INTERNATIONAL EUROPEAN UNIVERSITY
Education and Research Institute "European Business School"
Department of Information Technology

Approved by
The Scientific and Methodical Council of the University, protocol dd._____, 2023, No. _____

Chair of SMC _____

WORKING PROGRAM OF THE ACADEMIC DISCIPLINE:

CROSS-PLATFORM PROGRAMMING

Knowledge area: 12 Information Technology

Specialty: 121 Software Engineering

Educational program: Software Engineering

Discipline status: Elective

Kyiv – 2023

The working program of the Cross-platform programming academic discipline is based on the 121 Software Engineering educational and professional program for the first (Bachelor) level of the 121 Software Engineering specialty approved by the University Academic Council on May 26, 2022, protocol No. 4.

Developer: Oleksandr Nesterenko, Doctor of Science (Techn.), professor

Reviewers: Zoia Sherman, PhD in Physics and Mathematics
Oleksandr Falovskyi, PhD in Technology

Guarantor of the educational program: Oleksandr Nesterenko, Doctor of Science (Techn.), professor

The working program of the academic discipline is reviewed and approved by the Department of Information Technology, protocol dd. August 31, 2023, No. 1.

Head of the Department
Doctor of Science (Techn.),
professor  O.V. Nesterenko,

The program is reviewed and approved by the Academic Council of the European Business School, protocol dd. September 11, 2023, No. 1.

Chair of the Academic Council _____ Y.S. Remyha,
PhD in Economics, associate professor,
Acting Director of the European Business School

Registration No. __5/23_____

INTRODUCTION

The **program of the Cross-platform programming academic discipline** is designed according to the Higher Education Standard of Ukraine (hereinafter referred to as the Standard) of the knowledge area: 12 Information Technology, specialty: 121 Software Engineering.

Discipline description (annotation). This discipline is one of the elective disciplines for professional training future software developers.

Table 1

<table>
<tr><td rowspan="2">Criteria</td><td rowspan="2">Knowledge area, training program, educational level</td><td colspan="2">Discipline characteristics</td></tr>
<tr><td>full-time mode of study</td><td>part-time mode of study</td></tr>
<tr><td>Number of credits – 4</td><td>Knowledge area:
12 INFORMATION TECHNOLOGY</td><td colspan="2"><u>Elective</u></td></tr>
<tr><td>Sections – 1</td><td rowspan="8">Specialty:
121 SOFTWARE ENGINEERING</td><td colspan="2">Year of training</td></tr>
<tr><td>Content sections – 1</td><td>2023</td><td></td></tr>
<tr><td rowspan="2">Individual research task:</td><td colspan="2">Semester</td></tr>
<tr><td>3rd</td><td>3rd</td></tr>
<tr><td rowspan="2">Total amount of hours – 120</td><td colspan="2">Lectures</td></tr>
<tr><td>16 hours</td><td>4 hours</td></tr>
<tr><td rowspan="4">Weekly load:
class hours – 3
independent work of students – 4</td><td colspan="2">Practical and laboratory classes</td></tr>
<tr><td>32 hours</td><td>8 hours</td></tr>
<tr><td rowspan="4">Educational level:
Bachelor</td><td colspan="2">Independent work</td></tr>
<tr><td>72 hours</td><td>108 hours</td></tr>
<tr><td colspan="2">Type of control:</td></tr>
<tr><td>Pass/Fail test</td><td>Pass/Fail test</td></tr>
</table>

Subject matter of the academic discipline: theory and practice of applying platform-independent algorithmic and data structures in programming based on cutting-edge software development technologies.

Interdisciplinary links: The academic discipline is related to such disciplines as Operating systems, Fundamentals of programming and Object-oriented programming.

1. GOAL AND OBJECTIVES OF THE ACADEMIC DISCIPLINE

1.1. The **goal** of the Cross-platform programming discipline is to provide Bachelors with theoretical knowledge and practical skills in component

programming, principles of cross-platform software system development technology, principles of using cross-platform programming tools.

