Research Software Engineering in Python for Reproducible Computational Science

Course description sheet

Basic information

Field of study Computer Physics **Major** - **Organisational unit** Faculty of Physics and Applied Computer Science **Study level** First-cycle (engineer) programme **Form of study** Full-time studies **Profile** General academic	**Didactic cycle** 2025/2026 **Course code** JCMPS.li20.18703.25 **Lecture languages** English **Mandatoriness** Elective **Block** Elective Modules in Foreign Language **Course related to scientific research** Yes

Course coordinator	Sylwester Arabas
Lecturer	Sylwester Arabas

Period	**Method of verification of the learning outcomes**	**Number of ECTS credits**
Semester 6	Completing the classes **Activities and hours** Lectures: 10 Laboratory classes: 14 Project classes: 6	3

Goals

C1	To get students acquainted with the challenges and realities of result reproducibility in computational research
C2	To equip students with practical research software engineering skills enabling reproducibility

Course's learning outcomes

Code	Outcomes in terms of	Learning outcomes prescribed to a field of study	Methods of verification
Knowledge – Student knows and understands:			
W1	Student knows and understands the implications of scientific result reproducibility or lack thereof	CMP1A_W01, CMP1A_W03, CMP1A_W04	Participation in a discussion, Test, Project, Completion of laboratory classes
W2	Student understands the technical, legal and practical prerequisites for result reproducibility	CMP1A_W01, CMP1A_W04	Participation in a discussion, Test, Project, Presentation, Completion of laboratory classes
W3	Student understands the strengths and limitations of both Python and alternative computational ecosystems	CMP1A_W01, CMP1A_W03, CMP1A_W04	Participation in a discussion, Test, Project, Completion of laboratory classes
Skills – Student can:			
U1	Student can apply software design patterns in Python in order to make the developed code unit-testable	CMP1A_U01, CMP1A_U02, CMP1A_U03	Project, Completion of laboratory classes
U2	Student can engineer a Python package using tools for automating versioning, dissemination and persistent archival	CMP1A_U01, CMP1A_U03	Project, Completion of laboratory classes
U3	Student can engineer a computational workflow that is anonymously and independently executable in the cloud	CMP1A_U01, CMP1A_U02, CMP1A_U03, CMP1A_U05	Project, Presentation, Completion of laboratory classes
Social competences – Student is ready to:			
K1	Student is ready to instruct others on how to reproduce computations using tools from the Python ecosystem	CMP1A_K01, CMP1A_K02, CMP1A_K03	Participation in a discussion, Project, Presentation

Program content ensuring the achievement of the learning outcomes prescribed to the module

The course will explore the application of programming techniques in scientific research, with a focus on software engineering concepts and methods that enhance reliability, auditability, maintainability, and interoperability in computational sciences. It will cover practical solutions for collaborative development, as well as strategies for the public dissemination and archival of software. Additionally, basic legal concepts related to software licensing will be introduced.

Student workload

Activity form	Average amount of hours* needed to complete each activity form
Lectures	10
Laboratory classes	14
Project classes	6

Contact hours	5
Preparation for classes	10
Preparation of project, presentation, essay, report	20
Realization of independently performed tasks	10

Student workload	**Hours** 75
Workload involving teacher	**Hours** 30

* hour means 45 minutes

Program content

No.	Program content	Course's learning outcomes	Activities
1.	The lecture will cover the following themes: • reproducibility in computational research as a major principle underpinning the scientific method; • technical, practical and legal challenges to reproducibility; • software licensing in the context of computational research; • cloud computing and collaborative development tools; • computational notebooks and the challenges and opportunities in their maintanance; • Python packaging ecosystem; • SOLID, DRY, KISS and related principles; • software design patterns and testing principles; • Python vs. other programming languges; • selected Python packages applicable in computational research; • best practices in scientific research software engineering; • selected data and metadata storage formats.	W1, W2, W3	Lectures

No.	Program content	Course's learning outcomes	Activities
2.	Laboratory classes will feature hands-on training in: software design patterns, test automation, documentation-generation tools, tools for improving code quality, Python packaging, code versioning and archival, workflows for computations in the cloud, Python Just-in-time compilation tools, Jupyter notebooks and the tools for their maintenance, vector graphics and animations in Jupyter notebooks. All coursework will be carried out in Python.	W3, U1, U2, U3	Laboratory classes
3.	During the project classes, students will be tasked with reproducing a computational result (e.g., a plot or table) from a peer-reviewed journal paper. A pool of proposed papers will be provided, though students may also choose papers outside of this pool. The project will be divided into four graded subtasks: 1. A brief presentation summarizing the study, the selected figure, and the required tools. 2. The development and public dissemination of the program code (via GitHub) needed to reproduce the figure, including automated tests to confirm the results match the expected values, along with textual instructions for peers. 3. Independent execution of code developed by another student, followed by the preparation of a summary of comments on the code and its results. 4. A presentation discussing the challenges encountered and the results achieved.	W2, U1, U3, K1	Project classes

