KEN-BCS 1120: Procedural Programming (2023-2024)

1. Course description

The course (4 ECTS) 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 and recursion. Programming skills will be acquired during practical sessions using the programming language Java.
This course is followed by several courses such as, but not limited to, Objects in Programming, Data Structures and Algorithms, Software Engineering or Projects 1-1 and 1-2, thus is heavily important for your curriculum. Make sure you learn how to program like a master! Programming will follow you throughout your DACS studies (and the rest of your life, even if it's not Java).

2. Teacher information

Lecturers and tutors

Enrique Hortal (he/him)	enrique.hortal@maastrichtuniversity.nl	C4.020
Daniel Cámpora (he/him)	d.camporaperez@maastrichtuniversity.nl	C4.032

Tutor

Tom Bitterman (he/him)	tom.bitterman@maastrichtuniversity.nl

Teaching Assistants

Spriha Joshi (TAs coordinator)

Abhimanyu Anand
Adrien Bersia
Ignacio Cadarso Quevedo
Buse Dağıdır
Annada de Freitas Sousa
Anna-Lena Krause
Vasilis Papadakis
Gabrijel Radovčić
Fivos Tzavellos
More?

How to find us?

We will use Canvas for important announcements and Discord for communicating things about the course. Make sure that you follow the questions asked there and participate in the relevant discussions.

Other than that, we are mostly available online, via our email addresses (see above). We aim to always respond to your emails quickly, typically after 10-30 minutes to 2-3 working days. If you ever wait longer than that without hearing from us, please mail us a reminder. You will probably receive a more helpful answer from us if your email contains at least one clearly-phrased question or request. A reference to the course in the subject of the email (e.g. CS1 – Doubt about Module 3) would be helpful as well.

If you want to discuss something in person, you can ask us after class. You can also -usually- find us in our DACS offices (PHS1 building) but due to our busy schedules, it's always good to check via email (or ask for an appointment).

3. Textbook & Resources

Textbook(s):	Java: A Beginner's Guide, Eighth Edition by H. Schildt (2018). McGraw-Hill Education. Old book (for those who are retaking the course and/or already have it) Big java: late objects by C. S. Horstmann (2013). John Wiley & Sons.
Slides & exercises:	Available online via Canvas
Software:	Visual Studio Code (Check Assignment 0 for instructions)

4. Course format and composition

In this course, we are going to be using a format which includes live coding lectures, individual study at home (including additional video material, book chapters, quizzes, homework, etc.), on-site (practical) tutorials and dedicated sessions (e.g recap and exam preparation lectures). All sessions are split into several groups (2 groups for lectures and live coding sessions and up to 32 for tutorials) except for one session in Week 7 which will be plenary (more information will be delivered during the course). Below we explain the different components.

Live Coding Lectures: During live coding lectures you will be instructed in the basics of new concepts via simple illustrative examples. You are expected to engage with the lecturer and your fellow classmates by asking questions and by participating in the relevant in-class quizzes (held via Wooclap).

Individual study (Video material, Quizzes, Homework): Each week, we will provide some additional material (as videos or other formats) which we expect you to cover individually at home. Our expectation is that you cover this material **before** the relevant live coding lecture, so that you are prepared and able to see how the concepts you studied are applied in practice.
Each live coding lecture is followed by one (or more) small quiz(zes), where you test your knowledge, plus some homework exercises to practice. Ideally, you should have finished homework before coming to the next tutorial, but due to the tight schedule try at least to have it prepared. Homework is not mandatory and is not to be delivered (or graded) but will help you build the foundation needed to attempt the practical sessions. Regarding homework, you can freely ask for help from the lectures, TAs and classmates in the dedicated Discord channel (more later) or during dedicated (Ask a TA) sessions.
Additionally, this year, a set of **optional** assignments will be provided. We will make available 5(+1) assignments covering the different modules of the course. These assignments are intended as personal work but feel free to collaborate with other peers if you consider it helpful.
To hand in assignments for programming courses at DACS we use an online system named CodeGrade. This system automatically compiles, tests and "grade" your code after you upload it. Every module includes a problem/assignment, solving these problems correctly (as determined by CodeGrade) may be a good indicator of your proficiency on the topic with respect to the expected Learning Goals until this point. Because of the way the auto-grading works, you must carefully follow the submission instructions for every assignment, and always check that the written output, method names, file names, parameters, etc. match what was described in the assignment.

