KHMELNYTSKYI NATIONAL UNIVERSITY

APPROVED
Dean of IT Faculty
T. HOVORUSHCHENKO
"01" 09 2025.

WORKING PROGRAMME OF THE EDUCATIONAL COMPONENT

Software engineering basics

Field of Study: F – Information Technology
Specialty: F2 – Software Engineering
Level of Higher Education: First (Bachelor's) Level
Educational and Professional Programme: Software Engineering
Course Load: 8 ECTS credits **Course Code:** CPT.05
Language of Instruction: English
Status of the Educational Component: Compulsory (Professional Training)
Faculty: Faculty of Information Technology
Department: Department of Software Engineering

Form of Study	Year	Semester	Total Credits		Number of hours						Course project	Coursework	Semester control form	
					Contact Hours					Independent Work (incl. Individual Tasks)			pass/ fail test	Exam
			ECTS credits	hours	Total	Lectures	Laboratory works	Practical classes	Seminar classes					
D	1	2	5	150	50	16	34			100			+	

The working programme is based on the Educational and Professional Programme "Software Engineering" within the specialty F2 "Software Engineering".

Program's author _____ Anastasia DOMINA

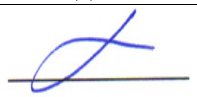
Approved at the meeting of the Department of Software Engineering
Minutes No. 1 dated August 28, 2025
Head of the Department _____ L.P. Bedratyuk

The working programme was reviewed and approved by the Academic Council of the Faculty of Information Technology

Chair of the Academic Council _____ Tetiana HOVORUSHCHENKO

Khmelnytskyi 2025

ЛИСТ ПОГОДЖЕННЯ

Посада	Назва кафедри	Підпис	Ім’я, ПРІЗВИЩЕ
Зав. кафедри, д-р фіз.-мат. наук, проф.	Інженерії програмного забезпечення		Леонід БЕДРАТЮК
Гарант ОП, д-р фіз.-мат. наук, проф.	Інженерії програмного забезпечення		Леонід БЕДРАТЮК

SOFTWARE ENGINEERING BASICS

Type of Educational Component	Compulsory
Level of Higher Education	First (Bachelor's) Level
Language of Instruction	English
Semester	Second
Number of ECTS Credits Assigned	5
Forms of Study the Course is Designed For	Full-time

Learning outcomes. After studying the discipline, the student should: have professional terminology and basic concepts of software engineering; *describe* areas of software engineering knowledge according to SWEBOK; characterize models and processes of the software life cycle based on current standards; to navigate modern software development methodologies and technologies; use methods, notations and tools for software modeling and design; build structural-functional, infological and object-oriented software models; evaluate the degree of adequacy of the developed models; document the decisions made based directly on the models; demonstrate a culture of thinking when developing models; analyze the subject area and determine the functional requirements for the software; analyze the problems and trends in the development of software engineering; apply the methods and means of software engineering in the study of the cycle of professionally oriented disciplines.

Course Content. Engineering basics of software. About modeling dreams. Fundamentals of software requirements engineering. Software development technologies. Ethical and cultural aspects of software engineering.

Planned Learning Activities. The minimum amount of classroom-based learning activities in one ECTS credit for a course at the first (Bachelor's) level of higher education in full-time study mode is 10 hours per 1 ECTS credit.

Forms (Methods) of Instruction: Lectures (using problem-based learning and visualisation methods), Laboratory works, Independent work

Assessment Methods: Laboratory work defence, Testing

Form of Final Assessment: pass/fail test

Learning Resources:

