STUDIJŲ DALYKO APRAŠAS

Studijų dalyko pavadinimas	Studijų dalyko kodas
Taikomasis ir mokslinis programavimas Python kalba	

Dėstytojas (-ai)	Dalyką vykdantis padalinys
Koordinatorius: Irus Grinis	Vilniaus universitetas Matematikos ir informatikos fakultetas Informatikos institutas

Studijų pakopa	Dalyko pobūdis
Pirmoji	Pasirenkamasis

Dėstymo būdas	Dėstymo semestras	Dėstymo kalba
Kontaktinis	2, 4 arba 6 semestras	Anglų / lietuvių

Būtinosios išankstinės žinios
Procedūrinis programavimas

Kreditų skaičius	Studento darbo krūvis	Kontaktinės valandos	Savarankiškas darbas
5	130	64	66

Dalyko tikslas ir ugdomos kompetencijos

Tikslas. Ugdyti gebėjimą projektuoti, įgyvendinti, testuoti ir pristatyti patikimus Python kalba parašytus taikomųjų ir mokslinių uždavinių sprendimus. Dalyke pagrindinės Python programavimo ir programinės įrangos projektavimo žinios siejamos su skaitiniais skaičiavimais, mokslinių duomenų apdorojimu, vizualizavimu ir skaičiuojamuoju modeliavimu. Studentai mokosi naudotis plačiai taikomomis atvirojo kodo mokslinio programavimo bibliotekomis, kritiškai jas vertinti, tikrinti skaitinius rezultatus ir atlikti atkuriamuosius skaičiavimo eksperimentus.

Bendrosios kompetencijos:

- bendravimas ir bendradarbiavimas (BK1);
- mokymasis visą gyvenimą (BK2).

Dalykinės kompetencijos:

- esminių konceptualiųjų pagrindų žinios ir gebėjimai (DK4);
- programinės įrangos kūrimo žinios ir gebėjimai (DK5);
- technologinės ir metodologinės žinios bei gebėjimai, profesinė kompetencija (DK6).

Studijų rezultatai, mokymo ir vertinimo metodai

Studijų rezultatai: studentai gebės	Mokymo ir mokymosi metodai	Vertinimo metodai
Kurti gerai struktūrizuotas Python programas, naudojant funkcijas, modulius, klases, išimtis ir tinkamas duomenų struktūras.	Paskaitos; trumpos programavimo užduotys; kodo peržiūra.	Laboratoriniai darbai; tarpinis atsiskaitymas; egzaminas raštu.
Kurti atkuriamas darbo sekas, naudojant interaktyviąsias užrašines, izoliuotas aplinkas, dokumentaciją, testus ir versijų valdymą.	Vadovaujami laboratoriniai darbai; savarankiška praktika.	Laboratorinių darbų pateikimas; projekto saugykla ir gynimas.
Įkelti, transformuoti ir analizuoti skaitinius bei lentelinius duomenis naudojant NumPy ir pandas.	Demonstracijos; probleminis mokymasis laboratorinių darbų metu.	Laboratorinės užduotys; tarpinis atsiskaitymas.
Taikyti masyvus, kompleksinius skaičius ir matricas, atpažinti skaitinio tikslumo ir stabilumo problemas.	Matematiniai pavyzdžiai; atvejų analizė; skaičiavimo eksperimentai.	Laboratorinės užduotys; egzaminas raštu.
Įgyvendinti ir palyginti pagrindinius šaknų radimo, integravimo, diferencialinių lygčių sprendimo, regresijos ir optimizavimo skaitinius metodus.	Paskaitos; eksperimentai su SciPy; savarankiškas darbas.	Laboratorinės užduotys; egzaminas raštu; projektas.
Vizualizuoti mokslinius rezultatus, vertinti neapibrėžtumą ir tikrinti modelius pagal etaloninius arba nepriklausomai gautus rezultatus.	Duomenų tyrimai; eksperimentai; dėstytojo moderuojamos diskusijos.	Laboratorinių darbų ataskaitos; projekto gynimas.

Studijų rezultatai: studentai gebės	Mokymo ir mokymosi metodai	Vertinimo metodai
Suformuluoti ir pristatyti taikomojo mokslinio programavimo problemą, pagrįsti pasirinktus metodus ir aptarti jų ribotumus.	Komandinis projektas; literatūros paieška; pristatymai.	Projekto ataskaita, kodas ir žodinis gynimas.

Dalyko turinys: temų suskirstymas

16 studijų savaičių planas. Valandų paskirstymas pagal užsiėmimų rūšis ir savarankišką darbą.

