# Algorithmization and Programming

COURSE SYLLABUS

| | | | |
|---|---|---|---|
| **Code and name of specialty** | 122 – Computer science | **Institute** | **Computer Sciences and Software Engineering** |
| **Program name** | Computer Science and Intelligent Systems | **Department** | Software Engineering and Management Information Technologies |
| **Type of program** | **Educational and Professional** | **Language of instruction** | **Ukrainian** |

## LECTURER

**Bilova Mariia Oleksyivna** Mariia.Bilova@khpi.edu.eu

**PhD, Candidate of Engineering Sciences, Associate Professor of Department of Software Engineering and Information Technology Management. Number of scientific and educational publications more than 50 (h-index = 3, i10-index = 0 in Google Scholar - https://scholar.google.com/citations?user=b3YLGToAAAAJ; ORCID ID - https://orcid.org/0000-0001-7002-4698, Scopus ID https://www.scopus.com/authid/detail.uri?authorId=57190442390).**
**Leading lecturer of the courses: Algorithmization and Programming** *(Bachelors) (Ukrainian)*

## GENERAL DESCRIPTION OF THE COURSE

| | |
|---|---|
| **Summary** | The course "Algorithmization and Programming" is a course in the cycle of professional compulsory training of the specialty 126 "Information Systems and Technologies". It is taught in the first and second semesters in the amount of 300 hours (10 ECTS credits), in particular: lectures – 80 hours, laboratory classes – 80 hours, independent work – 140 hours. The course includes four content modules and five tests. The study of the discipline ends with the exam. |
| **Course objectives** | Algorithmic thinking formation and skills acquisition of programs development using C and Python programming languages for applied problems solving from various subject areas according to modern principles and tendencies of construction of the application software. |
| **Types of classes and control** | Lectures, laboratory works, control works, self-study. The course ends with a final exam |
| **Term** | 1, 2 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Student workload (credits) / Type of course** | 4 / Mandatory | **Lectures (hours)** | 80 | **Workshops (hours)** | 80 | **Self-study (hours)** | 140 |

| | |
|---|---|
| **Program competences** | GC1. Ability to abstract thinking, analysis and synthesis.<br>GC2. Ability to apply knowledge in practical situations.<br>GC3. Knowledge and understanding of the subject area and understanding of professional activity.<br>GC6. Ability to learn and master modern knowledge.<br>GC7. Ability to search, process and analyze information from various sources.<br>GC9. Ability to work in team.<br>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.<br>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. |

| Learning outcomes | Teaching and learning methods | Forms of assessment (continuous assessment CAS, final assessment FAS) |
|---|---|---|
| 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. | Interactive lectures with presentations, discussions, practical classes, teamwork, case method, method of feedback from students, problem-based learning | Written individual assignments for laboratory work (CAS), assessment of knowledge in laboratory classes (CAS), collection of data on individual assignments and reporting on research results (CAS), final / semester control in the form of a semester exam, according to the learning process schedule (FAS) |
| PLO5. Design, develop and analyze algorithms for solving computational and logical problems, evaluate the efficiency and complexity of algorithms based on the use of formal models of algorithms and computational functions. | Interactive lectures with presentations, discussions, practical classes, teamwork, case method, research, project training | Written individual assignments for laboratory work (CAS), assessment of knowledge in laboratory classes (CAS), collection of data on individual assignments and reporting on research results (CAS), final / semester control in the form of a semester exam, according to the learning process schedule (FAS) |
| 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. | Interactive lectures with presentations, discussions, practical classes, teamwork, case method, research, problem-based learning | Written individual assignments for laboratory work (CAS), assessment of knowledge in laboratory classes (CAS), collection of data on individual assignments and reporting on research results (CAS), final / semester control in the form of a semester exam, according to the learning process schedule (FAS) |

## ASSESSMENT AND GRADING

| points corresponding to | Total score (points) for all types of learning activities | ECTS grading scale | The national grading scale | Allocation of grade points | |
|---|---|---|---|---|---|
| | | | | | 100% **final assessment** in the form of exam (30%) and current assessment (70%). |

| | | | | | |
|---|---|---|---|---|---|
| | 90-100 | A | excellent | | 30% **exam**: semester exam, according to the schedule of the educational process<br>70% **continuous assessment**:<br>**- semester 1:**<br>• 45% assessment of tasks in laboratory work;<br>• 25% intermediate control (control tasks and individual tasks)<br>**- semester 2:**<br>• 30% assessment of tasks in laboratory work;<br>• 40% intermediate control (control tasks and individual tasks) |
| | 82-89 | B | good | | |
| | 74-81 | C | | | |
| | 64-73 | D | satisfactory | | |
| | 60-63 | | | | |
| | 35-59 | FX | Unsatisfactory (with the exam retake option) | | |
| | 0-34 | F | Unsatisfactory (with mandatory repetition of the course) | | |