Before uploading code to CodeGrade, make sure the program works on your machine. So run it and test it thoroughly. You may upload your submission as often as you like, however, CodeGrade has a limit of 2 submissions every 10 minutes. If your submission fails the autotest, study the error message and try to find a solution by altering and testing your code, before re-submitting.

Although these assignments are not graded, their completion is highly recommended. To maximize the learning experience:

- Do not copy source code from a fellow student or alumni
- Do not copy source code from the internet

Please note that the grade extracted from the CodeGrade autotests are just indicative but not necessarily demonstrate the perfect functioning of the code delivered. Students are responsible for their code and ensuring that it meets all the requirements specified in the assignment description.

Tutorials: These sessions are intended as hands-on sessions where you, together with other colleagues and guided by a TA will work on a programming task. The tasks proposed in the 5 tutorials are part of the development of a basic old-style fun adventure game, "Jerry's Adventure". The (50-minute long) sessions will be structured as follows:

- 5' of silent reading of the assignment
- 5'-10' to "brainstorm" with the TA and start drafting a piece of pseudocode to solve the proposed assignment
- 30' to start translating the pseudocode into actual code, supported by the TA
- 5' to wrap up and establish future lines to improve the implementation of the game

As the time is very limited, it is highly recommended that you read the assignment before the session. Please also notice that the session is not intended to complete the assignment but to establish the basis to continue working on it afterwards (see more in the Game Awards section below).

Office hours: The office hours are intended to solve conceptual doubts. For coding-related issues, you must use other channels, namely Discord and the *Ask a* TA sessions. Each office hours session is dedicated to a specific module and doubts related to it will be prioritized. Therefore, clarifying doubts related to other (previous) modules is not assured.

***Ask a TA* sessions:** Two *Ask a TA* sessions will be organized during the course. In these sessions, you will have the opportunity to ask some TAs for help to solve coding-related issues. Notice that these sessions are STRICTLY dedicated to coding-related issues (assignments, game development, etc.) and any other doubts must be solved via Discord or by the lecturers during dedicated office hours (see above).

Game Awards: The last session of the course is not a lecture one. It will be dedicated to the game. In this session, we will organize an awards ceremony where the best game developer(s) will be granted a small prize. To be invited to participate, you must deliver your improved version of the game after each tutorial session. Delivering each Lab's game before the established deadline will give you "one badge". Collecting at least 4 out of 5 badges (Lab 2 covers two tutorials) will allow you to enter the final contest for "the best game" which will be held in the last week's session. Feel free to work in pairs (or small groups) on the game. It will be a great opportunity to learn from each other and a good experience for the project(s) to come! More information will be made available as the game (and the course) progresses.

Discord:

To facilitate remote communication between students and teachers alike, some courses use an online communication system named Discord. This software is available for a wide range of operating systems as an installable application, or in your browser at https://discord.com/.

You will find the invite link which you can use to join the Discord server on Canvas. When you click on it, you will receive access to the DACS Discord Server and can immediately start to interact with fellow students, teachers and teaching assistants.

Rules and Etiquette

Teachers can be recognised by their role (and red color).

To make Discord a safe, useful and fun environment for everyone please take care of the etiquette and rules (read also Paragraph 7 about your class behavior):

- Change your nickname in the DACS Server to your own name, you may use any combination of first and last name. If you do not change your name, you can not participate in the Discord server or (if you do not comply) may be kicked from it.

