GHENT UNIVERSITY

Course Specifications

Valid as from the academic year 2024-2025

Programming I (I700197)

Course size *(nominal values; actual values may depend on programme)*

Credits 4.0 **Study time 120 h**

Course offerings and teaching methods in academic year 2024-2025

A (semester 1)	Dutch	Gent	lecture
			seminar

Lecturers in academic year 2024-2025

Köse, Demir Ali	LA26	staff member
Verwaeren, Jan	LA26	lecturer-in-charge

Offered in the following programmes in 2024-2025	crdts	offering
Bachelor of Science in Bioscience Engineering Technology	4	A
Linking Course Master of Science in Bioscience Engineering Technology: Agriculture and Horticulture (main subject Horticulture)	4	A
Linking Course Master of Science in Bioscience Engineering Technology: Agriculture and Horticulture (main subject Plant and Animal Production)	4	A
Linking Course Master of Science in Bioscience Engineering Technology: Food Industry	4	A

Teaching languages

Dutch

Keywords

Programming, programming language Python, problem-oriented reasoning

Position of the course

In engineering, one often encouters scientific problems that require an automated processing of information. To be able to solve these problems, skills that allow an engineer to: (1) analyse a problem and propose a structured solution strategy; and (2) transform this strategy into a computer program (using a programming language) are crucial in modern-day engineering. In this course, students learn the principles of structured, modular programming and learn to apply these principles to solve problems in a time-efficient and automated manner. Aditionally, this course focuses on the translation of a task (presented in a natural language) into a sequence of instructions that can be executed by a computer. The programming language Python will be used in this course.

Contents

This course is a basic programming course in Python. Students learn to develop source code (and computer programs) that can assist them in solving complex problems. Therefore, students need to (1) learn the syntax and semantics of a programming language (Python in this course); and (2) gain insight into the reasoning process that allows to translate problems into a task that can be implemented in a computer language. More precisely, students learn to:

- use the basic components of Python (data types, variables and operators)
- perform arithmetic and logical operations
- use control (conditional and repetitive execution)
- design and of functions
- use built-in data structures (lists, dictionaries, sets and tuples)

Initial competences

No initial competences are required, however, some basic skills w.r.t. modern-day use of a personal computer are recommended.

Final competences

1 The student can translate a well-defined task, described in a natural language, into source code in the programming language Python (correct use of syntax, data structures, control, functions and scripts).
2 The student can design (and implement) a tailor-made algorithm to solve a specific (simple) problem.
3 The student can interpret, test and debug a given source code.
4 The student has insight into the use of Python as a scientific programming environment.
5 The student can transfer the acquired programming skills to other programming languages and scientific software.

Conditions for credit contract

Access to this course unit via a credit contract is determined after successful competences assessment

Conditions for exam contract

This course unit cannot be taken via an exam contract

Teaching methods

Seminar, Lecture

Extra information on the teaching methods

- Plenary lectures (12 h) and practical exercises (30 h, PC-labs).
- The students will complete a group work (4 students per group). Each group submits a written report, the submission is followed by an oral defense. This group work is part of the permanent evaluation process.

Study material

Type: Syllabus

Name: Course notes Programmeren I
Indicative price: Free or paid by faculty
Optional: no
Language : Dutch
Available on Ufora : Yes

Type: Slides

Name: Lecture slides Programming I
Indicative price: Free or paid by faculty
Optional: no
Language : Dutch
Available on Ufora : Yes

Type: Software

Name: Python interpreter and VSCode
Indicative price: Free or paid by faculty
Optional: no
Available on Athena : No
Online Available : Yes

References

H.P. Langtangen (2009), A primer on Scientific Programming with Python. Springer, 686p.
W. Punch and R. Enbody, 2nd edition (2013), The Practice of Computing using Python, Pearson, 764p.

Course content-related study coaching

- The practical course notes allow students to complete the PC-labs at their own pace. Teaching assistants are available to assist the students during the PC-labs.
- Students have the possibility to ask additional questions after each plenary lecture or PC-lab. Alternatively, an appointment can be made with one of the lecturers.
- Students are taught how to install a Python interpreter on their personal computer.

Assessment moments

end-of-term and continuous assessment

Examination methods in case of periodic assessment during the first examination period

Written assessment

Examination methods in case of periodic assessment during the second examination period

Written assessment

Examination methods in case of permanent assessment

Oral assessment, Assignment

Possibilities of retake in case of permanent assessment

examination during the second examination period is possible

Extra information on the examination methods

- The end-of-term evaluation consists of an open book exam that consists of a number of hands-on programming exercises. A mark is given based on the correctness and style of the source code that is submitted.
- The permanent evaluation is based on the evaluation of a group work. The students need to complete a (small) programming project (4 students per group), submit a report and source code. The report and the source code provide the basis of an in-depth discussion in the format of an oral exam. Each student is evaluated independently.
- The submission deadline for the group work and the date of the oral exam will be communicated to the students at the start of the semester (the oral exam takes place around the 8th week of the semester).

Calculation of the examination mark

The final score is the weighted arithmetic mean of the evaluation moments according to the percentages below:

- End-of-term evaluation (programming exam): 85%
- Permanent evaluation (group work): 15%

Bijkomende toelichting:

- End-of-term evaluation: evaluation by means of an open book exam. Submitted source code is corrected and a mark is given.
- Permanent evaluation: submitted source code and a written report (of the group work) provide the basis for a small oral exam. Students that fail to submit the source code or report, or do not participate at the oral exam get a mark of zero.
- In the second examination period, students are given the opportunity to re-submit the project that is part of the permanent evaluation (task). A new mark will be given based on an (individual) oral defense.