1. Rod Stephens. Beginning Software Engineering : Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada and the United Kingdom. 2022. 685 P.
2. Elvis C. Foster. Software Engineering. A Methodical Approach: Second Edition. Published 2022 by CRC Press, Taylor & Francis Group. Boca Raton-London-New York. 579 p.
3. Mr. Rohit Manglik, Principles of Software Engineering. (2024). EduGorilla Publication, 296 p.
4. Rod Stephens. Beginning Software Engineering: Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada and the United Kingdom. 2022. 685 p.
5. Elvis C. Foster. Software Engineering. A Methodical Approach: Second Edition. Published 2022 by CRC Press, Taylor & Francis Group. Boca Raton-London-New York. 579 p.
6.University Electronic Library. [Electronic resource]. – Available at: http://library.khmnu.edu.ua/
7. Institutional Repository of Khmelnytskyi National University. [Electronic resource]. – Available at: http://elar.khmnu.edu.ua/jspui/?locale=en
Modular Learning Environment. [Electronic resource]. – Available at: https://msn.khmnu.edu.ua/

Teacher: assistant Domina A.I.

3 EXPLANATORY NOTE

«Software Engineering Basics» discipline, as one of the professional disciplines, occupies a leading place in the training of «bachelor» level specialists in the specialty 121 "Software engineering" under the educational and professional program «Software engineering» and is a theoretical and practical foundation for further study by students of the cycle of professionally oriented disciplines.

Prerequisites – CPT.03 Programming

Postrequisites – CPT.10 Software Modelling and Evaluation, CPT.14 Software Requirement Analysis and Quality

According to the Standard of higher education in the specified specialty and educational program, the discipline must ensure:

– ***competences***: IC – the ability to solve complex specialized tasks or practical problems of software engineering, characterized by complexity and uncertainty of conditions, using theories and methods of information technologies; PC02 – the ability to participate in software design, including modeling (formal description) of its structure, behavior and functioning processes; PC04 – the ability to formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards; PC05 – the ability to follow specifications, standards, rules and recommendations in the professional field when implementing life cycle processes; PC08 – the ability to apply fundamental and interdisciplinary knowledge to successfully solve software engineering tasks; PC10 – the ability to accumulate, process and systematize professional knowledge regarding the creation and maintenance of software and recognition of the importance of lifelong learning; PC11 – the ability to implement phases and iterations of the life cycle of software systems and information technologies based on appropriate models and software development approaches.

– ***program learning outcomes***: PLO01 – to analyze, purposefully search for and choose the necessary information and reference resources and knowledge for solving professional tasks, taking into account modern achievements of science and technology; PLO02 – know the code of professional ethics, understand the social significance and cultural aspects of software engineering and adhere to them in professional activities; PLO03 – to know the main processes, phases and iterations of the software life cycle; PLO04 – know and apply professional standards and other legal documents in the field of software engineering; PLO10 – conduct a pre-project survey of the subject area, system analysis of the project object; PLO11 – select initial data for design, guided by formal methods of requirements description and modeling; PLO12 – apply effective approaches to software design in practice; PLO14 – apply in practice instrumental software tools of domain analysis, design, testing, visualization, measurement and documentation of software; PLO16 – to have skills in team development, approval, design and release of all types of technical documentation; PLO23 – to be able to document and present the results of software development.

The purpose of the discipline. The purpose of the discipline is: to teach students the fundamentals of software engineering in accordance with the content of the international professional standard Guide to the Software Engineering Body of Knowledge (SWEBOK); to form in them a general idea of tasks, processes, methodologies, methods and means of information technologies used in software engineering; familiarize them with the basic provisions of software development technologies, including consideration of issues related to analysis, design, implementation, testing, documentation and support of software products.

Subject of discipline. Methods and means of software engineering with an emphasis on key aspects of theoretical and practical training in the processes of software development organization, taking into account the SWEBOK core of knowledge .

Tasks of the discipline. Formation of students of higher education basic theoretical knowledge and practical skills in the field of software engineering at all stages of the life cycle of software, mastering the necessary equipment for further study of the cycle of professionally oriented disciplines.