1.2. **Key objectives** of the Cross-platform programming discipline:

- to learn environments and tools for creating cross-platform programs;
- to master basic concepts, concepts and principles of cross-platform programming;
- to develop knowledge and skills of designing cross-platform interfaces;
- to develop knowledge and skills of designing cross-platform programs;
- to master basic concepts, concepts and principles of component programming;
- to develop knowledge and skills of designing components and libraries.

1.3. **Competencies and learning outcomes** encouraged by the discipline (interrelation with the statutory content of student training stipulated in learning outcome terms of the Standard).

According to the Standard requirements, the discipline provides students with the following ***competencies***:

Table 2

Integral competence	Ability to solve complicated specialized tasks and practical problems in software development characterized by complexity and uncertainty of conditions.
General competencies	Understanding of the processes occurring in the computer's program environment. Ability to apply knowledge in practical situations.
Specialized (professional, subject) competencies	Understanding of interactions in the computer's program and operating environment. Ability to think algorithmically and logically. Understanding of possibilities of applying methods and tools of operating systems in practice when developing application software.

Specification of competencies according to the National Qualifications Framework descriptors in the Competency matrix form is given in Table 3.

Integrated final program learning outcomes encouraged by the academic discipline:

Program learning outcomes Bachelor's qualifying paper

Table 3

Competency matrix

No.	Competence	Knowledge	Skills / Abilities	Communication	Autonomy and responsibility
Integral competence					
1.	Ability to solve complicated specialized tasks and	Theories of designing operating systems	To use information technologies,	Software interaction	Independent design and

	practical problems in software development characterized by complexity and uncertainty of conditions.	and interaction with application software	basic system and application software to solve practical problems		testing on the production site
General competencies					
2.	Understanding of the processes occurring in the operating system when processing information with application software. Ability to use knowledge in practical programming situations. Ability to search, process and analyze information for application in programming.	structure of modern information systems, general principles of their functioning	apply operations of interaction with the OS environment in when developing application programs	Relation between theoretical and practical knowledge	Monitoring of information processing processes
Specialized (professional, subject) competencies					
3.	Understanding of interactions in the computer's program and operating environment. Ability to think algorithmically and logically. Understanding of possibilities of applying methods and tools of operating systems in practice when developing application software.	basics of multiprogramming of virtualization and distributed computing, relevant rules and functions of application programming	the use of software tools in the operating environment	Application of parallel work technologies	Description of information processes

Learning outcomes:

After learning the discipline, students should

know:

- principles of cross-platform software system development technology;
- principles of using cross-platform programming tools;
- theoretical foundations in the application of cross-platform programming tools;
- architecture and standards of component models, communication tools and distributed computing;
- strategies for integrating software components;
- basic middleware platforms and component models;
- formal and visual methods of component design;
- basics of designing cross-platform programs in Java, C #. C++;

be able to:

– use programming tools, basic system calls to solve practical problems of applied programming;

– use functions and libraries of software tools to develop application programs;

– use the basic principles of developing cross-platform software systems for various projects.

2. INFORMATION CAPACITY OF THE ACADEMIC DISCIPLINE

Content module 1. Basic concepts of cross-platform programming

Topic 1.1. Introduction. Definition of cross-platforming
Topic 1.2. Programming languages implementing cross-platforming at the compilation level
Topic 1.3. Programming languages implementing cross-platforming at the runtime level
Topic 1.4. Development environments

Content module 2. Development of Java applications

Topic 2.1. Fundamentals of Java programming.
Topic 2.2. Variables, Operators, Operations.
Topic 2.3. Elements of object-oriented programming.
Topic 2.4. Design and development of library components on the Java platform.
Topic 2.5. Development of Java applets.

Content module 3. Fundamentals of the Android platform

Topic 3.1. Fundamentals of the Android platform.
Topic 3.2. Application and user interface development.
Topic 3.3. Publication of an Android application.

