SYLLABUS

INTERNATIONAL EUROPEAN UNIVERSITY

EUROPEAN SCHOOL OF BUSINESS

Object-oriented programmin

2023

SYLLABUS

Course Name	
	Object-oriented programmin
Lecturer (s)	
	Serhii Mitin, senior lecturer at the Department of Information Technology
Lecturer’s profile	
	https://it.ieu.edu.ua/pro-yeash/struktura-yeash/kafedra-informatsiinykh-tekhnolohii#zzz-007
Consultations	
online consulting	–
offline consulting	Friday 3 p.m. – 5 p.m. room 509
Contact number	
	+380631032700
E-mail	
	serhiimitin@ieu.edu.ua
Discipline page	
	https://classroom.google.com/u/1/c/NTUxNTIyMDU4MjIz

Form of final control	test	def. test	exam
	☐	☒	☒

SYLLABUS

1 Brief discipline annotation

The Object-oriented programming discipline is aimed at training Bachelors of knowledge area: 12 Information technology, specialty: 121 Software engineering. It is one of the fundamental natural science disciplines for future software developers.

2 Background for studying discipline

The discipline program is based on the knowledge acquired while learning the Basics of programming discipline_1.

3 Goal and objectives of the discipline

The goal of the discipline is to provide students with algorithmic thinking abilities and skills in program development using object-oriented programming (OOP) technology.
Objectives of the discipline:
- to master the object-oriented approach for the design and development of software systems;
- to understand the object-oriented design methodology, master it and use it throughout the software life cycle;
- to develop software using state-of-the-art software development tools.

4 Learning outcomes

After learning the discipline, students should
know:
- principles of information systematization;
- advanced technologies and tools for developing program systems;
- programming paradigms;
- modern programming languages;
- basic data structures and algorithms;
- methods and standards of documentation design

be able to:
- apply knowledge of fundamental disciplines to solve professional problems;
- use programming languages, information technology description languages, specification languages;
- use tools for designing and developing information systems, goods and services of information technologies;
- apply efficient algorithms to solve professional problems;
- use state and international standards in information technologies;
- use state-of-the-art technologies and development tools at all stages of the IS life cycle;
- model systems and processes, states and behavior of complex informatization objects in the process of designing information systems and technologies;
- master modern technologies of automation of design of complex objects and systems, products and services of information technologies, modern paradigms and programming languages;
- compile technical documentation

SYLLABUS

5 ECTS credits

4 ECTS credits / 120 academic hours

6 Course Content

Sections and topics	Type of classes/hours		
	Lectures	Laboratory work	Independent work
Section 1: Content section 1: Object-oriented programming in C++ language			
Topic 1.1. Key principles of OOP	2		4
Topic 1.2. Programming using classes		2	2
Topic 1.3. Developing and setting up programs for processing object data types		2	2
Topic 1.4. The concept of object-oriented analysis and design	2		4
Topic 1.5. Canonical diagrams of the Unified Modeling Language (UML)		2	2
Topic 1.6. Inheritance and class hierarchies. Simple and multiple inheritance	2		4
Topic 1.7. Programming of simple and multiple inheritance class hierarchies.		4	4
Topic 1.8. Program development using inheritance		2	2
Topic 1.9. Polymorphism	2		4
Topic 1.10. Programming of function and operation overloading.		2	4
Topic 1.11. Designing virtual functions and polymorphic clusters		2	2
Content section 2: Introduction to generalized programming			
Topic 2.1. Parametric polymorphism. Function templates. Class templates	2		4
Topic 2.2. Program development using function templates		2	2
Topic 2.3. Program development using class templates		2	2

SYLLABUS

6 Course Content

Topic 2.4. Program development using parametrically polymorphic classes and functions		2	2
Topic 2.5. Methods of object composition and interaction	2		4
Topic 2.6. Development of object interaction programs		2	2
Content section 3. C++ standard libraries			
Topic 3.1. C++ standard libraries	2		4
Topic 3.2. Developing and setting up programs of streaming and file I/O using C++ tools		2	2
Topic 3.3. Development and debugging of programs for processing data of sequential container classes		2	2
Topic 3.4. Iterators and associative containers of the standard template library	2		4
Topic 3.5. The use of iterators and associative containers for solving typical data processing tasks		2	2
Topic 3.6. Application development using the templates library.		2	2
Pass/Fail test		2	6

7 List of obligatory tasks

1. Class relation and class hierarchy
2. Types of polymorphism and concept of generalized programming.
3. Inheritance. Access control in inheritance.
4. Virtual functions.
5. Generalized programming. Function templates and template classes
6. Purpose and composition of C++ standard library

8 List of selective tasks

1. CASE tools and CASE technologies. Composition of CASE tools
2. Event-based programming in MSVS
3. Controls in applications with window interface

9 Discipline features

Period of teaching	Semester	International disciplinary integration	Year of study	Courses: general training/ professional training/elective
1 semesters	3rd semester	available	2	Professional training

10 Hardware and software

Personal computer, program development environments, graphical and text editors.

11 Assessment system and requirements

As part of discipline teaching, one carries out the current and final control of students' knowledge. The final grade is given according to the total rating of students.
The results of the current control of students' knowledge is assessed in general between 0 and 100 scores.
Students are admitted to the final control if they fulfil the requirements of the training program and obtain at least 36 scores for the current learning activity.
Final assessment of students' knowledge is conducted in the form of Pass/Fail test.
The overall score of the discipline is 100. The total grade for the discipline is given according to the national and European scale.

12 Discipline policy

Teaching of the discipline is based on cutting-edge educational technologies aimed at increasing the level of students' interest in the course, providing theoretical and practical knowledge of the discipline. To activate the learning and cognitive activity of students, the discipline includes the consolidation of knowledge obtained at the lecture and acquisition of practical skills in lecture topics during laboratory classes.

13 Absence policy

Scores are not given for missed lectures. If students miss a laboratory work, they should perform all tasks of the missed laboratory work before the next laboratory work and present the results to the lecturer.
Students who have missed classes without valid reasons and have not participated in current control activities are not admitted to the final semester control. In this case, a mark 'non-admission' is put in the exam record on the day of the exam.
Repeated taking of the exam of the discipline is appointed in case of accomplishing all types of educational, individual work stipulated by the working program of the academic discipline and is carried out according to the approved schedule of academic failure liquidation.

14 Policy of late task performance

Tasks and laboratory works submitted later are assessed with a lower grade. The grade is reduced by one point for each week of lateness.

15 Academic integrity policy

Participants in the educational process rely on the academic integrity principles. One should provide references to sources of information when using someone else's ideas, statements, data, as well as verified information.

16 Recommended sources of information

Primary:
1. Schildt H.S. C for professional programmers. Translated from English. Williams, 2011. 704 p.
3. Ian Graham. Object-oriented methods. Principles and practice. Translated from English. Williams, 2004. 880 p.
3. Fowler M. UML. Fundamentals. A Brief Guide to the Standard Object Modeling Language. 3rd edition. Translated from English. 2004
Additional:
1. Laforet R. Object-Oriented Programming in C++. Translated from English. 2018, 928 p.
2. Nicolai Josuttis. C++ Standard Library: Reference Guide, 2nd edition: Translated from English. Kyiv: Williams, 2014. 1136 p.
Internet resources:
https://msdn.microsoft.com/uk-ua/
http://www.uml.org/
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2013/n3690.pdf.

17 Tips on successful study during the course

Note: examine lecture materials and perform tasks and laboratory works synchronously with the curriculum. Thus, your abilities and insistence will be the key to success!