| | |
|---|---|
| **Course policy** | Follow the rules of the University internal regulations. Take an active part in the learning process. Students must attend all classes according to the study schedule and adhere to the norms of academic ethics. To study the course, students need to have their personal computer and (or) use computers of the computer center at the department. Students must work with compulsory and recommended reading, including Internet resources. Students must complete and submit all laboratory works 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

| | | | | | |
|---|---|---|---|---|---|
| **Topic 1** | Algorithms and features of their development | **Laboratory work** 0 | Scratch Project | **Self-Study** | Development of the activity diagram according to the task |
| **Topic 2** | Numeral systems | **Laboratory work** 1 | Basic syntactic constructions of the C programming language | | Transition between numeral systems |
| **Topic 3** | Data types and variables in C programming language | **Laboratory work** 2 | Arrays and functions | | Implementation of the program based on the previously developed activity diagram |
| **Topic 4** | Conditional operators | **Laboratory work** 3 | Algorithms | | Implementation of array sorting by two algorithms of the teacher's choice |

| | | | | | |
|---|---|---|---|---|---|
| **Topic 5** | Loops | **Laboratory work 4** | Work with memory | | Implementation of the structure in according to the task |
| **Topic 6** | Analysis of program performance | **Laboratory work 5** | Data structures | | |
| **Topic 7** | Arrays | | | | |
| **Topic 8** | Functions | | | | |
| **Topic 9** | Sorting | | | | |
| **Topic 10** | Features of working with memory | | | | |
| **Topic 11** | Structures | | | | |

**RECOMMENDED READING**

**Compulsory**


1. Kornienko M.M., Ivanova I.D. (2011) Informatics. Fundamentals of algorithmization and programming. Ranok,
2 Cormen T. H., Thomas H., Cormen Charles E., Leiserson Ronald L. (2009)Introduction to Algorithms Third Edition Rivest Clifford Stein The MIT Press
3. Stephens R. (2013) Essential Algorithms: A Practical Approach to Computer Algorithms. John Wiley & Sons, Inc
4. C Programming Absolute Beginner's Guide. (2014) Third Edition. Pearson Education.
5. Seacord R. C. (2020) Effective C: An Introduction to Professional C Programming. No Starch Press.
6. Thomas Mailund Pointers in C Programming. A Modern Approach to Memory Management, Recursive Data Structures, Strings, and Arrays. Apress,
7. Anquetil R (2019) Fundamental Concepts for Web Development: HTML5, CSS3, JavaScript and much more. Independently published.
8. Myers M. (2017) Smarter way to learn Python
9. Stephenson B. (2019) The Python Workbook. Springer: Texts in Computer Science.
10. Matthes E. (2021) Python Crash Course. Old Lion Publishing House.
11. Rudenko V.D., Zhugastrov O.O. (2019) Basics of algorithmization and programming in Python. – Ranok.
12. Allen G. (2019) Taylor Author of SQL All-in-One For Dummies. 9th edition. Hoboken
13. Upadhyay K. Ch. (2020) HTML5 For Web Designers. Complete Hypertext Markup Language Guidance. - Independently published
14. Grant K. J.(2018) CSS in depth. Manning Publications Co.
15. Grinberg M. (2018) Flask Web Development: developing web applications with Python. 2nd edition. - O'Reilly Media, Inc.


**Recommended**


16. Skiena S. S. (2020) The Algorithm Design Manual. Third edition. Springer, Texts in Computer Science,–
17. Sweigart A. l. (2017) Invent Your Own Computer Games with Python, 4th edition. No Starch Press.
18. Martin R. (2019) Clean code.
19. Chacon S., Scott Chacon, Ben Straub. (2014) Pro Git Apress
20. The GNU C Reference Manual [Electronic resource]. Access mode: Retryeved from: https://www.gnu.org/software/gnu-c-manual/gnu-c-manual.html
21. Beej's Guide to C Programming [Electronic resource]. – Access mode: https://beej.us/guide/bgc/html//index.html
22. CS50 «Introduction to Computer Science» on Prometheus [Electronic resource]. – Access mode : https://edx.prometheus.org.ua/courses/Prometheus/CS50/2016_T1/info
23. CS50 [Electronic resource]. – Access mode : https://cs50.harvard.edu/
24. C Programming Language Documentation. [Electronic resource]. – Access mode: https://devdocs.io/c/
25. Front-End Developer Handbook 2018 / Cody Lindley – Frontend Masters. –2018. – 168 p. [Electronic resource]. – Access mode : https://legacy.gitbook.com/book/frontendmasters/front-end-developer-handbook-2018/details.
26. SQL Tutorial [Electronic resource]. – Access mode : https://www.w3schools.com/sql/
27. CSS Snapshot 2017. W3C Working Group Note [Electronic resource]. – Access mode : https://www.w3.org/TR/css-2017/.
28. HTML 5.2. W3C Recommendation [Electronic resource]. – Access mode : https://www.w3.org/TR/html52/.
29. UML [Electronic resource]. – Access mode : https://www.uml.org/
30. Fundamentals of UML [Electronic resource]. – Access mode : https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-basics.html

## ACADEMIC INTEGRITY

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.