Faculty Board
The Faculty Board, chaired by the dean of the Faculty of Science and Engineering, runs the Faculty. The Faculty Board is charged with the general management and administration, as well as its policy regarding academic research and education.

Faculty Council
The Faculty Council is entitled to submit proposals and present their opinion to the Faculty Board regarding any matters relating to faculty administration, policy, education and research. The Faculty Council has rights of approval, e.g. regarding faculty regulations, research programmes, and the implementation of a binding study advice, and rights of advice, e.g. regarding the budget.

Directors of Studies
The programme directors Dr. Pietro Bonizzi for the Bachelor programme and Dr. Matus Mihalak for the Master programmes are responsible for the organization and coordination of all teaching activities. The Education Programme Committee (EPC) advises the programme directors.

Education Programme Committee
There is one EPC for the Bachelor's Data Science and Artificial Intelligence and the Master's Data Science for Decision Making and Artificial Intelligence. The EPC is responsible for advising the Faculty Board, the Programme Directors and the Board of Examiners. Furthermore, the EPC is entitled to advice in any subject related to the programme, and consists out of eight members, four students and four members of the academic staff. In addition, the quality assurance officer has the role of advisor in the committee.

All correspondence for the Education Programme Committee should be addressed to dacs-secretariat@maastrichtuniversity.nl or by postal mail to:
Department of Advanced Computing Sciences - Maastricht University
P.O. Box 616, 6200 MD Maastricht.

Board of Examiners
The Board of Examiners is in charge of the organization and supervision of the examinations and is appointed by the Faculty Board.

All correspondence for the Board of Examiners should be addressed to
dacs-boe@maastrichtuniversity.nl or by postal mail to:
Department of Advanced Computing Sciences - Maastricht University
Board of Examiners,
P.O. Box 616, 6200 MD
Maastricht

Board of Admissions for the Master's Programmes
The Board of Admissions is responsible for granting the admission requests for entering a master's programme, and is appointed by the Faculty Board. All correspondence for the Board of Admissions should be addressed to dacs-admissions@maastrichtuniversity.nl or by postal mail to:
Department of Advanced Computing Sciences - Maastricht University
Student Affairs Office,
P.O. Box 616, 6200 MD
Maastricht

It is possible to follow incidental courses at the transnational University Limburg (located at Hasselt University, Belgium). Students who want to make use of this possibility should individually ask permission to the Board of Examiners of the master's programmes AI and DSDM, Maastricht University. More information on the transnational University Limburg, its staff members, and information on the content of the courses can be found at: www.uhasselt.be/informatica

4.3 Teaching material

On each project, a project book is published. The project books and the education schedules of each period are available two weeks before the start of a new period, at the latest. What the study material (= obligatory literature) or recommended literature is, is available on the course information page in Canvas, or more specifically Canvas, or in the student handbook.

4.4. Participation in the Education

The students are expected to be available from Monday through Friday from 08.30 to 18.00. for educational activities.

4.5. Announcements on Educational Matters

Announcements concerning educational matters will be published through Canvas. Canvas Students are mainly approached through e-mail and through Canvas. We advise students to check for new announcements/emails daily.

4.6. Change of Address Student

Except for the Canvas, , the Student Affairs Office makes use of mailings to students. . If there is a change in the study address or the address of the student's parents, this should immediately be changed in Studielink. Do not forget to mention the commencing date of the change. During the academic year, the student's study address is considered as their postal address. You may contact dacs-admissions@maastrichtuniversity.nl for help.

4.7. Project Rooms

Scheduled practical lectures have priority over private use of the project rooms by students. The rooms are open to students from Monday through Friday from 08.00 until 18.00. Wireless internet is available throughout the whole building.
For questions about the system management, please refer to the system managers of the Department of Advanced Computing Sciences, tel. +31(0)43-388 54 93 or by mail:
Lo-fse@maastrichtuniversity.nl.

House rules for all project/meeting rooms;
- Users are not allowed to download illegally acquired materials;
- Users are not allowed to illegally download materials
- Users are not allowed to install illegally acquired software;
- Users should use their own devices for saving data, or save your data on your personal network drive (I:);
- Users should handle the furniture with care;
- For the regulation of the air conditioning system, students may contact the Student Support staff.

