Syllabus
Course Program

Python Advanced Programming Course

Specialty
121 – Software Engineering
122 – Computer Science

Institute
Institute of Computer Science and Information Technology

Educational program
Software Engineering
Computer Science and Intelligent Systems

Department
Software Engineering and Management Intelligent Technologies (321)

Level of education
Bachelor's level

Course type
Special (professional), Elective

Semester
5

Language of instruction
English, Ukrainian

Lecturers and course developers

Svitlana Kovalenko
Svitlana.Kovalenko@khpi.edu.ua
Candidate of Engineering Sciences, Associate Professor, Associate Professor of Software Engineering and Management Intelligent Technologies Department

Google Scholar:
https://scholar.google.com/citations?user=jeD1w74AAAAJ&hl
ORCID: https://orcid.org/0000-0001-6770-6778
Scopus: https://www.scopus.com/authid/detail.uri?authorId=57212035934
Web of Science: https://www.webofscience.com/wos/author/record/F-8252-2017)
More about the lecturer on the department's website

General information

Summary

The objective of the discipline is to further expand and deepen knowledge and skills in Python programming; expand understanding of programming concepts and develop skills in developing complex software solutions using a variety of technologies and tools, further acquisition of skills in software design, development and testing

Course objectives and goals

Developing students' theoretical and practical knowledge of working with files and folders, using third-party Python libraries for scientific calculations and data visualization; working with databases in Python; creating GUI applications using Tkinter libraries.

Format of classes

Lectures, laboratory classes, self-study, consultations. Final control in the form of a credit.

Competencies

121-К01. Ability to think abstractly, analyze and synthesize.
121-К02. Ability to apply knowledge in practical situations.
121-К05. Ability to learn and master modern knowledge.
121-К06. Ability to search, process and analyze information from various sources.
121-К13. Ability to identify, classify and formulate software requirements.
121-К15. Ability to develop architectures, modules and components of software systems.
121-К19. Knowledge of data information models, ability to create software for data storage, extraction and processing..
121-К22. The ability to accumulate, process and systematize professional knowledge in the creation and maintenance of software and recognize the importance of lifelong learning.
121-К23. Ability to implement phases and iterations of the life cycle of software systems and information technologies based on appropriate software development models and approaches.
121-К25. Ability to reasonably choose and master tools for software development and maintenance.
121-К26. Ability to think algorithmically and logically.
122- GC1. Ability to think abstractly, analyze and synthesis.
122- GC2. Ability to apply knowledge in practical situations.
122- GC6. Ability to learn and master modern knowledge.
122- GC7. Ability to search, process and analyze information from various sources.
122- GC8. Ability to generate new ideas (creativity).
122- GC11. Ability to make informed decisions.
122- PC8. Ability to design and develop software using various programming paradigms: generalised, object-oriented, functional, logical, with appropriate models, methods and algorithms of computation, data structures and control mechanisms.
122- PC9. Ability to implement a multi-level computing model based on client-server architecture, including databases, knowledge and data warehouses, to perform distributed processing of large data sets on clusters of standard servers to meet the computing needs of users, including cloud services.
122- PC10. Ability to apply methodologies, technologies and tools to manage the life cycle processes of information and software systems, information technology products and services in accordance with customer requirements.
122- PC12. Ability to ensure the organization of computing processes in information systems for various purposes, taking into account the architecture, configuration, performance indicators of operating systems and system software.