Nr.	Tema	Pask.	Sem.	Prat.	Lab.	Kont.	Savar.
1	Mokslinio programavimo Python ekosistema: interpretatorius, scenarijai ir interaktyviosios užrašinės; virtualiosios aplinkos; paketai; atkuriamumo pagrindai.	2			2	4	4
2	Python pagrindai po procedūrinio programavimo: reiškiniai, valdymo struktūros, funkcijos, dekoratoriai, vardų sritys ir moduliai.	2			2	4	4
3	Sąrašai, kortežai, žodynai ir aibės; glaustosios konstrukcijos (comprehensions), iteratoriai, generatoriai; klaidos ir išimtys.	2			2	4	4
4	Programų organizavimas: failai ir keliai, klasės, duomenų klasės (dataclasses), tipų anotacijos, vienetiniai testai, dokumentacija ir projekto struktūra.	2			2	4	4
5	Skaičių vaizdavimas: sveikųjų ir slankiojo kablelio skaičių aritmetika, tikslumas, apvalinimo paklaidos; mokslinių duomenų formatai.	2			2	4	4
6	NumPy I: masyvai, duomenų tipai (dtypes), indeksavimas, pjūviai, transliavimas (broadcasting), vektorizavimas ir našumas.	2			2	4	4
7	NumPy II: vektoriai, matricos, kompleksinių skaičių masyvai, tiesinių lygčių sistemos, matricų skaidiniai ir sąlygotumas.	2			2	4	4
8	pandas: lenteliniai duomenys, trūkstamos reikšmės, grupavimas ir lentelių jungimas; praktinis tarpinis atsiskaitymas.	2			2	4	4
9	Mokslinė vizualizacija: Matplotlib, koordinačių ašys ir matavimo vienetai, kelios duomenų sekos, statistiniai grafikai ir iliustracijų kokybė.	2			2	4	4
10	SciPy I: lygčių šaknų radimas, interpoliavimas, skaitinis diferencijavimas ir integravimas.	2			2	4	4
11	SciPy II: pradinės reikšmės uždaviniai ir paprastųjų diferencialinių lygčių modeliai; skaitinio sprendimo tikslumas.	2			2	4	4
12	Optimizavimas, mažiausiųjų kvadratų metodas, modelių pritaikymas ir regresijos pagrindai; modelio atitikties vertinimas.	2			2	4	4
13	Atsitiktinumas, aprašomoji statistika, imčių sudarymas, Monte Karlo modeliavimas ir neapibrėžtumo įvertinimas.	2			2	4	4
14	Simboliniai ir skaitiniai darbo metodai: SymPy, analitinis rezultatų patikrinimas, skaičiavimo sudėtingumas ir programų profiliavimas.	2			2	4	4
15	Integruotos taikomosios užduotys: pasirinktos biologijos, fizikos ir kvantinės informacijos skaičiavimo problemos; projekto rengimas.	2			2	4	4
16	Mokslinių rezultatų pateikimas, atkuriamos interaktyviosios užrašinės ir programos kodas; projekto pristatymas, gynimas ir dalyko apibendrinimas.	2			2	4	6
	Baigiamasis egzaminas (raštu)					**2**	
	IŠ VISO	**32**			**32**	**64**	**66**

Santrumpos. Pask. – paskaitos; Sem. – seminarai; Prat. – pratybos; Lab. – laboratoriniai darbai; Kont. – kontaktinės valandos; Savar. – savarankiško darbo valandos.

Vertinimo strategija

Vertinimo komponentas	Svoris	Atsiskaitymo laikas	Vertinimo kriterijai
Laboratoriniai darbai	40 %	Visą semestrą	Keturi vertinami darbai (kiekvienas po 10 %): (1) Python programų projektavimas ir testavimas; (2) skaitiniai masyvai ir tiesinė algebra; (3) duomenų analizė ir vizualizavimas; (4) skaitinis modeliavimas ir rezultatų patikra. Vertinimo kriterijai: sprendimo teisingumas – 40 %, tinkami metodai ir rezultatų tikrinimas – 25 %, kodo kokybė ir testai – 20 %, atkuriamumas ir dokumentacija – 15 %.
Tarpinis atsiskaitymas	15 %	8-oji savaitė	Individualus teorinis ir praktinis atsiskaitymas, kuriame tikrinamos Python programavimo, duomenų struktūrų, skaičių vaizdavimo, NumPy pagrindų ir pradinio duomenų apdorojimo žinios bei gebėjimai.
Taikomasis mokslinis mini projektas	20 %	Projekto gynimas 16-ąją savaitę	Individualus arba 2–3 studentų grupės projektas, pateikiant programos kodą ir glaustą mokslinę ataskaitą. Studentai suformuluoja problemą, atlieka analizę arba modeliavimą, patikrina rezultatus, pateikia vaizdinius įrodymus, aptaria ribotumus ir apgina savo sprendimus. Vertinimo kriterijai: problema ir metodas – 25 %, įgyvendinimas – 25 %, rezultatų patikra – 25 %, ataskaita ir atkuriamumas – 15 %, pristatymas ir gynimas – 10 %.
Baigiamasis egzaminas raštu	25 %	Egzaminų sesija	Atviri, trumpo atsakymo ir praktinių uždavinių sprendimo klausimai, kuriais tikrinamas gebėjimas interpretuoti kodą ir skaitinius rezultatus, parinkti algoritmus, vertinti skaitinį tikslumą, taikyti duomenų apdorojimo darbo sekas ir paaiškinti mokslinio programavimo principus.
IŠ VISO	**100 %**		
Reikalavimai dalyko vertinimui eksterno būdu			Studentas turi surinkti bent 50 procentų balų semestro metu tam, kad galima būtų laikyti egzaminą eksternu.