3. STRUCTURE OF THE ACADEMIC DISCIPLINE

Content modules and topics	Amount of hours				
	Total	including			
		Lectures		Laboratory works	Independent work
Content module 1. Basic concepts of cross-platform programming					
Topic 1.1. Introduction. Definition of cross-platforming	10	2		2	6
Topic 1.2. Programming languages implementing cross-platforming at the compilation level	9	1		2	6

Topic 1.3. Programming languages implementing cross-platforming at the runtime level	9	1		2	6
Topic 1.4. Development environments	10	2		2	6
Total per content module	38	6		8	16
Content module 2. Development of Java applications					
Topic 2.1. Fundamentals of Java programming.	9	1		2	6
Topic 2.2. Variables, Operators, Operations.	9	1		2	6
Topic 2.3. Elements of object-oriented programming.	11	1		4	6
Topic 2.4. Design and development of library components on the Java platform.	11	1		4	6
Topic 2.5. Development of Java applets.	12	2		4	6
Total per content module	52	6		16	30
Content module 3. Fundamentals of the Android platform					
Topic 3.1. Fundamentals of the Android platform.	9	1		2	6
Topic 3.2. Application and user interface development.	11	1		4	6
Topic 3.3. Publication of an Android application.	10	2		2	6
Total per content module	30	4		8	18
Total hours	120	16		32	72

4. TOPICS OF LECTURES

No.	Topic (brief content)
1	Topic 1.1. INTRODUCTION. DEFINITION OF CROSS-PLATFORMING. Classification and features of modern programming languages. Examples of cross-platform software. Emulators. Levels of cross-platforming: hardware / software, compilation / execution.
2	Topic 1.2. PROGRAMMING LANGUAGES IMPLEMENTING CROSS-PLATFORMING AT THE COMPILATION LEVEL. C, C++ Topic 1.3. PROGRAMMING LANGUAGES IMPLEMENTING CROSS-PLATFORMING AT THE RUNTIME LEVEL. Java, C#.
3	Topic 1.4. DEVELOPMENT ENVIRONMENTS. Microsoft Visual Studio, Rider, Eclipse, NetBeans, IntelliJ IDEA programming environments.
4	Topic 2.1. FUNDAMENTALS OF JAVA PROGRAMMING. The concept of expression. Basic input and output operators. Input and output format of different types of data (numbers, strings, pointers, etc.). The use of command line arguments. Topic 2.2. VARIABLES, OPERATORS, OPERATIONS. Arithmetic operations. The assignment operator. Increment and decrement operators. Bitwise logical operations. Left and right shift operations. Comparison operators. Priority and order of operations.

5	Topic 2.3. ELEMENTS OF OBJECT-ORIENTED PROGRAMMING. Objects. Classes. The relationship between classes. Nested and inner classes in Java. Anonymous classes. Abstract classes and methods. Topic 2.4. DESIGN AND DEVELOPMENT OF LIBRARY COMPONENTS ON THE JAVA PLATFORM. Designing the EJB 4 component. Designing and developing a graphical interface.
6	Topic 2.5. DEVELOPMENT OF JAVA APPLETS. JVM virtual machine. Syntax for calling an applet. The system of protection. Testing in the AppletViewer program.
7	Topic 3.1. FUNDAMENTALS OF THE ANDROID PLATFORM. Introduction to the Android platform. The level of the kernel. The level of libraries. The level of applications. Overview of the Android SDK. Topic 3.2. APPLICATION AND USER INTERFACE DEVELOPMENT. The concept of an application resource. Types of resources. Working with resources in the Eclipse environment. Strings. Sizes and colors in Android. Arrays of strings. Graphics. Features of UI development for mobile platforms. General characteristics of screens. The main rules of layout. The use of different versions of the program design. UI editor in Eclipse.
8	Topic 3.3. PUBLICATION OF AN ANDROID APPLICATION. Activity and the Intent class. Working with JSON. WebView. Library for API operations.

5. TOPICS OF LABORATORY AND PRACTICAL CLASSES

No.	Laboratory work	Amount of hours
1	Modern programming languages; programming environments	2
2	Cross-platforming at the compilation level.	2
3	Cross-platforming at the runtime level.	2
4	Introduction to the Eclipse development environment	2
5	Data types in Java.	2
6	Conditional operators, loops, arrays.	2
7	Creation of objects.	4
8	Design and development of library components on the Java platform	4
9	Developing Java applets	4
10	Screen elements and their properties.	2
11	Event handlers.	4
12	Development of the user interface.	2

6. INDEPENDENT WORK

No.	Topic (brief content)
1	Examples of cross-platform software.
2	Syntax and semantics of C, C++, Java, C#.
3	Eclipse development environment.
4	Nested and inner classes in Java.
5	Features of using objects, blocks, classes.