Learning outcomes

121- PLO01. Analyze, purposefully search for and select information and reference resources and knowledge necessary for solving professional problems, taking into account modern achievements of science and technology.
121- PLO02. Know the code of professional ethics, understand the social significance and cultural aspects of software engineering and adhere to them in professional activities.
121- PLO08. Be able to develop a human-machine interface.
121- PLO12. Apply effective software design approaches in practice.
121- PLO13. Know and apply methods of developing algorithms, designing software and data and knowledge structures.
121- PLO15. Motivated to choose programming languages and development technologies to solve the problems of creating and maintaining software.
121- PLO18. Know and be able to apply information technologies for data processing, storage and transmission.
121- PLO23. Be able to document and present the results of software development.
122- PLO1. To apply knowledge of the basic forms and laws of abstract and logical thinking, the basics of the methodology of scientific knowledge, forms and methods of extracting, analyzing, processing and synthesizing information in the subject area of computer science.
122- PLO4. To use methods of computational intelligence, machine learning, neural network and fuzzy data processing, genetic and evolutionary programming to solve problems of recognition, forecasting, classification, identification of control objects, etc.

122- PLO10. To use tools for developing client-server applications, design conceptual, logical and physical models of databases, develop and optimize queries to them, create distributed databases, data warehouses and showcases, knowledge bases, including cloud services, using web programming languages.
122- PLO11. Have the skills to manage the life cycle of software, products and services of information technology in accordance with the requirements and restrictions of the customer, be able to develop project documentation (feasibility study, terms of reference, business plan, agreement, contract).
122- PLO20. Develop the architecture of software systems and their individual components in the construction of intelligent control systems in various industries, as well as manage the life cycle processes of software of intelligent control systems.

Student workload

The total volume of the course is 150 hours (5 ECTS credits): lectures - 32 hours, laboratory classes - 32 hours, self-study - 86 hours.

Course prerequisites

The study of this discipline is directly based on: "Object-oriented programming", "Basic course of Python programming"

Features of the course, teaching and learning methods, and technologies

Teaching and learning methods:
interactive lectures with presentations, discussions, laboratory classes, teamwork, case method, student feedback, problem-based learning.

Forms of assessment:
written individual assignments for laboratory work (CAS), assessment of knowledge in laboratory classes (CAS), express surveys (CAS), online tests (CAS), final/semester control in the form of a semester exam, according to the schedule of the educational process (FAS).

Program of the course

Topics of the lectures

Topic 1. Python File Input/Output
The difference between binary and text files. Opening and closing text files. Content manager with. Modes of opening files. Reading, writing, and modifying data in files.
Topic 2. Working with files and folders
Modules os, pathlib, glob. Getting the content of a folder. Getting file attributes. Creating and deleting folders. Searching for files by mask. Copying, moving and renaming files and folders. The zipfile module and working with archives.
Topic 3. Package manager and Jupyter Notebook
Standard library modules and third-party modules. Pip and its most commonly used commands. Jupyter Notebook: installation, debugging, and usage. Google Collaboratory. How to share Jupyter Notebook. Git and GitHub: installation, basics, creating a remote repository, rendering and distributing notebooks.
Topic 4. Date and time handling in Python
The import command import and its uses. Built-in modules datetime and time. The concepts of timestamp and timedelta. Operations with dates and times. Methods of the Date(), Time() and Datetime() classes.
Тема 5. Formatting Python code. PEP8. Linters and formatters
PEP8 and rules for formatting Python code to improve readability. Module black. Liners. Installing and using formatters and linters in different IDEs.
Topic 6. Python decorators
The concept of first class objects. Calling a function from another function. Closure of a function. Internal and external functions. Creating a decorator. *args, **kwargs. Creating and using class decorators.
Topic 7. Python for scientific calculations
The Numpy module and the history of its creation. ndarrays and their difference from Python lists. Standard Numpy data types. Attributes of ndarrays arrays. Creation and operations on arrays. Indexing,

slicing and advanced indexing. Subarrays as a display and creating copies of arrays. Changing the shape of arrays: the reshape() and resize() methods. Calculations on ndarrays and universal functions. Calculation time, magic commands %timeit and %time. Aggregation functions. Broadcasting of ndarrays. Comparisons on arrays, masks and Boolean logic.

Topic 8. Introduction to visualization in Python

The Matplotlib module. Styles. Creating graphs in Jupyter Notebook and other IDEs. Saving a graph to a file, types of graph files. Procedural and object-oriented interfaces of Matplotlib. The Figure() and Axes() classes. Simple line graph: creating and debugging. Scatter plots: features, creation and setting. Seaborn and plotly libraries.