Learning outcomes. After studying the discipline, the student should: have professional terminology and basic concepts of software engineering; describe areas of software engineering knowledge according to SWEBOK; characterize models and processes of the software life cycle based on current standards; to navigate modern software development methodologies and technologies; use methods, notations and tools for software modeling and design; build structural-functional, infological and object-oriented software models; evaluate the degree of adequacy of the developed models; document the decisions made based directly on the models; demonstrate a culture of thinking when developing models; analyze the subject area and determine the functional requirements for the software; analyze the problems and trends in the development of software engineering; apply the methods and means of software engineering in the study of the cycle of professionally oriented disciplines.

4 STRUCTURE OF THE COURSE CREDITS

Topic Title	Number of hours allocated to:		
	Lectures	Lab work	Indep endent work
Topic 1. Engineering basics of software	2	4	12
Topic 2. Basics of modeling	4	8	24
Topic 3. Basics of software requirements engineering	4	8	24
Topic 4. Development technologies software	4	8	24
Topic 5. Ethical and cultural aspects of software engineering	2	6	16
Total	16	34	100

5 COURSE PROGRAM

5.1 CONTENT OF LECTURES

Lecture number	**List of lectures topics, their annotations**	**Hours number**
1	*2*	*3*
Topic 1. Engineering basics of software		
1	**Lecture 1. Introduction to software engineering** Introduction to software engineering, types of software, software cost components, roles in software engineering, standardization and SWEBOK structure, software life cycle and its processes, classic life cycle models (waterfall, spiral, iterative, incremental), modern development methodologies including RUP, MSF, MOF, and agile approaches such as Scrum, XP, Crystal, DSDM, FDD, and ASD. Lit.: [1], [2], [5]	2

Topic 2. Basics of modeling		
2	**Lecture 2. Structural and functional analysis and modeling. SADT methodology** Structural and functional modeling approaches in software engineering, SADT methodology and principles, functional modeling with IDEF0 and IDEF3, construction of diagrams and relevant CASE tools, Data Flow Diagrams (DFD), their syntax, semantics, notations (Jordan-De Marco, Heine-Sarson), hierarchical decomposition, tools for DFD modeling, comparison of SADT and DFD models, subject area concepts, infological modeling, levels of data modeling (conceptual, logical, physical), Entity-Relationship Diagrams (ERD), types of relationships, data normalization, ER notations, methods of ER-diagram construction, and CASE tools for info modeling automation. Lit.: [3], [6]	2
3	**Lecture 3. Unified modeling language UML.** Object-oriented approach to software development, its general concepts and principles, Rational Unified Process (RUP) methodology, characteristics and foundations of Unified Modeling Language (UML), types of UML diagrams, use case diagrams and methods for describing use cases, event streams and their textual descriptions, state and activity diagrams with their structural elements and construction features, usage scenarios with main and alternate flows, interaction diagrams including sequence and collaboration diagrams, their elements and construction specifics, class diagrams and their construction features, and CASE tools for object-oriented modeling automation. Lit.: [4 – 6]	2
Topic 3. Basics of software requirements engineering		
4	**Lecture 4. The process of developing software requirements** The concept of software requirements. Classification of requirements. Methodologies and standards regulating work with software requirements (SWEBOK, DSTU ISO/IEC/IEEE 12207:2018, ISO/IEC/IEEE 29148-2018, DSTU ISO/IEC TR 24766:2016). Properties of software requirements. Participants in the development of software requirements. The general process of developing software requirements. Sources of software requirements. Basic strategies for identifying requirements. Peculiarities of identifying requirements for a software product "to order" and for the open market. Feasibility analysis. Prototyping. Modeling. Communications about the project team and the customer in the process of developing software requirements. Lit.: [1], [2], [5]	2
5	**Lecture 5. Analysis, systematization and specification of software requirements** Purpose of software requirements analysis. Basic procedures for verification and validation of software requirements. Specification of software requirements. Specification of functional requirements for software using use cases. Formalization of requirements. Technical task for the software product. Rules for drawing up technical specifications. Lit.: [1], [2], [5]	2

