Object-Oriented Programming (introductory practice)

COURSE SYLLABUS

Code and name of specialty	122 – Computer Science	**Institute**	**Computer Sciences and Software Engineering**
Program name	Computer Science and intelligence systems	**Department**	Software Engineering and Management Information Technologies
Type of program	**Educational and Professional**	**Language of instruction**	**Ukrainian**

LECTURER

Nikulina Olena Mykolayivna Olena.Nikulina@khpi.edu.eu

Doctor of Technical Sciences, Associate Professor, Professor of Software Engineering and Management Information Technologies Department. Number of scientific and educational publications – 90. (h-index = 5, i10-index = 1 in Google Scholar - https://scholar.google.com/citations?hl=en&user=ZEe2GlcAAAAJ&view_op=list_works&sortby=title; ORCID ID-https://orcid.org/0000-0003-2938-4215, Scopus ID-57203114988).

Leading lecturer of the courses: *Object-Oriented Programming (Bachelors) (Ukrainian), Numerical Methods (Bachelors) (Ukrainian), Operations Research (Bachelors) (Ukrainian), Intelligent Control Systems (Bachelors), Distributed Computing Models and Software (PhD) (Ukrainian)*

GENERAL DESCRIPTION OF THE COURSE

Summary	The course “Object-Oriented Programming” is a course in the cycle of professional compulsory training of the specialty 122 “Computer Science”. It is taught in the fourth semester in the amount of 120 hours (4 ECTS credits), in particular: lectures – 32 hours, laboratory classes – 32 hours, independent work – 56 hours. The course includes three content modules and three tests. The study of the discipline ends with the exam. The discipline is interrelated with such disciplines as "Introduction to the profession" and "Algorithmization and programming"
Course objectives	Learn the basics of software design; to study the technologies of object-oriented programming; learn techniques for working with visual programming environments; acquisition of skills of development and testing of software products operating under the control of modern operating systems; formation of students' abstract thinking, which should help solve applied problems related to various fields of knowledge. Assimilation of the necessary knowledge for mastering modern technologies of object-oriented analysis, design and programming of object-oriented model in different programming languages
Types of classes and control	Lectures, Laboratory work, control works, consultations. The course ends with a final exam
Term	3

Student workload (credits) / Type of course	4 / Mandatory	**Lectures (hours)**	32	**Workshops (hours)**	32	**Self-study (hours)**	56

Program competences	GC1. Ability to abstract thinking, analysis and synthesis. GC2. Ability to apply knowledge in practical situations. GC3. Knowledge and understanding of the subject area and understanding of professional activity.

	GC6. Ability to learn and master modern knowledge. GC9. Ability to work in team 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)
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, project training	
PLO19. Create intelligent management systems using methods of mathematical modeling and analysis of complex systems, methods of modeling and analysis of business processes, information technologies for the management of business systems.	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)

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
90-100	A	excellent
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)

Allocation of grade points

100% **final assessment** in the form of exam (20%) and current assessment (80%).
20% **exam**: semester exam, according to the schedule of the educational process
80% **continuous assessment**:
- 30% assessment of tasks in laboratory work;
- 30% intermediate control (3 control works)
- 20% coursework

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

Lecture 1	Creating and using C ++ classes. Encapsulation. Exception handling	**Laboratory work 1**	Classes and objects in C ++. Overloading operations Innovation Campus: PT08		
Lecture 2	Inheritance. Using polymorphism and templates in C ++	**Laboratory work 2**	Class composition. Inheritance. Innovation Campus: PT08, PT10		
Lecture 3	Using the tools of the standard C ++ library.	**Laboratory work 3**	Virtual and abstract classes. Exceptions. Innovation Campus: PT08, PT10, PT11		
Lecture 4	C ++ containers. C ++ algorithms.	**Laboratory work 4**	Containers. Templates. standard libraries. Innovation Campus: PT08, PT11		
Lecture 5	Using basic Java language tools	**Laboratory work 5**	Graphical user interface Innovation Campus: PT11		
Lecture 6	Working with Java arrays and strings. Creating classes	**Laboratory work 6**	Classes and objects in Java. Nested classes. Composition. Innovation Campus: PT 03, PT21		
Lecture 7	Use of polymorphism. Working with generalizations and collections in Java	**Laboratory work 7**	Inheritance. interfaces and abstract classes. Innovation Campus: PT21		
Lecture 8	Working with exceptions and files in Java	**Laboratory work 8**	Inheritance. interfaces and abstract classes. Exceptions. Generalization. Innovation Campus: PT21	**Self-Study**	Working with character streams and byte streams
Lecture 9	Working with XML documents in Java	**Laboratory work 9**	Containers. Innovation Campus: PT22		
Lecture 10	Creating a Java GUI application	**Laboratory work 10**	Graphical user interface Innovation Campus: PT22		JavaFX application architecture.
Lecture 11	Using basic C # tools.	**Laboratory work 11**	Classes and objects in C #. Reloading operations Innovation Campus: PT23		
Lecture 12	Working with C # arrays and strings. Creating classes. Inheritance.	**Laboratory work 12**	Nested classes. Composition. Inheritance. Innovation Campus: PT23		Structures and enumerations
Lecture 13	C # polymorphism. Interfaces. Design and implementation of generalized classes and methods. Exception	**Laboratory work 13**	Virtual and abstract classes. Exceptions. Innovation Campus: PT23		