- Stay polite, even though it is easier to express yourself over a digital medium, regular terms of behaviour to teachers and fellow students apply.
- Do not post or share solutions to assignments, partial or otherwise.
- Please refrain from sending private messages to teachers. The teachers' main focus is helping students in the text and speech channels.
- Feel free to discuss the assignment and help other students as long as this does not lead to situations where plagiarism can arise.
- Teachers and TAs are - in principle - available on Discord during lab hours. If a teacher answers outside the time allocated for the course this is their choice and not mandatory. Moreover, we cannot offer any guarantee on the speed with which a teacher replies to a question.
- You are allowed to use emoticons, GIFs or stickers, however be mindful of the situation, it should not bother students and teachers. Teachers have the final say in the use of media in channels and whether they deem it appropriate and according to the class rules (see Paragraph 7).

5. Study workload

This course is worth **4 ECTS** points, which equates to **112 hours of study**. Based on this, we built the weekly plan below. As indicated there, we expect 8-12 hours of self-study per week. The estimated workload can be summarized as follows:

- 18 contact hours during Lectures and Live coding sessions
- 5 contact hours for tutorials
- 5 preparation hours (before each lecture/live coding session)
- 5 review hours (after each lecture/live coding session)
- 50 hours to complete the assignments
- 10+ hours for the exam preparation
- (optional) 7+ hours to complete the tutorials' game developments
- (optional) x hours to develop your our version of the game for the Awards Ceremony

Additionally, you have the chance to attend (contact hours):

- 10 office hours
- 4 Q&A sessions with TAs for coding-related doubts
- 2 "leisure" hours in the Game Awards ceremony

[it continues on the next page]

Rough weekly plan

Week	Day	Hours	Content	Type	Who	Book chapters
1	Tuesday 5/9	2	**Introduction to CS**	Lecture / Intro	Enrique	1,2,4,6 (check Canvas for details)
	at your own pace	1	**Variables and Methods**	Watch material	You!	
	Wednesday 6/9	2	>>	Live coding	Daniel	
	at your own pace	1	>>	Review material	You!	
	Thursday, 7/9	2	>>	Office hours	Enrique	
	Friday 8/9	1	>>	Tutorial	TA	
	at your own pace	6	>>	Assignment 1 (due 15/9)	You!	
2	at your own pace	1	**Conditionals**	Watch material	You!	2,3 (check Canvas for details)
	Tuesday 12/9	1	>>	Live coding	Enrique	
	at your own pace	1	>>	Review material	You!	
	Thursday, 14/9	2	>>	Office hours	Daniel	
	Friday 15/9	1	>>	Tutorial	TA	
	at your own pace	8	>>	Assignment 2 (due 22/9)	You!	
3	at your own pace	1	**Loops**	Watch material	You!	2,3 (check Canvas for details)
	Tuesday 19/9	2	>>	Live coding	Daniel	
	at your own pace	1	>>	Review material	You!	
	Thursday, 21/9	2	>>	Ask a TA	TAs	
	Thursday, 21/9	2	>>	Office hours	Daniel	
	Friday 22/9	1	>>	Tutorial	TA	
	at your own pace	8	>>	Assignment 3 (due 29/9)	You!	
4	at your own pace	1	**Arrays**	Watch material	You!	5 (check Canvas for details)
	Tuesday 26/9	2	>>	Live coding	Daniel	
	at your own pace	1	>>	Review material	You!	
	Wednesday, 27/9	2	**RECAP**	Q & A session	Daniel	
	Friday 29/9	1	>>	Tutorial	TA	
	Friday 29/9	2	>>	Office hours	Daniel	
	at your own pace	8	>>	Assignment 4 (due 6/10)	You!	
5	at your own pace	1	**Recursion**	Watch material	You!	6 (check Canvas for details)
	Monday 2/10	2	>>	Live coding	Enrique	
	Tuesday 3/10	2	**Branching**	Live coding	Enrique	
	at your own pace	1	>>	Review material	You!	
	Wednesday, 4/10	2	>>	Office hours	Enrique	
	Friday 6/10	1	>>	Tutorial	TA	
	Friday 6/10	2	>>	Ask a TA	TAs	
	at your own pace	10	>>	Assignment 5 (due 19/10)	You!	
6	**Project 1.1**					
7	Tuesday 17/10	2	**Revising all content**	Exam preparation	Daniel	
	at your own pace	10	Revising all content	Review material	You!	
	Wednesday 18/10	2	**Game award ceremony**	Special session	Enrique	
	at your own pace	10	Exam examples	Assignment 6 (due 19/10)	You!	
8	**TBA**	**20**	**Study and Exam**		**You!**	