Topic 4. Development technologies software		
6	**Lecture 13. Software design basics** Basic concepts of software design. Software design according to SWEBOK, according to standards DSTU ISO/IEC/IEEE 12207:2018, DSTU ISO/IEC/IEEE 42010:2018. Architectural and detailed design. Concept of software structure and architecture. The main types of software architectures. Architectural styles and templates (patterns). Physical presentation of architectural models of systems in the UML language (implementation diagrams). Methodical, technological, instrumental and organizational support for the software design process. Structural and functional software design. Software design using an object-oriented approach. User interface design. Software construction according to SWEBOK. Software construction management. Construction models. Measurement in construction. Overview of programming paradigms. Classification of programming languages. Applied and theoretical methods of programming. Tools and integrated programming environments. Translators. Coding. Module design rules. Integration. Program optimization. Ways to save memory. Ways to reduce execution time. Software design standards. Lit.: [1], [2], [5]	2 2
7	**Lecture 7. Basics of software testing and quality** Fundamentals of software testing including its purpose, main types, testing levels, and strategies; error localization and correction; testing peculiarities for object-oriented modules, integration, and user interfaces; verification and validation procedures; development of test plans and test data strategies; core concepts in software quality, quality metrics, models, and standards. Documentation development principles across the software life cycle: pre-project, project, user-oriented, electronic and interactive guides, documentation standards, and the role of technical writers. Software configuration management processes including stages, procedures, technological support, readiness levels (alpha, beta, release), development organization methods, maintenance techniques, reengineering, reverse engineering, and refactoring. Lit.: [1], [5]	2
Topic 5. Ethical and cultural aspects of software engineering		
8	**Lecture 8. Ethical and cultural aspects of software engineering** The concept of ethics and morality. Categories of ethics. Professional and corporate ethics. Computer ethics . Principles and the rules of behavior. The role of professional communities in the development of ethical responsibilities of software developers. IEEE-CS/ACM Code of Ethics and Professional Practice and its principles. Corporate culture of software development (using the example of the Microsoft company). Lit.: [2] , [5]	2
	Total	**16**

5.2. CONTENT OF LABORATORY WORKS

No	The topic of the laboratory session	Hours number
	Second semester	
1	Modeling based on electronic spreadsheets. Analysis of models Lit.: [7]	4
2	Structural and functional modeling according to the SAD T methodology (IDEF0) Lit.: [7]	4
3	Analysis and modeling of data flows. Decomposition of processes in IDEF3 notation	4

	Lit.: [7]	
4	Infologic modeling. Construction of entity-relationship diagrams Lit.: [7]	4
5	Analysis of software requirements. Construction of use case UML-diagrams Lit.: [7]	4
6	Analysis of software requirements. Construction of states and activity UML-diagrams Lit.: [7]	4
7	Analysis of software requirements. Construction of interaction (sequence and cooperation) UML-diagrams Lit.: [7]	4
8	Class design. Development of UML-diagrams of implementation. Lit.: [7]	6
	Total for the 2nd semester:	**34**

5.3. CONTENT OF INDEPENDENT WORK

Independent work of students of all forms of study involves systematic processing of the course material from relevant sources, preparation for laboratory works and testing. Students have access to the course page in the Modular Learning Environment, which contains the Working Programme of the course and the necessary teaching and learning materials.