Extended information/Additional elements

Teaching methods and techniques :

Discussion, Peer assessment, Problem Based Learning, Project Based Learning, Demonstration, Lectures

Activities	Methods of verification	Credit conditions
Lectures	Test	
Lab. classes	Completion of laboratory classes	
Project classes	Participation in a discussion, Project, Presentation	

Conditions and the manner of completing each form of classes, including the rules of making retakes, as well as the conditions for admission to the exam

Students with two or less unexcused absences on lab and project classes are admitted to the final test.
Laboratory assignments are extensions to excercises performed during the class and are to be submitted within two weeks.

Method of determining the final grade

The final grade is an equal-weight mean of grades from: (i) lecture-material test, (ii) laboratory assignments and (iii) project. A minimum of 50% of points from each component is required.

Manner and mode of making up for the backlog caused by a student justified absence from classes

Excused absences from classes requires the student to independently master the material covered during these classes.

Prerequisites and additional requirements

- programming in Python
- basic concepts from applications of computational methods in scientific research

Rules of participation in given classes, indicating whether student presence at the lecture is obligatory

Active participation in the lectures is highly encouraged - including questions and discussion.
Attendence on laboratory and project classes is obligatory. Two unexcused absences are acceptable.

Literature

Obligatory

1. Irving, Hertweck, Johnston, Ostblom, Wickham & Wilson 2021: "Research Software Engineering with Python" (https://third-bit.com/py-rse)

Optional

1. GMD executive editors 2019: "The publication of geoscientific model developments v1.2" (https://doi.org/10.5194/gmd-12-2215-2019)
2. Merali 2010 (Nature 467): "Error... Why scientific computing does not compute" (https://doi.org/10.1038/467775a)
3. Perkel 2021 (Nature 593) "Reactive, reproducible, collaborative: computational notebooks evolve" (https://doi.org/10.1038/d41586-021-01174-w)
4. Perkel 2020 (Nature 584) "Challenge to scientists: does your ten-year-old code still run?" (https://doi.org/10.1038/d41586-020-02462-7)

Scientific research and publications

Publications

1. Derlatka, Manna, Bulenok, Zwicker & Arabas 2024, SoftwareX (https://doi.org/10.1016/j.softx.2024.101897): "Numba-MPI v1.0: Enabling MPI communication within Numba/LLVM JIT-compiled Python code"
2. D'Aquino, Arabas, Curtis, Vaishnav, Riemer & West 2024, SoftwareX (https://doi.org/10.1016/j.softx.2023.101613): "PyPartMC: A Pythonic interface to a particle-resolved, Monte Carlo aerosol simulation framework"
3. Bartman, Banaskiewicz, Drenda, Manna, Olesik, Rozwoda, Sadowski & Arabas 2022, JOSS (https://doi.org/10.21105/joss.03896): "PyMPDATA v1: Numba-accelerated implementation of MPDATA with examples in Python, Julia and Matlab"
4. Bartman, Bulenok, Gorski, Jaruga, Lazarski, Olesik, Piasecki, Singer, Talar & Arabas, 2022, JOSS (https://doi.org/10.21105/joss.03219): "PySDM v1: particle-based cloud modelling package for warm-rain microphysics and aqueous chemistry"

Learning outcomes prescribed to a field of study

Code	Content
CMP1A_K01	student is ready for reviewing their own competences and external content, gathered from various sources, in context of the state-of-the-art in science and technology
CMP1A_K02	student is ready to transfer and share their professional expertise to the industry and society for the sake of science commercialization and public interest
CMP1A_K03	student is ready to take responsibility for his professional activity and to obey legal and ethical rules pertinent to professional environment
CMP1A_U01	student can handle complex problems of science and technology using appropriate tools of scientific computing
CMP1A_U02	student can undertake new approaches for non-typical or novel scientific and technological problems
CMP1A_U03	student can perform analytical breakdown of technical or physical problem to propose cost and time-efficient solutions
CMP1A_U05	student can lead a scientific, interdisciplinary project alone or in collaboration, with the awareness of the role of self-directed and lifelong learning for success
CMP1A_W01	student knows and understands mathematical methods used in science and engineering, particularly in: scientific computing and data handling
CMP1A_W03	student knows and understands algorithms, numerical methods, programming techniques and IT tools used for computer physics
CMP1A_W04	student knows and understands the role of science and engineering for socio-economic environment and knowledge-based society, taking into account ethical and legal paradigms