4.8. Faculty Counsellors for Students

Study Adviser
The study advisers Tessa Fox and Wendy Brandt are staff members whom you can contact if you have any questions concerning your study and can be reached at telephone number +31(0)43-388 35 61, in room C1.020 at PHS, or through dacs-studyadvice@maastrichtuniversity.nl.
They are familiar with the organization of the education, the faculty organization and the study. The student counsellor is a primary advisor for students. If your study comes to a standstill for whatever reason, you can contact the student counsellor. It is also the right person to talk to if you have any questions to which you cannot find any answers in the faculty prospectus or during faculty information meetings. But also in case of personal circumstances due to which your study or personal life are suffering, for instance illness, mental health problems or family circumstances, your student counsellor can listen to you. Conversations are confidential. Based on these talks the student counsellor can direct you to some further assistance. The student counsellor may also call up students for a talk if it appears that their results are falling back. More information and a scheduling tool can be found here.

Internationalization
For any questions you may have about studying for a semester at a foreign university, or about a practical training abroad, for support, and for direct information you can contact during opening hours the international relations officer of the Department of Advanced Computing Sciences Luc Giezenaars via dacs-international@maastrichtuniversity.nl.

4.9 Student Services Centre

The Student Services Centre is responsible for the preparation and execution of the policy of Maastricht University in the area of general student provisions. In short, this department has a number of specialized service units for student-related issues such as accommodation, sports, information on studies and work and career advice. In addition, there is a central information desk in the main entrance hall of the Visitors' Centre, to which current and prospective students may address their questions.
Visiting address: Bonnefantenstraat 2, Tel.: +31(0)43-388 53 88,
www.maastrichtuniversity.nl/ssc.

4.9.1 Visitors' Centre and student registration

Information Desk
The information desk in the UM Visitors' Centre at Bonnefantenstraat 2 is the first point of contact for current and new students. It provides the following services:

- Help with admission and (re)registration;
- Information on and help with visas, scholarships, bank accounts and (health) insurance;
- Changing of address;
- Payment of tuition fees;
- Cancellation of registration;
- Reimbursement of tuition fees;
- Proof of payment/registration;
- Collection of your first UM-card;
- Help with housing;
- Appointments with student deans, psychologists, and career services;
- UM gifts.

Tel.: +31(0)43-388 53 88,
e-mail: study@maastrichtuniversity.nl
FAQ: https://ssc.esc.maastrichtuniversity.nl
Opening hours Monday-Friday 08h30 - 18h00.

Visa and Scholarship Office
The Visa and Scholarship Office is responsible for immigration matters and scholarships for prospective and current students.
For any questions on visas, please visit our website: www.maastrichtuniversity.nl/visa
or e-mail: visa@maastrichtuniversity.nl.

UM Career Services
UM Career Services offers workshops, job interview simulations, Quick career advise and more intensive counseling. For more information, please see www.maastrichtuniversity.nl/careerservices or contact your study adviser at the Department of Advanced Computing Sciences

Student Guidance
Psychological support (Student psychologists)
Student Psychologists may be consulted in case of personal problems. Examples of complaints and problems include:

- Study related problems like study stress and fear of failure;
- Psychological complaints such as anxiety, depression, eating disorders, stress-related complaints, lack of confidence, dealing with traumatic experiences.

The student psychologists can help you by means of individual guidance and/or group training (in Dutch and English). Check the current offer via: https://intranet.maastrichtuniversity.nl/en/dacs-students/advice-support-and-guidance.