Number of the week	**Type of independent work**	**Hours number**
1	*2*	*3*
1	Development of theoretical material. Preparation for the implementation of Lab 1.	4
2	Development of theoretical material. Preparation for the defense of Lab 1. Preparation for the implementation of Lab 2.	5
3	Development of theoretical material. Preparation for the defense of Lab2. Preparation for the implementation of Lab 3.	4
4	Development of theoretical material. Preparation for the defense of Lab 2. Preparation for the implementation of Lab 3.	5
5	Development of theoretical material. Preparation for the defense of Lab 3. Preparation for the implementation of Lab 4.	4
1	*2*	*3*
6	Development of theoretical material. Preparation for the defense of Lab 3. Preparation for the implementation of Lab 4.	4
7	Development of theoretical material. Preparation for the defense of Lab 4. Preparation for execution of Lab 5. Preparation for test control on topics 1-2.	5
8	Development of theoretical material. Preparation for the defense of Lab 4. Preparation for the implementation of Lab 5. Preparation for test control on topics 1-2.	5
9	Development of theoretical material. Preparation for the defense of Lab 5. Preparation for the implementation of Lab 6.	4

10	Development of theoretical material. Preparation for the defense of Lab 5. Preparation for the implementation of Lab 6.	4
11	Development of theoretical material. Preparation for the defense of Lab 6. Preparation for the implementation of Lab 7.	4
12	Development of theoretical material. Preparation for the defense of Lab 6. Preparation for the implementation of Lab 7.	4
13	Development of theoretical material. Preparation for the defense of Lab 7. Preparation for the implementation of Lab 8.	4
14	Development of theoretical material. Preparation for the defense of Lab 7. Preparation for the implementation of Lab 8.	4
15	Development of theoretical material. Preparation for test control on topics 3-5.	5
16	Development of theoretical material. Preparation for the defense of Lab 8. Preparation for test control on topics 3-5.	5
17	Development of theoretical material. Preparation for the defense of Lab 8.	4
	Total	**100**

Notes: TC – Test Control, T1–T7 – Topics of the course.

6. TEACHING TECHNOLOGIES AND METHODS

The learning process for the course is based on the use of both traditional and modern teaching technologies and methods, in particular: lectures (using visualisation methods, problem-based and interactive learning, motivational techniques, and information and communication technologies); laboratory works (using training exercises, problem situation analysis, explanation, discussions, etc.); independent work (study of theoretical material, preparation for laboratory works, ongoing and final assessment), with the use of information and computer technologies and distance learning technologies.

7. METHODS OF ASSESSMENT

Ongoing assessment is carried out during practical classes, as well as on the days of control activities established by the working programme and the academic schedule.
The following methods of ongoing assessment are used:
– test-based assessment of theoretical material;
– evaluation of the results of laboratory work defence.
When determining the final semester grade, the results of both ongoing assessment and final assessment are taken into account. The final assessment is conducted on all the material of the course according to examination papers prepared in advance and approved at the meeting of the department.
A student who has scored less than 60 percent of the maximum score for any type of academic work is not allowed to undergo the semester assessment until the amount of work stipulated by the Working Programme is completed. A student who has achieved a positive weighted average score (60 percent or more of the maximum score) for all types of ongoing assessment but has failed the examination is considered to have an academic debt.
Elimination of academic debt for the semester assessment is carried out during the examination session or according to the schedule set by the dean's office in accordance with the *Regulation on Control and Assessment of Learning Outcomes of Students at Khmelnytskyi National University*.

8. COURSE POLICY

The policy of the academic course is generally determined by the system of requirements for the student as stipulated by the current University regulations on the organisation and teaching and learning support of the educational process. In particular, this includes completing the safety briefing; attendance at course