	handling.				
Lecture 14	Working with files. Working with C # XML documents. Creating and using generalizations.	**Laboratory work 14**	Generalization. Containers. Innovation Campus: PT24		Creating your own container types.
Lecture 15	Creating a .NET GUI application C # Delegates. Events.	**Laboratory work 15**	Graphical user interface Innovation Campus: PT24		GDI+ graphics tools
Lecture 16	Creating a .NET GUI application C # Delegates. Events.	**Laboratory work 16**	UML charts Innovation Campus: PT10, PT24		

RECOMMENDED READING

Compulsory	1. Bjarne Stroustrup. (2007) The C++ Programming Language. Third Edition. Addison-Wesley, 2. B. Lippman, Josee Lajoie (2008) C++ Primer. Third Edition. Addison-Wesley, 3. Savich U. (2004) Programming in C ++ St. Petersburg: Peter; Kiev: Publishing group BHV. 4. Nikulina O. M. (2014) Fundamentals of the program in the visual middle. Methodical instructions to the laboratory to take the course "System program" Kharkiv: NTU "KhPI". 5. Schildt, G.(2015) Java 8. The Complete Guide; 9th ed. Moscow: LLC "I.D. Williams". 6. Horstmann K.S, Cornell G. (2007) Java 2. Library of the Professional, Volume 1. Fundamentals, 7th ed. Moscow: Publishing house "Williams". 7. Shieldt G., Holmes D. (2005) The art of Java programming. Moscow: Publishing house "Williams". 8. Eckel B.(2009) Philosophy of Java. Programmer's library. 4th ed. St.Peterburg 9. Kopytko M.F, Ivankiv K.S.(2002) Basics of Java programming: Lecture texts. Lviv: LNU Publishing Center. Ivan Franko 10. Golub B. M.(2006) C #. Concept and syntax. Teaching. manual / BM Golub Lviv: Ivan Franko Lviv National University Publishing Center 11. Brnakevich I.E., Vagin P.P. (2002) Java programming: the use of fundamental classes: Lecture texts. Lviv: Ivan Franko Lviv National University Publishing Center 12. Shildt G. C.(2009) # 3.0. Complete guide. Dialectics-Williams.	**Recommended**	1. Walpa O.D.(2006) Borland C ++ Buelder. Express course. 2. Basics of the program (part 2). Rozrobnik to the course of L.V. Ivanov. Retryeved from: http://www.iwanoff.inf.ua/programming_2_ua/index.html 3. Deytel H.M.,(2006) Deytel How to Program in Java. Book 2. Files, networks, databases. Binom, 4. Object-oriented programming (part 1). Course developer LV Ivanov. Retryeved from: http://www.iwanoff.inf.ua/oop_ua/index.html 5. Richter J.(2017) Programming on Microsoft .NET Framework 2.0 in C. 6. Richter J., Van de Bospurt Williams M. (2014) Programming in C # for professionals / J. Richter, 7. Object-oriented programming (part 2). Course developer LV Ivanov. Retryeved from: http://www.iwanoff.inf.ua/oop_ua/index.html 8. Design Patterns: Elements of Reusable Object-Oriented Software Retryeved from: // http://www.uml.org.cn/c++/pdf/DesignPatterns.pdf 9. Brnakevich I. E. Vagin. P.P.(2002) Programming in Java: the use of fundamental classes: Lecture texts, Lviv: Ivan Franko Lviv National University Publishing Center. 10. Dudziany I. M. (2007) Object-oriented modeling of software systems: Textbook. manual. Lviv: Ivan Franko Lviv National University Publishing Center.

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.