Privaloma ir rekomenduojama literatūra

Autorius / organizacija	Metai	Leidinys / šaltinis	Leidimas / leidykla / nuoroda
PRIVALOMA LITERATŪRA			
W. McKinney	2022	Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter	3-iasis leidimas; O'Reilly Media https://wesmckinney.com/book/
J. VanderPlas	2022	Python Data Science Handbook	2-asis leidimas; O'Reilly Media
Python Software Foundation	Internete	The Python Tutorial and Python Standard Library	Aktuali dokumentacija https://docs.python.org/3/
NumPy kūrėjai	Internete	NumPy documentation: user guide and reference	Aktuali dokumentacija https://numpy.org/doc/stable/
REKOMENDUOJAMA LITERATŪRA			
SciPy kūrėjai	Internete	SciPy documentation: user guide and API reference	Aktuali dokumentacija https://docs.scipy.org/doc/scipy/
pandas kūrėjai	Internete	pandas user guide	Aktuali dokumentacija https://pandas.pydata.org/docs/
Matplotlib kūrėjų komanda	Internete	Matplotlib documentation and tutorials	Aktuali dokumentacija https://matplotlib.org/stable/
SymPy kūrėjų komanda	Internete	SymPy documentation	Aktuali dokumentacija https://docs.sympy.org/latest/

Papildomi moksliniai arba konkrečiai dalykinei sričiai skirti straipsniai ir duomenų rinkiniai pateikiami semestro metu, atsižvelgiant į laboratorinių darbų bei projektų temas.

COURSE UNIT DESCRIPTION

Course unit title	Course unit code
Applied and Scientific Programming in Python	

Lecturer(s)	Department where the course unit is delivered
Coordinator: Irus Grinis	Institute of Computer Science Faculty of Mathematics and Informatics Vilnius University

Cycle	Type of the course unit
First	Optional

Mode of delivery	Semester or period of delivery	Language of instruction
Face-to-face	2, 4 or 6 semesters	English/Lithuanian

Prerequisites
Procedural programming.

Credits allocated	Student workload (hours)	Contact hours	Individual work (hours)
5	130	64	66

Purpose of the course unit and competences to be developed

Purpose. To develop the ability to design, implement, test and communicate reliable Python solutions to applied and scientific problems. The course connects core Python programming and software design with numerical computation, scientific data processing, visualisation and computational modelling. Students learn to use and critically evaluate widely adopted open-source scientific libraries, to validate numerical results and to carry out reproducible computational experiments.

Generic competences:

- communication and collaboration (GK1);
- life-long learning (GK2).

Specific competences:

- knowledge and skills of underlying conceptual basis (SK4);
- software development knowledge and skills (SK5);
- technological and methodological knowledge and skills, professional competence (SK6).

Learning outcomes, teaching and assessment methods

Learning outcomes: students will be able to	Teaching and learning methods	Assessment methods
Develop well-structured Python programs using functions, modules, classes, exceptions and suitable data structures.	Lectures; short programming exercises; code review.	Laboratory work; midterm; written exam.
Create reproducible workflows with notebooks, isolated environments, documentation, tests and version control.	Guided laboratories; independent practice.	Laboratory submissions; project repository and defence.

Learning outcomes: students will be able to	Teaching and learning methods	Assessment methods
Load, transform and analyse numerical and tabular data using NumPy and pandas.	Demonstrations; problem-based laboratory work.	Laboratory tasks; midterm.
Use arrays, complex numbers and matrix operations, and recognise numerical precision and stability issues.	Mathematical examples; case studies; computational experiments.	Laboratory tasks; written exam.
Implement and compare basic numerical methods for roots, integration, differential equations, regression and optimisation.	Lectures; experiments with SciPy; individual work.	Laboratory tasks; written exam; project.
Visualise scientific results, assess uncertainty and check models against reference or independently obtained results.	Data studies; experiments; guided discussion.	Laboratory reports; project defence.
Formulate and present an applied scientific programming problem, justify methods and discuss limitations.	Team project; literature search; presentations.	Project report, code and oral defence.