classes is compulsory. For valid reasons (documentarily confirmed), theoretical training may, with the lecturer's approval, take place online. Successful completion of the course and the formation of professional competences and programme learning outcomes require preparation for each laboratory work (studying the theoretical material for the topic of the work), active participation during the class, thorough preparation of the report, defence of the results, participation in discussions regarding the constructive decisions made during the laboratory works, etc.
Students must meet the established deadlines for completing all types of academic work in accordance with the Working Programme of the course. A missed laboratory class must be completed within the deadline set by the lecturer, but no later than two weeks before the end of the theoretical classes in the semester.
The student's mastery of the theoretical material of the course is assessed through testing.
When performing laboratory work, the student must comply with the policy of academic integrity (cheating, plagiarism — including with the use of mobile devices — is prohibited). If a violation of academic integrity is detected in any type of academic work, the student receives an unsatisfactory grade and must re-do the task on the relevant topic (type of work) as stipulated by the Working Programme. Any form of academic dishonesty is unacceptable.
Within the framework of studying the course, students are provided with recognition and crediting of learning outcomes acquired through non-formal education, available on accessible platforms (https://prometheus.org.ua/, https://www.coursera.org/), which contribute to the formation of competences and the deepening of learning outcomes defined in the Working Programme of the course, or ensure the study of a relevant topic and/or type of work from the course syllabus (for more details, see the *Regulation on the Procedure for Recognition and Crediting of Learning Outcomes of Students at Khmelnytskyi National University*).

9. ASSESSMENT OF STUDENTS' LEARNING OUTCOMES DURING THE SEMESTER

Assessment of a student's academic achievements is carried out in accordance with the *Regulation on the Control and Assessment of Students' Learning Outcomes at Khmelnytskyi National University*. During the ongoing assessment of the work performed by the student for each structural unit and the results obtained, the lecturer awards a certain number of points as set out in the Working Programme for that type of work.
Each structural unit of academic work may be credited only if the student has scored at least 60 percent (the minimum level for a positive grade) of the maximum possible points assigned to that structural unit.
When assessing students' learning outcomes for any type of academic work (structural unit), it is recommended to use the generalised criteria provided below:

Table – Assessment Criteria for Student Learning Outcomes

Grade and Level of Achievement of Intended Learning Outcomes and Competences	General Description of Assessment Criteria
Excellent (High) 10	The student has mastered the content of the learning material deeply and comprehensively, easily navigates it, and skillfully uses the conceptual framework; can link theory to practice, solve practical problems, and confidently express and justify their opinions. An excellent grade assumes a logical presentation of the answer in the language of instruction (orally or in writing), demonstrates high-quality work design and proficiency in special devices, tools, and application programs. The student does not hesitate when the question is modified, can make detailed and generalizing conclusions, and demonstrates

	practical skills in solving professional tasks. Two or three minor inaccuracies were made in the response.
Very Good (Above Average) 9	The student has demonstrated deep and consistent mastery of the learning material, freely operates with the conceptual framework, and shows a high level of independent thinking. They confidently navigate the material, can connect theory with practice, and give logical, consistent, and well-reasoned answers, although one or two minor mistakes or inaccuracies may be present. The response is generally well-structured, and the written work is of high quality. The student shows initiative in solving practical tasks and sufficient flexibility of thinking under changed conditions of the task. They possess the necessary practical skills and can work independently without substantial support from the instructor.
Good (Average) 8	The student has mastered the learning material to the planned extent, generally possesses the conceptual framework, and is oriented in the studied topics but makes minor mistakes or shows an unclear understanding of certain provisions. The response is mostly based on learned examples and models and may contain some inaccuracies, template formulations, or repetitions. Theoretical knowledge is applied to solve typical practical tasks but without initiative or flexibility under changed conditions. The answers are based on independent thinking but have a lower depth of reasoning compared to higher levels. The written work may be of moderate quality, although its structure is preserved.
Satisfactory (Sufficient) 7	The student has mastered the basic curriculum material sufficiently to allow further learning and performance of typical practical tasks in the field. They possess basic concepts, although the answers contain inaccuracies or errors that can be identified and corrected with the help of guiding questions from the instructor. The response shows the ability for elementary analysis but is mostly reproductive thinking. The structure of the discipline is partially mastered, and orientation in the material is unstable; however, the answers are logically complete. The student demonstrates skills in performing simple tasks of typical scenarios and usually hesitates when answering modified or complex questions but tries to justify their opinion.
Satisfactory (Marginally Sufficient) 6	The student has demonstrated fragmented mastery of the basic curriculum material; their answers are mostly superficial, limited to simple reproduction of individual facts or definitions. They show a low level of orientation in the structure of the discipline and often hesitate even when answering typical questions. The answers contain significant mistakes, and the student does not always understand the meaning of concepts and is unable to independently correct deficiencies. Performance of practical tasks is possible only in the simplest cases and under constant instructor supervision. Nevertheless, the student has minimal but present signs of forming professional skills necessary for further study with appropriate support.

