FUNDAMENTALS OF PYTHON PROGRAMMING

COURSE SYLLABUS

Code and name of specialty	121 Software Engineering , 122 Computer science , 126 Information Systems and Technologies	**Institute / faculty**	Faculty of Computer Science and Software Engineering
Program name	Software Engineering Computer Science and Intelligent Systems Information Systems Software	**Department**	Software Engineering and Management Information Technologies
Type of program	Educational and Professional	**Language of instruction**	Ukrainian, English

Lecturer

Full name, *e-mail*	Kozulia Mariia, mariia.kozulia@khpi.edu.ua

PhD, Candidate of Engineering Sciences (21.06.01 - ecology safety), Associate Professor of Department of Software Engineering and Information Technology Management. Work experience – since 2016. Author (co-author) of more than 75 scientific and educational publications (h-index= 6, i10-index= 2 in Google Scholar - https://scholar.google.ru/citations?user=tRyBDzQAAAAJ&hl=ru; ORCID https://orcid.org/0000-0002-4090-8481). Main courses: «Fundamentals of Python programing» (lectures and laboratory classes), «Advanced course of Python programing» (lectures and laboratory classes), «Decision making theory» (lectures and laboratory classes in English), «Models and methods of decision support systems» (lectures and laboratory classes in English), «Green computing» (lectures and laboratory classes in English).

GENERAL DESCRIPTION OF THE COURSE

Summary	The course "Fundamentals of Python Programming" is a discipline in the cycle of selective training in the specialty 121 "Software Engineering", 122 Computer science , 126 Information Systems and Technologies. It is taught in the third semester in the amount of 180 hours (6 ECTS credits), in particular: lectures - 32 hours, laboratory classes - 32 hours, independent work - 116 hours. There are no individual tasks. The study of the discipline ends with a credit.
Course objectives	Formation of students' theoretical and practical knowledge of the basics of programming in Python, mastering the methods of data collection and processing.
Types of classes and control	Lectures, laboratory classes. Current control - laboratory work, intermediate modular control. Final control – credit.
Term	3

Student workload (credits) / Type of course	6/ elective	**Lectures (hours)**	32	**Workshops (hours)**	32	**Self-study (hours)**	116

(mandatory / elective)							

Program competences	121-GC 01. Ability to abstract thinking, analysis and synthesis. 121-GC 02. Ability to apply knowledge in practical situations. 121-GC 05. Ability to learn and master modern knowledge. 121-GC 06. Ability to search, process and analyze information from various sources. 121-PC15. Ability to develop architectures, modules and components of software systems. 121-PC16. Ability to formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards. 121-PC19. Knowledge of information data models, the ability to create software for data storage, retrieval and processing. 121-PC26. Ability to algorithmic and logical thinking. 122-GC1. Ability to abstract thinking, analysis 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-GC12. Ability to evaluate and ensure the quality of performed work. 122-PC3. Ability to think logically, build logical conclusions, use formal languages and models of algorithmic calculations, design, develop and analyze algorithms, evaluate their efficiency and complexity, solvability and unsolvability of algorithmic problems for adequate modelling of subject areas and creation of software and information systems. 122-PC8. Ability to design and develop software using different programming paradigms: generalized, object-oriented, functional, logical, with appropriate models, methods and algorithms of calculations, data structures and management mechanisms. 126-GC 1. Ability to abstract thinking, analysis and synthesis. 126-GC 2. Ability to apply knowledge in practical situations. 126-GC 5. Ability to learn and master modern knowledge. 126-GC 6. Ability to search, process and summarize information from various sources. 126-GC 8. Ability to evaluate and ensure the quality of work performed. 126-PC 4. Ability to design, develop and use tools for the implementation of information systems, technologies and infocommunications (methodological, informational, algorithmic, technical, software and others). 126-PC 6. Ability to use modern information systems and technologies (production, decision support, data mining, etc.), cybersecurity techniques and techniques in the performance of functional tasks and responsibilities. 126-PC 8. Ability to manage the quality of products and services of information systems and technologies during their life cycle.