Note: the due dates included in the Assignment rows are just indicative. The submission of the assignments is neither mandatory nor part of the grade.

6. Intended Learning Outcomes and Learning Goals

The Intended Learning Outcomes (ILOs) of the PrP course are the following:

Intended learning outcomes (ILOs)	Teaching & learning activities (TLAs)
Knowledge and understanding: Student will be able to explain the methodological and theoretical principles of basic procedural computer programming.	- Classical lectures
Applying knowledge and understanding: Students will be able to implement basic, procedural, non-object oriented, computer programs, using methods, arrays, loops, conditional statements and recursive execution.	- Programming exercises
Making judgement: Students will be able to judge the quality and correctness of simple procedural, non-object-oriented, computer programs.	- Classical lectures - Pair programming exercises
Communication: Students can discuss (oral and written) the correct use of standard procedural programming constructs and algorithmic basics.	- Classical lectures - Programming exercises
Learning skills: Students learn to recognize their own lack of knowledge and understanding and take appropriate action such as consulting additional material or other sources of help	- Programming exercises

These ILOs can be split into more concrete Learning Goals (LGs). The summary of these LGs, grouped by course module, is shown below:

- Module 1: Introduction to computer science
    - You know the four components of the Von Neumann Architecture
    - Control Unit
    - Arithmetic/Logic Unit
    - Memory
    - Input/Output
    - You understand the execution cycle on the Von Neumann Architecture
    - You understand the binary representation
    - You know the basics of assembler
    - You know the difference between assembly and high-level programming languages
    - You know the difference between compilation vs interpreted code
- Module 2: Variables and methods
    - You know the difference between a constant and a variable and know when to use which one
    - You know how to name, declare, and instantiate variables
    - You can use variables in your code effectively using operators
    - You know properties about variables and constants such as data types and scope
    - You know about casting, how and when to use it. Pitfalls and possible errors
- Module 3: Conditionals
    - You understand the control flow, how and when to use it
    - You understand the differences when comparing basic type variables and Strings (and other objects)
    - You understand the Boolean expression evaluation and the concept of Lazy evaluation
    - You understand the nested conditionals, how and when to use them
    - You understand the class Scanner, how and when to use it
    - You know some useful classes such as Math (random)
    - You know some methods for String manipulation

- Module 4: Loops
    - You know how to use different kinds of loops
        - for
        - while
        - do while
    - You understand the conditions under which loops run and stop
    - You are familiar with the pitfalls of loops and how to avoid them
    - You know how to write and recognize nested loops
- Module 5: Arrays
    - You know what arrays are and how to declare them
    - You can use an array to store and access values
    - You can combine loops and arrays to iterate over values
    - You are aware of the limitations of arrays
    - You know what multi-dimensional arrays are and how to use them
    - You know what pass-by-value means and what the consequences are
    - You know the difference between single variables and arrays
    - You are aware of common pitfalls when using arrays
- Module 6: Recursion
    - You understand the differences between "regular" loops and recursion
    - You understand the concepts of Base case and Recursive call, how and when to use them
    - You know the concept of mutual recursion
    - You understand the problem of (lack of) efficiency in recursive solutions and a possible solution (tabling)

