Programming Paradigms

2025 - 2026

Is part of the next programmes:

- M0012004 Master of Computer Science: Software Engineering
- M0012005 Master of Computer Science: Data Science and Artificial Intelligence
- M0012006 Master of Computer Science: Computer Networks
- M0032004 Master of Mathematics: Financial and Applied Mathematics
- M0048004 Master of Computer Science: Software Engineering
- M0048005 Master of Computer Science: Data Science and Artificial Intelligence
- M0048006 Master of Computer Science: Computer Networks
- M0090004 Master of Teaching in Science and Technology: Computer Science
- U0001008 Courses open to exchange students in Sciences

Course Code:	2001WETPPA
Study Domain:	Computer Science
Semester:	2E SEM

Contact Hours:	45
Credits:	6
Study Load (hours):	168
Contract Restrictions:	No contract restriction
Language of Instructions:	ENG
Lecturer(s):	T Guillermo Alberto Perez
Examperiod:	exam in the 2nd semester
ONEVEN	
Bi-annual course:	Thought in academic years starting in an odd year

1. Prerequisites *

speaking and writing of:

- English

reading and comprehending of:

- English

general notion of the basic concepts of

programming languages, such as: syntax and semantics, recursion, data structures

2. Learning outcomes *

- You know the basic properties of the lambda calculus as a computational model.

- You understand the concept of confluence as well as its importance in the lambda calculus.
- You are able to explain the role of unification, resolution, and proof trees in logic programming.
- You are able to compute in the lambda calculus by applying beta-reductions to transform expressions.
- You are able to code simple programs in logic and functional programming languages.
- You are able to evaluate which programming paradigm is best suited for a coding task.
- You are more open minded in terms of learning and adopting new programming languages and paradigms as well as their theoretical foundations.

3. Course contents *

The course covers several programming paradigms that play an important role in computer science and in particular in artificial intelligence. The emphasis is on functional and logical programming languages.
First the theoretical foundation of the functional paradigm is treated in the form of a brief introduction into the lambda-calculus. Then, an introduction is given into the language Haskell. In the treatment of the logical paradigm, the emphasis is on the language Prolog: its advantages and limitations.

4. International dimension *

- This course stimulates international and intercultural competences.

5. Teaching method and planned learning activities

5.1 Used teaching methods *

Class contact teaching

- Lectures
- Practice sessions

Personal work

Assignments

- Individually

Paper

- Individually

5.2 Planned learning activities and teaching methods

The course consists of 3 parts of roughly equal size, in which we consider consecutively the lambda-calculus, the functional language Haskell and the logical language Prolog. The theoretical topics will be covered by way of lectures. For Haskell and Prolog there are also individual assignments, apart from the practice sessions. There will also be a paper regarding other programming paradigms which you will have to prepare and present by the end of the semester.

5.3 Facilities for working students *

Classroom activities

- no specific facilities

6. Assessment method and criteria *

6.1 Used assessment methods *

Examination

- Written examination without oral presentation
- - Closed book

- - Multiple-choice
- - Exercises

Continuous assessment

- Assignments (no second assessment period)

Other assessment methods

- Written assignment (no second assessment period)
- Presentation (no second assessment period)

6.2 Assessment criteria *

For the part on lambda-calculus the students must be able to demonstrate the concepts and results discussed on simple examples, and must be able to give their motivation. For Haskell and Prolog they have to be able to write correct, functioning programs using the language constructs that are introduced, albeit programs of limited size (in the order of a half, at most a full page).
The Haskell and Prolog assignments are taken into account for 20% the marks, the paper on other programming paradigms (as well as its presentation) count for 20% of the marks --- both can only be turned in during the first session. The remaining 60% of the marks is covered by the theory exam.
The theory exam is closed book. However, students may bring **one sheet** with notes on the course contents.
Regarding the structure of the exam: For all learning goals which have a component of knowledge, there is at least one multiple choice question from previous exams and/or a problem-solving question which can be easily answered using content from the slides (or their cheat-sheet which they can bring to the exam). For all learning goals which have a component of ability, there is at least one problem-solving question which requires step-by-step realization of some algorithm covered during the class. For multiple-choice questions, no cut-off point is used and negative points are not assigned for incorrect answers.
Students may use generative AI tools for their assignment. In an oral exam of the assignment or during a contact moment, students are expected to elaborate on their assignment (and how they used generative AI).

7. Study material

7.1 Required reading *

Syllabus (via Blackboard)

7.2 Optional reading

The following study material can be studied voluntarily :

- Foundations of Functional Programming. Lawrence C. Paulson. 2000. https://www.cl.cam.ac.uk/~lp15/papers/Notes/Founds-FP.pdf
- Lambda Calculus. Tobias Nipkow. 2018. https://www21.in.tum.de/teaching/lambda/WS1718/lambda.pdf
- From Logic Programming to Prolog. Krzysztof R. Apt. 1997. London: Prentice Hall.
- Modern Programming Languages: A Practical Introduction. Adam Brooks Webber. 2010. Franklin, Beedle & Associates, Incorporated.
- http://learnyouahaskell.com/

8. Contact information *

docent: guillermoalberto.perez@uantwerpen.be
teaching assistant: brent.vanbladel@uantwerpen.be

9. Tutoring