Learning outcomes	**Teaching and learning methods**	**Forms of assessment (continuous assessment CAS, final assessment FAS)**
121-PO01. Analyze, purposefully search for and select the necessary information and reference resources and knowledge to solve professional problems, taking into account modern advances in science and technology. 121-PO07. Know and apply in practice the fundamental concepts, paradigms and basic principles of operation of language, tools and computing software engineering. 121-PO11. Choose source data for design, guided by formal methods of describing requirements and	Mini-lecture "Introduction to Python. Data types. Data input and output » Mini-lecture "Conditional construction if..else, if… elif… else. Python Math Module » Problem lectures. "Anonymous functions, lambda instructions. Generator functions. Code Documentation » Problem lecture. "Encapsulation. Getter. Setters. Inheritance. Association. Weak links. " Problem lecture. "Polymorphism. Iterators. Metaclass. Multimedia methods. "	**Current CAS assessment:** Assessment of students' work in the laboratory Intermediate modular control **Final FAS assessment:** Credit

modelling. 121-PO12. Put effective approaches to software design into practice. 121-PO13. Know and apply methods of algorithm development, software design and data and knowledge structures. 121-PO15. Being motivated to choose programming languages and development technologies to solve problems of software design and maintenance. 121-PO17. Be able to apply methods of component software development. 122-PLO1. Apply knowledge of the fundamental forms and laws of abstract-logical thinking, the basics of the methodology of scientific knowledge, forms and methods of extraction, analysis, processing, and synthesis of information in the subject area of computer science. 122-PLO9. Develop software models of subject areas, choose a programming paradigm from the standpoint of convenience and quality of its application to implement methods and algorithms that solve problems in the computer science field. 122-PLO11. Have the skills to manage the life cycle of software, products, and services of information technology under the requirements and restrictions of the customer, be able to develop project documentation (feasibility study, technical task, business plan, agreement, contract). 122-PLO13. Know the system programming languages and methods for the software development that interacts with the components of computer systems, know network technologies, computer network architectures, have practical skills in administration technology of computer networks and their software. 122-PLO20. Develop the architecture of software systems and their particular components during the construction of intelligent management systems in various fields, as well as manage the life cycle of intelligent management systems software. 126-PLO 3. To use basic knowledge of informatics and modern information systems and technologies, programming skills, technologies of safe work in computer networks, methods of creation of databases and Internet resources, technologies of development of algorithms and computer programs in high-level languages with application of project-oriented programming to solve problems of design and use of	Problem lecture. “Threads. Working with threads. Multithreading. Demons. ” Problem lecture. "Race conditions. Thread sync. "	

information systems and technologies. 126-PLO 4. Conduct a systematic analysis of design objects and justify the choice of structure, algorithms and methods of information transfer in information systems and technologies. 126-PLO 5. Argue the choice of software and hardware for the creation of information systems and technologies based on the analysis of their properties, purpose and technical characteristics, taking into account the requirements for the system and operating conditions; have the skills to debug and test software and hardware of information systems and technologies. 126-PLO 6. Demonstrate knowledge of the current level of information systems technology, practical skills of programming and use of applied and specialized computer systems and environments for their implementation in professional activities. 126-PLO 7. Justify the choice of technical structure and develop appropriate software that is part of information systems and technologies.		

ASSESSMENT AND GRADING

Ranges of points corresponding to grades	Total score (points) for all types of learning activities	ECTS grading scale	The national grading scale	Allocation of grade points	
	90-100	A	Excellent		**100% final assessment** in the form of credit **(30%)** and current assessment **(70%).** **30%** credit **70%** current rating: Module №1 (10%) Module №2 (20%) Laboratory work (40%)
	82-89	B	Good		
	74-81	C			
	64-73	D	Satisfactory		
	60-63	E			
	35-59	FX	Unsatisfactory (with the exam retake option)		
	0-34	F	Unsatisfactory (with mandatory repetition of the course)		

Course policy	The student is required to attend all classes according to the curriculum and adhere to the norms of academic ethics. To study the discipline you need to have your own personal computer and / or use the computers of the computer center of the department. The student must work with required and additional literature, including information resources on the Internet. All laboratory work must be completed and submitted by the student during the semester in which the discipline is taught, before the start of the test week. Without the personal presence of the student the final control is not carried out.