For making an appointment use the online tool on the website:
www.maastrichtuniversity.nl/studentguidance

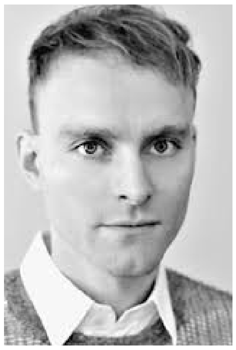
Study related legal support (Student deans).
For more information: www.maastrichtuniversity.nl/studentguidance
e-mail: studentendecanen@maastrichtuniversity.nl
Open visiting hours at the SSC, please check the website for correct timeslots

Studying with a disability, chronic illness or dyslexia
It is important to Maastricht University that students with a functional impairment can successfully complete their studies without too much delay. By functional impairment UM means all disorders that are of a permanent or temporary character. Amongst these are all motor, sensory or psychological disorders, but also non-visible disorders, such as dyslexia, chronic illness, physical complaints, depression and the like. Disability Support is available to students (with a functional impairment), prospective students, student counsellors, teachers, parents and others who are interested.
For more info: https://www.maastrichtuniversity.nl/support/during-your-studies/studying-disability
e-mail: disability@maastrichtuniversity.nl
Open visiting hours: Monday - Thursday from 11h00 to 13h00.
Tel.: +31(0)43-388 52 72.

5 Staff

5.1 Academic Staff


Dr. Francesco Barile


Dr. Olivier Bilenne


Dr. Pietro Bonizzi


Dr. Martijn Boussé


Dr. Cameron Browne

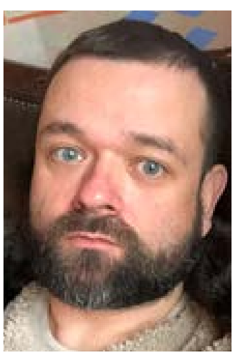

Dr. Daniel Cámpora Pérez


Dr. Rachel Cavill


Dr. Steven Chaplick


Dr. Remzi Celebi


Dr. Pieter Collins


Dr. Walter Crist III


Lucas Dahl


Dr. Otti D'Huys


Dr. Menica Dibenedetto


Dr. Phippe Dreesen


Dr. Kurt Driessens


Prof. dr. Michel Dumontier


Dr. Barbara Franci


Dr. Rishav Hada


Dr. Enrique Hortal Quesada

Contents
Education
Bachelor
Master
Facilities
Staff


Spriha Joshi


Dr. Joël Karel


Dr. Steven Kelk


Dr. Parveen Kumar

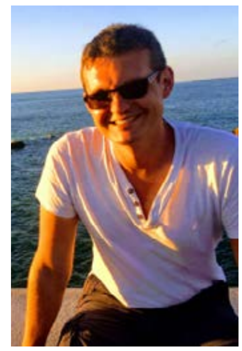

Dr. Stefan Maubach


Dr. Matus Mihalak


Dr. Rico Möckel


Dr. Marieke Musegaas


Prof. Dr. Ir. Ralf Peeters


Dr. Eric Piette


Dr. Mirella Popa


Dr. Tjitze Rienstra


Dr. Linda Rieswijk


Dr. Ir. Nico Roos


Dr. Gijs Schoenmakers


Dr. Katharina Schneider


Dr. Christof Seiler


Dr. Yusuf Can Semerci


Dr. Chiara Sironi


Prof. Dr. Jan Scholtes


Dr. Evgueni Smirnov


Dr. Jerry Spanakis


Dr. Georgios Stamoulis


Dr. Chang Sun


Dr.ir. Marijn ten Thij


Prof. Dr. Frank Thuijsman


Prof. Dr. Nava Tintarev


Dr. Yuquan Wang


Prof. Dr. Gerhard Weiss


Dr. Ronald Westra


Prof. Dr. Anna Wilbik


Prof. Dr. Mark Winands

5.2 Lab assistants


Dean Boonen


Spriha Joshi

5.3 Support Staff


Wendy Brandt


Esther Breuls


Céline Duijsens - Rondagh


Ivanka Dzon


Claire van Doorn


Tessa Fox


André Fraats


Charlotte Hamelers - Geelen

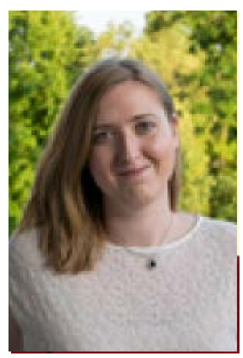

Iris Hoogsteder


Astrid Lamers


Anita Legtenberg


Bas Lemmens


Ellen Narinx - Schrauwen


Desirée Parren


Dénise van Spingelen


Miranda Vermeer


Dieudonnée van de Willige