Computer Mathematics (Part 3)

COURSE SYLLABUS

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

LECTURER

Name and surname, email **Nataliia Khatsko, *nataliia.khatsko@khpi.edu.ua***

Ph.D., associate professor at the Department of Software Engineering and Management Information Technologies of NTU «KhPI», associate professor. Prepared and published more than 40 research papers and textbooks (SCOPUS Author ID https://www.scopus.com/authid/detail.uri?authorId=57200820629; Researcher ID https://app.webofknowledge.com/author/#/record/17252627; GoogleScholar https://scholar.google.com.ua/citations?user=US7Ovx4AAAAJ&hl=uk; ORCID orcid.org/0000-0002-2543-0280)

Basic courses: "Computer Mathematics (parts 1, 2, 3)", "Practical seminar on mathematical methods in software engineering", "Formal methods of software verification", "Formal methods of software systems research".

GENERAL DESCRIPTION OF THE COURSE

Summary	The course " Computer Mathematics (Part 3) " is a discipline in the cycle of professional training in the specialty 121 "Software Engineering". It is taught in the fifth semester in the amount of 180 hours (6 ECTS credits), in particular: lectures - 32 hours, practical classes - 32 hours, independent work - 116 hours. The course includes two content modules and two modular tests. The discipline ends with an exam.
Course objectives	Formation of students' theoretical and practical knowledge about the theory of finite automata and its accompanying concepts (grammar, language, regular expression). Formation of students' modern system of views in the field of computer discrete mathematics, acquisition of practical skills in the use of formal methods and models of discrete mathematics in the processing of discrete information and description of discrete processes associated with software development.
Types of classes and control	Lectures, practical classes, self-study. Final control - exam.
Term	5

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

Program competences	GC01. Ability to abstract thinking, analysis and synthesis. GC05. Ability to learn and master modern knowledge. GC06. Ability to search, process and analyze information from various sources. PC26. Ability to algorithmic and logical thinking.

Learning outcomes	**Teaching and learning methods**	**Forms of assessment**

		(continuous assessment CAS, final assessment FAS)
PLO01. 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. PLO05. Know and apply relevant mathematical concepts, methods of domain, system and object-oriented analysis and mathematical modelling for software development.	lectures, practical classes, home tasks, self-work with literary sources.	**Continuous assessment CAS**: Student's assessment during practical classes. Tests **Final assessment FAS**: Exam

ASSESSMENT AND GRADING

	core (points) for all types of learning activities	ECTS grading scale	The national grading scale		
Ranges of points corresponding to grades	90-100	A	excellent	**Allocation of grade points**	**100% final evaluation** in the form of defense of the course project (40%) and current evaluation (60%). **40% exam:…** **60% current rating:**
	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	Students must attend all classes according to the study schedule and adhere to the norms of academic ethics. Students must work with compulsory and recommended reading, including Internet resources. Students must complete and submit all practice works (workshops) during the semester in which the course is taught, before the examination session. The final assessment is not carried out without the personal presence of students.

COURSE STRUCTURE AND CONTENT

Lecture 1-2	Introduction to formal proof	**Practical lessons 1-2**	Deductive proofs. Construction of proofs.	**Self-study**	Studying the material with the help of literature, homework.
Lecture 3-4	Alphabets, grammars and languages	**Practical lessons 3-4**	Solving problems of output of chains according to the rules of output, analysis of chains		
Lecture 5	Regular expressions, languages.	**Practical lesson 5**	Modular control work.		

Lecture 6-7	Deterministic finite automata	**Practical lessons 6-7**	Problem solving. Description of deterministic finite automata.		
Lecture 8-9	Nondeterminate finite automata	**Practical lessons 8-9**	Problem solving. Description of nondeterministic finite automata, and their transformation.		
Lecture 10-11	Finite state machines with epsilon transitions	**Practical lessons 10-11**	Conversion of automata.		
Lecture 12-13	Finite automata and regular expressions	**Practical lessons 12-13**	Conversion of a deterministic finite automaton into a regular expression.		
Lecture 14-15	Convert a regular expression to a finite automaton	**Practical lessons 14-15**	Convert a regular expression to a finite automaton.		
Lecture 16	Pushdown Automata	**Practical lesson 16**	Modular control work.		

RECOMMENDED READING

Compulsory	1. Ajit Singh Formal Language And Automata Theory. (2019). LAMBERT Academic Publishing. 2. Abejide Ade-Ibijola New Finite Automata Applications in Novice Program Comprehension. (2017). LAP LAMBERT Academic Publishing. 3. Neeru Gupta (2020). Beginner's Guide-Automata Theory. 4. Ezhilarasu Umadevi Palani. (20190. Finite Automata Problems & Solutions- LAP Lambert Academic Publishing. 5. Stoyan Mihov, Klaus U. Schulz. (2019). Finite-State Techniques: Automata, Transducers and Bimachines. Cambridge University Press. 6. Hopcroft, John E., Motwani, Rajeev, Ullman, Jeffrey D. (2014). Introduction to automata theory, languages, and computation. Boston: Pearson Education, Inc. 7. Хаггарти, Р. (2017). Дискретная математика для программистов: учебное пособие. Москва: Техносфера.	**Recommended**	1. Michael Sipser. (2006). Introduction to the Theory of Computation. Thomson. 2. Daniel, I. A., Cohen, John Wiley. (1996). Introduction to Computer Theory. 3. John C Martin. Introduction to languages and the Theory of Computation. 4. Lewis H. P., Papadimition, C. H. Elements of Theory of Computation. 5. Mishra, Chandrashekaran. Theory of Computer Science and Automata languages and computation. (2nd ed.). 6. Foster, E. C. Software Engineering : A Methodical Approach. New York: Apress. 7. J. Richard Büchi. (1989). Finite Automata, Their Algebras and Grammars. Towards a Theory of Formal Expressions. 8. Goswami, D., Krishna, K. V. (2010). Formal Languages and Automata Theory. 9. Завалишин, Е. П. (2007). Логика. Учебное пособие для вузов. Тула: Изд-во ТулГУ. 10. Gerda Ivanickienė. The theoretical material and exercises. INFORMATION RESOURCES ON THE INTERNET 1. D. Goswami and K. V. Krishna. Formal Languages and Automata Theory. Retrieved from http://www.iitg.ernet.in/dgoswami/Flat-Notes.pdf 2. Introduction to Automata and Complexity Theory. Retrieved from http://infolab.stanford.edu/~ullman/ialc/spr10/spr10.html 3. Formal Languages and Automata Theory. Retrieved from http://cs.fit.edu/~dmitra/FormaLang.

Academic integrity

Graduate students are expected to adhere to the Code of Ethics of Academic Relations and Integrity" of NTU "KhPI".

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