7. Assessment

We expect all students to complete the quizzes and participate in class. Students are evaluated through a final exam taken using the software Testvision (and using UM Chromebooks) which is graded out of 10. The course assignments or any other proposed exercise (quizzes, homework, etc.) do not count for the final grade. However, around 2 out of 10 points from the exam will be allocated to an exercise closely related to the assignments. The structure of the exam can be estimated as follows:

- Theoretical questions + small coding exercises (identify errors, analyse a piece of code, fill in the gaps, etc.): ~50%
- Practical exercises (usually, 2-3 exercises):
    - Exercise closely related to the proposed assignments: ~20%
    - Pseudocode for "real-life" coding problems: ~10%
    - Java code for "real-life" coding problems: ~20%

The final grade is computed as follows:

final grade (out of 10) = exam grade

Resits

In the case of no passing grade during the normal period, you can take a resit exam (same format as the normal period exam). If you delivered your assignments during the normal period they still count towards your final grade. In this case, your final grade will be computed as follows:

resit final grade (out of 10) = resit exam

8. Class Policies (diversity/inclusivity, cheating)

Behavior in Class (online and offline)

- Let us know about your preferred names (if they differ from the ones in the course catalogue).
- We do not tolerate any form of discrimination or violence in classes. If we observe or hear about a person in class attacking, harassing or ridiculing someone else because of their outer appearance, language, age, origin, gender, sexual orientation, religion, nationality, abilities, or because of physical or mental impairments or disabilities, we will take action to stop this kind of behavior.
- We expect you to respect other participants in class, and to avoid any behavior that might impact other students' ability to do well in the class. This includes talking loudly or distracting people during lectures. You should feel free to turn your attention elsewhere if you want, but we expect you to never do so at the expense of other students' ability to focus on the class.
- Same rules apply to the class online environment: Pay attention to your comments, words, etc. when chatting during lectures and/or on Discord.
- Our own behavior in class should follow the rules given above, just as much as yours. In case we say or do something that contradicts these guidelines, please point out the mistake and be prepared to discuss the issue with us, either in class or in private.

Cheating/Honor Code

- No code or solutions are to be distributed to other students either electronically (i.e. e-mail) or on paper or posted online where they can easily be discovered (i.e. Facebook, webpages, forums, discussion groups, etc.)
- Be sure that you read the terms of the license for the code that you are reusing. Most code that is found openly on the internet has specific licenses that you should not violate.
- When there is doubt regarding the honorability of an action, you need to ask!
- Please do not copy your assignments and/or do not try to cheat! You won't learn any programming (and you will need it for the rest of your studies).

9. Tips & Tricks for the course

- Regardless of what fellow students will tell you, PrP (former CS1) is one of the toughest block 1 courses. Make sure that you have a good understanding of the content. Otherwise, it will be difficult to catch up.
- If you are having trouble with the contents of one lecture/tutorial, you can catch up on that lecture/tutorial at home. You can talk to other class participants, go through the lecture materials, or use the recommended or other online resources, or books to read up on specific topics. If, **after checking all the material proposed**, you need additional materials, feel free to talk to us. We will be happy to provide resources as needed.
- If you have no idea how to get started on an assignment, start by re-reading the relevant lecture contents and game assignment. You can also discuss the task with other students. If you ask for tips in the tutorial or *Ask a TA* sessions, the teaching assistants will help you develop your own approach to the problem.
- If you are unhappy throughout several sessions (lectures/tutorials) of the course and feel like you are not progressing, do not hesitate to talk to us. You can contact us in person or via email (see above how you can contact us). Together, we can figure out what the problem is and find solutions that work for you.
- Course lecturers enjoy teaching and are looking forward to teaching you the basics of programming. Any feedback for us is valuable so feel free to contact us.