Course content: breakdown of topics

16-week study plan. Distribution of lectures, laboratory work, contact hours and individual work.

No.	Topic	Lect.	Sem.	Pract.	Lab.	Cont.	Indiv.
1	Scientific Python ecosystem: interpreter, scripts and notebooks; virtual environments; packages; reproducibility basics.	2			2	4	4
2	Python essentials after procedural programming: expressions, control flow, functions, decorators, namespaces and modules.	2			2	4	4
3	Lists, tuples, dictionaries, sets; comprehensions, iterators, generators; errors and exceptions.	2			2	4	4
4	Program organisation: files and paths, classes, dataclasses, typing, unit tests, documentation and project structure.	2			2	4	4
5	Numerical representation: integer and floating-point arithmetic, accuracy, rounding error; scientific data formats.	2			2	4	4
6	NumPy I: arrays, dtypes, indexing, slicing, broadcasting, vectorisation and performance.	2			2	4	4
7	NumPy II: vectors, matrices, complex-valued arrays, linear systems, decompositions and conditioning.	2			2	4	4
8	pandas: tabular data, missing values, grouping and joins; practical midterm assessment.	2			2	4	4
9	Scientific visualisation: Matplotlib, axes and units, multiple series, statistical plots, figure quality.	2			2	4	4
10	SciPy I: root finding, interpolation and numerical differentiation and integration.	2			2	4	4
11	SciPy II: initial-value problems and ordinary differential equation models; numerical solution accuracy.	2			2	4	4
12	Optimisation, least squares, model fitting and introductory regression; assessing model fit.	2			2	4	4
13	Randomness, descriptive statistics, sampling, Monte Carlo simulation and uncertainty estimation.	2			2	4	4

No.	Topic	Lect.	Sem.	Pract.	Lab.	Cont.	Indiv.
14	Symbolic and numerical workflows: SymPy, analytic cross-checks, complexity and profiling.	2			2	4	4
15	Integrated applications: selected computational problems from biology, physics and quantum information; project development.	2			2	4	4
16	Scientific reporting, reproducible notebooks and code; project presentation, defence and course synthesis.	2			2	4	6
	Final exam (written)					**2**	
	TOTAL	**32**			**32**	**64**	**66**

Abbreviations. Lect. – lectures; Sem. – seminars; Pract. – practical classes; Lab. – laboratory work; Cont. – contact hours; Indiv. – individual work hours.

Assessment strategy

Assessment component	Weight	Deadline	Assessment criteria
Laboratory assignments	40%	Throughout the semester	Four assessed assignments (each worth 10%): (1) Python design and testing, (2) numerical arrays and linear algebra, (3) data analysis and visualisation, (4) numerical simulation and validation. Assessment emphasises correctness (40%), appropriate methods and verification (25%), code quality and tests (20%), and reproducibility and documentation (15%).
Midterm assessment	15%	Week 8	Individual written/practical assessment of Python programming, data structures, numerical representation, NumPy fundamentals and introductory data handling.
Applied scientific mini-project	20%	Project defence in week 16	Individual or 2-3-student team project, supported by code and a concise scientific report. Students formulate a problem, implement an analysis or simulation, validate results, present visual evidence, discuss limitations and defend their choices. Criteria: problem/method (25%), implementation (25%), result validation (25%), report/reproducibility (15%), presentation/defence (10%).
Final written exam	25%	Institutional examination period	Open, short-answer and practical problem-solving questions testing interpretation of code and numerical results, selection of algorithms, precision, data workflows and scientific programming principles.
TOTAL	**100%**		
For external students			In order to be eligible to sit the examination externally, a student must achieve a minimum score of 50% during the semester.

Required and recommended reading

Author / organisation	Year	Title / resource	Edition / publisher / URL
REQUIRED READING			
W. McKinney	2022	Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter	3rd ed. O'Reilly Media; https://wesmckinney.com/book/
J. VanderPlas	2022	Python Data Science Handbook	2nd ed. O'Reilly Media
Python Software Foundation	Online	The Python Tutorial and Python Standard Library	Current documentation https://docs.python.org/3/

Author / organisation	Year	Title / resource	Edition / publisher / URL
NumPy developers	Online	NumPy documentation: user guide and reference	Current documentation https://numpy.org/doc/stable/
RECOMMENDED READING			
SciPy developers	Online	SciPy documentation: user guide and API reference	Current documentation https://docs.scipy.org/doc/scipy/
pandas developers	Online	pandas user guide	Current documentation https://pandas.pydata.org/docs/
Matplotlib development team	Online	Matplotlib documentation and tutorials	Current documentation https://matplotlib.org/stable/
SymPy development team	Online	SymPy documentation	Current documentation https://docs.sympy.org/latest/

Additional scientific or domain-specific articles and datasets are provided during the semester as relevant to laboratory work and project topics.