Topic 9. Creating a GUI with TkInter

The concept of GUI and visual programming. Classical and event-driven paradigm. Tk and TkInter. Geometry managers. Anchors. Setting window parameters. Basic TkInter widgets and working with them. Events and their processing. TkInter variables. Creating a menu. Dialog boxes. Pyinstaller and creating an .exe file. Creating batch files to run GUI applications.

Topic 10. Using databases in Python

Python and database support. Libraries for working with databases. SQLite and its main characteristics. Connecting to the database. Creating tables. Adding data to the table. Getting data, fetchall(). Filtering data. Deleting and updating data.

Topic 11. Software testing

The basics of software testing. Automatic and manual testing. Unit and integration testing. Libraries for testing. The structure of the test. Testing with pytest. Evaluation methods in unittest. Running tests. Understanding test results.

Topics of the workshops

Workshops are not provided within the discipline.

Topics of the laboratory classes

Topic 1: Working with files and folders
Topic 2. Exploring the Jupyter Notebook environment
Topic 3: Using the Numpy library
Topic 4. Visualization in Python
Topic 5. Creating a GUI application using TkInter
Topic 6: Software testing

Self-study

Students are recommended additional materials (videos, articles) for self-study and processing.

Course materials and recommended reading

Key literature

1. Mariano Anaya. Clean Code in Python: Develop maintainable and efficient code, 2nd Edition, Packt Publishing, 2021, 422p.
2. Robert Johansson. Numerical Python. Scientific Computing and Data. Science Applications with Numpy, SciPy and Matplotlib. Second Edition, 2019. 685 p.
3. Eric Matthes Python Crash Course, 2nd Edition, No Starch Press, 2019, 548p.
4. John Grayson. . Python and Tkinter Programming, Manning, 2020, 688p.
5. Agus Kurniawan. Python and SQLite Development, PE Press. 2021, 54p.

Additional literature

1. Dusty Phillips, Steven F. Lott Python Object-Oriented Programming, 4th edition, Packt, 2019, 714p
2. Luciano Ramalho. Fluent Python, O'Reilly Media, 2022, 980p
3. Al Sweigart. Automate the Boring Stuff with Python, 2nd edition: Practical Programming for Total Beginners, No Starch Press, 2020, 901p.

4. B. Stephenson: The Python Workbook: A Brief Introduction with Exercises and Solutions, 2 nd ed. // Springer, 2019
5. John Canning, Alan Broder, Robert Lafore. Data Structures & Algorithms in Python, Pearson Education

Assessment and grading

Criteria for assessment of student performance, and the final score structure

100% of the final grade consists of the results of the assessment in the form of a credit(40%) and current assessment (60%):
- 6 laboratory works (7% each);
- 2 tests (9% each);

Grading scale

Total points	National	ECTS
90–100	Excellent	A
82–89	Good	B
75–81	Good	C
64–74	Satisfactory	D
60–63	Satisfactory	E
35–59	Unsatisfactory (requires additional learning)	FX
1–34	Unsatisfactory (requires repetition of the course)	F

Norms of academic integrity and course policy

The student must adhere to the Code of Ethics of Academic Relations and Integrity of NTU "KhPI": to demonstrate discipline, good manners, kindness, honesty, and responsibility. Conflict situations should be openly discussed in academic groups with a lecturer, and if it is impossible to resolve the conflict, they should be brought to the attention of the Institute's management.
Regulatory and legal documents related to the implementation of the principles of academic integrity at NTU "KhPI" are available on the website: http://blogs.kpi.kharkov.ua/v2/nv/akademichna-dobrochesnist/

Approval

Approved by	08.06.2023	Head of the department Ihor HAMAIUN
	08.06.2023	Guarantors of the educational programs Andrii KOPP Uliya LITVINOVA