COURSE STRUCTURE AND CONTENT

Lecture 1	Introduction to Python. Data types. Data input and output	**Laboratory work 1**	Getting started with Python. PT50	**du al**	Dynamic work with data type.

Lecture 2	Conditional construction if..else, if… elif… else. Python's Math module.	**Laboratory work 2**	Working with conditional operators, conditional construction if..else, if… elif… else. Python's Math module. PT50		Exceptions, features of work and area of their use.
Lecture 3	List, String, tuple, range and work with them. Loop for, while. List compression.	**Laboratory work 3**	Working with data: list, string, tuple, range, sets, dictionaries and work with them. PT50		Features of creating metaclasses and their use.
Lecture 4	Sets, dictionaries and work with them.	**Laboratory work 4**	Working with loops. PT50		Threads. Scope and features of use.
Lecture 5	Scope of variables. Functions. Anonymous functions, lambda instructions. Generator functions. Recursion. Code documentation.	**Laboratory work 5**	Working with functions, exceptions and classes. PT51		Integration and system testing tools.
Lecture 6	Exceptions. Debugging with assert instructions.	**Laboratory work 6**	Work with built-in modules. PT51		
Lecture 7	Modules: creation and further use. Work with time. Working with the date.	**Laboratory work 7**	OOP (Object-Oriented programming). PT52		
Lecture 8	Packages: import, creation, further use. Overview of standard libraries. Regular expression templates.	**Laboratory work 8**	Working with files. PT52		
Lecture 9	Introduction to object-oriented programming. Creating a class and working with it. Magic Methods.	**Laboratory work 9**	Work with threads. Testing. PT52		
Lecture 10	Decorators. Encapsulation. Getters. Setters. Inheritance. Association. Weak links.				
Lecture 11	Polymorphism. Iterators. Metaclass. Multimedia methods.				
Lecture 12	Files. Working with files. Copying files.				
Lecture 13	Threads. Working with threads. Multithreading. Demons.				
Lecture 14 **Lecture 15**	Race conditions. Thread synchronization. Asynchrony.				
Lecture 16	Writing tests				
RECOMMENDED READING					

Compulsory	1. John Shovic & Alan Simpson. (2019). Python All-in-One. For Dummies, 2019. 720 p. 2. Bhasin, H. (2019). Python Basics: A Self-Teaching Introduction. Mercury Learning and Information, 450 p. 3. Irv Kalb, (2018). Learn to Program with Python 3: A Step-by-Step Guide to Programming. 2nd ed. Edition. Apress, 371 p. 4. Brian Okken, (2017). Python Testing with pytest: Simple, Rapid, Effective, and Scalable, 256 p. 5. Mark Lutz. (2017). Learning Python. O'Reilly Media: 5th Edition, 1648 p. 6. Python Tricks and Tips Magazine: 5th Edition. 2021: Gain Insider Skills Kindle Edition. Mehedi Hzn Press Publications, 164 p.	**Recommended**	7. Lamy, Jean .(2021). Baptiste Ontologies with Python. Apress. 8. Reuven, M. (2020). Lerner Python Workout 50 ten-minute exercises. Manning, 249 p. 9. Robert, C. (2020). Matthews Coding in Python: A Comprehensive Beginners Guide to Learn the Realms of Coding in Python. Independently published, 211 p. 10. Bill Lubanovic. (2020). Introducing Python : Modern Computing in Simple Packages. O'Reilly Media, Inc, USA, 500 p. **INTERNET INFORMATION RESOURCES** 1. Beginner's Guide to Python. [Electronic resource]. Access mode: https://wiki.python.org/moin/BeginnersGuide 2. The Python Standard Library. [Electronic resource]. Access mode: https://docs.python.org/3/library/index.html

Academic integrity

The student must adhere to the Code of Ethics of Academic Relations and Integrity of NTU "KhPI": to show discipline, politeness, friendliness, honesty, responsibility

The content of this syllabus is consistent with the course program.