6	Designing the EJB 4 component.
7	Designing and developing a graphical interface.
8	Development of Java applets.
9	The level of Android applications.
10	Strings, sizes and colors in Android.
11	UI editor in Eclipse. XML layout structures.
12	Storage of data and status in Android

7. TRAINING METHODS

Teaching the Cross-platform programming discipline, one uses information and practical training methods: classical lectures, laboratory and practical classes using simulation laboratory workshops, as well as consultations on the accomplishment of independent work of students, written assignments.

Methods of learning and cognitive activity: explanatory and illustrative method, reproductive method, problem presentation method, partially exploratory or heuristic method, research method.

Methods of stimulation and motivation of learning and cognitive activity: inductive and deductive teaching methods; methods of stimulation and motivation of learning.

8. CONTROL METHODS

The plan of the Cross-platform programming discipline implies carrying out of current and final control.

Current control is the assessment of the level of knowledge, skills and abilities of students carried out during the educational process by conducting a written survey at the end of sections (module colloquium). Final control is carried out in the form of a Pass/Fail test.

9. FORM OF STUDENT PERFORMANCE FINAL CONTROL

The form of final control is the **Pass/Fail test** taken on-campus (or in the form of computer test in case of a specific situation) in the period stipulated by the Dean's office or according to the individual schedule stipulated by the curriculum.

10. SCORING SYSTEM

Scoring during the semester

No.	Type of activity	Number of points per didactic unit	Number	Total points
1	Accomplishment of tests	2	8	16

2	Accomplishment of laboratory works	4	8	32
3	Accomplishment of independent tasks	1.5	8	12
Maximum grade				60

General assessment of student knowledge due to current control

The results of current control of student knowledge are assessed in general ranging from **0** to **60** points.

Students are allowed to final control if they fulfil the requirements of the training program and obtain at least **36** points for the current learning activity.

Final assessment of student knowledge

Final assessment of student knowledge is conducted in the form of the **Pass/Fail test**.

Allocation of assessment points during final control in the academic discipline

Grade in points for final assessment	Grade according to the national scale
35-40	Excellent
21-34	Good
10-20	Satisfactory
less than 10	Fail

Assessing the answer to the particular question, one takes into account the following gaps and mistakes:

– untidy preparation of work (nonconventional abbreviations, unclear handwriting, use of pencils instead of clear inks) (minus **2** points);

– incorrectness in certain economic categories and definitions (minus **4** points).

Assessment criteria for answers to theoretical questions of the exam card:

1. The full answer to the question rated as ***excellent*** should correspond to the following requirements:

– detailed, comprehensive representation of the content of the given problem;

– full list of economic categories and laws required to reveal the question;

– ability to carry out a comparative analysis of various theories, concepts, approaches and make logical conclusions and generalizations;

– ability to apply methods for the scientific analysis of economic phenomena, processes and characterize their features and forms of appearance;

– demonstration of the ability to express and reason your own attitude to alternative views on this question;

– use of relevant actual and statistical data, knowledge of dates and historical periods that prove key points of the answer.

2. The answer to the question is rated as ***good*** if:

– the answer for the highest grade does not reveal at least one of the above-mentioned points (if it is definitely required to reveal the question comprehensively), or if:

– revealing the question correctly in general according to the above-mentioned requirements, one makes some mistakes while using digital materials.

3. The answer to the question is rated as ***satisfactory*** if:

– the answer for the highest grade does not reveal four and more points specified in its requirements (if they are required to reveal the question comprehensively);

– there are four or more gaps characterizing individually assessment criteria;

– conclusions made during the answer do not correspond to correct or generally defined ones with the absence of evidence for opposite facts given in the answer;

– the character of the answer gives reason to state that persons fail to understand the question properly or do not know the correct answer, and that is why fail to answer in actual fact, making serious mistakes.

National and ECTS grading scale

<table>
<tr><td rowspan="2">Sum of points for all types of educational activities</td><td rowspan="2">ECTS grade</td><td colspan="2">Grade according to the national scale</td></tr>
<tr><td>for exam, term paper, practical training</td><td>for Pass/Fail test</td></tr>
<tr><td>90-100</td><td>A</td><td>excellent</td><td rowspan="5">pass</td></tr>
<tr><td>82-89</td><td>B</td><td rowspan="2">good</td></tr>
<tr><td>74-81</td><td>C</td></tr>
<tr><td>66-73</td><td>D</td><td rowspan="2">satisfactory</td></tr>
<tr><td>60-65</td><td>E</td></tr>
<tr><td>30-59</td><td>FX</td><td>fail with possible repeated pass</td><td>fail with possible repeated pass</td></tr>
<tr><td>1-29</td><td>F</td><td>fail with obligatory repeated learning of the discipline</td><td>fail with obligatory repeated learning of the discipline</td></tr>
</table>

The overall final grade in points according to the national and ECTS scales is put into the examination and test register, academic card and credit book of students.

11. METHODICAL SUPPORT:

- working program of the discipline;
- electronic course on the e-learning platform;
- plans of lectures, practical classes and independent work of students;
- key points of discipline lectures;
- methodical guidelines to laboratory and practical classes for students;
- methodical materials for independent work of students;
- test tasks for lecture topics;
- list of questions for the exam.

12. RECOMMENDED READING

Primary:

1. Onyshchenko V.V., Dovzhenko T.P. Specialized programming languages: study guide. Kyiv: DUT. 2019. 146 p.
2. Rovinskyi V.A. Study guide for the Cross-platform programming course. Ivano-Frankivsk: Vasyl Stefanyk Precarpathian National University, 2020. 151 p.
3. Kostenko A.V., Kostyrko V.S., Plesha M.I. Cross-platform programming: study guide. Lviv: LTEU Publishing House, 2019. 247 p.
4. George Heinemann, Gary Pollis, Stanley Selkow. Algorithms. A reference book with examples in C, C++, Java and Python. Translated from English. Dialectics, 2017. 432 p.
5. Deitel P., Deitel H., Deitel E., Morgano M. Android for programmers: developing applications. Translated from English. St. Petersburg: Peter, 2011. 560 p.
6. Hashemi S., Komatineni S., McLean D. Android Application Development. Translated from English. St. Petersburg: Peter, 2011. 736 p.

Additional:

1. Pomorova O.V., Hovorushchenko T.O. Designing user interfaces: study guide. Khmelnytskyi: KHNU. 2011. 206 p.
2. International Standard ISO/IEC 14882:2014(E) – Programming Language C++, ISBN-13: 978- 0321563842: [Electronic resource]. – Available at: https://isocpp.org/std/the-standard.
3. C/C++ language and standard libraries reference: [Electronic resource]. – Available at: https://msdn.microsoft.com/en-us/library/hh875057.aspx.
4. Fundamentals of programming: study guide for students in the 123 Computer Engineering specialty / V.H. Zaitsev, I.P. Drobiazko; Igor Sikorsky Kyiv Polytechnic Institute. 2019. – 240 p. [Electronic resource]
5. Burd B. Android® Application Development All-in-One For Dummies® , 3rd Edition. Hoboken: John Wiley & Sons, Inc., 2020. – 785 p.
6. Griffiths David, Griffiths Dawn. Head First Android Development. Sebastopol: O'Reilly Media, Inc., 2022. – 1410 p.
7. Phillips B. Stewart K. Marsicano K. Android. Programming for professionals. 2nd edition. Kyiv: Dialectics, 2021. 960 p.

Information resources

1. Building Multiplatform Projects with Gradle – Kotlin Programming Language. URL: https://kotlinlang.org/docs/reference/building-mpp-with-gradle.html
2. Building your first Kotlin Multiplatform app – Getting Started. URL: https://vivekc.xyz/building-your-first-kotlin-multiplatform-app-getting-started-8ad10d7d4e9f
3. Configure your build. Android Developers. URL: https://developer.android.com/studio/build
4. The Build Process objc.io. URL: https://www.objc.io/issues/6-build-tools/buildprocess/