Unsatisfactory (Insufficient)	The student has demonstrated disjointed, unsystematic knowledge, is unable to distinguish between the essential and the secondary, makes mistakes in defining concepts, distorts their meaning, presents the material chaotically and insecurely, and cannot use knowledge to solve practical tasks. As a rule, an "unsatisfactory" grade is given to a student who cannot continue learning without additional work on the discipline.

Structuring of the Course by Types of Academic Work and Assessment of Student Learning Outcomes

In-Class Work								***Assessment Activities***	***Semester Final Assessment***
Laboratory Work №:								Test control:	Pass/Fail test
1	2	3	4	5	6	7	8	1	
Number of points per type of academic work (min–max)									
6-10	6-10	6-10	6-10	6-10	6-10	6-10	6-10	12-20	According to the rating
48-80								**12-20**	**60-100***

Notes: If the number of points earned for any type of academic work in the course is below the established minimum, the student receives a failing grade and must retake the work within the deadline set by the lecturer (or dean). The institutional grade is determined in accordance with the table "**Correspondence between the Institutional Grading Scale and the ECTS Grading Scale**".

Assessment of Laboratory Work Defence Results

A laboratory work completed and formatted in accordance with the requirements established in the Methodological Guidelines is comprehensively assessed by the lecturer during its defence based on the following criteria:

– independence and accuracy of execution;
– completeness of the answer and understanding of the principles of building machine learning models;
– ability to justify the choice of algorithm or method;
– correctness of model implementation in the Python programming environment using appropriate libraries;
– ability to interpret the results of modelling and evaluate their suitability for solving the given task.

When assessing a laboratory session, the lecturer uses the generalised criteria outlined in the table "Assessment Criteria for Student Learning Outcomes" (minimum passing score – 3 points, maximum – 5 points).

If the student demonstrates a knowledge level below 60 percent of the maximum score established in the Working Programme for each structural unit, the laboratory work is not credited. In such a case, the student must study the topic more thoroughly, review the methodology, correct major mistakes, and re-defend the work at the time set by the lecturer.

Assessment of Test-Based Control Results

The test included in the Working Programme consists of 30 test items, each carrying equal weight. According to the table for structuring types of academic work, the student may receive between 12 and 20 points depending on the number of correct answers.

The test duration is 30 minutes. Students complete the test online in the Modular Learning Environment. If a failing grade is received, the test must be retaken before the next scheduled assessment.

Distribution of Points Depending on Correct Answers to Test Questions (Scale 12–20)

Number of Correct Answers	**1–17**	**18–19**	**20–21**	**22–23**	**24**	**25**	**26**	**27**	**28**	**29–30**
Percentage of Correct Answers	0–59	60–66	67–73	74–79	80–83	84–86	87–89	90–92	93–95	96–100
Score	–	12	13	14	15	16	17	18	19	20

The final semester grade according to the institutional grading scale and the ECTS grading scale is determined automatically after the lecturer enters the assessment results in points for all types of academic work into the electronic gradebook. The correspondence between the institutional grading scale and the ECTS grading scale is provided in the table "Correspondence" below.
For each individual type of task in the final semester assessment, the assessment criteria for student learning outcomes provided above (see **Table – Assessment Criteria for Student Learning Outcomes**) are applied.
The final semester grade according to the institutional grading scale and the ECTS grading scale is determined automatically after the lecturer enters the assessment results in points for all types of academic work into the electronic gradebook. The correspondence between the institutional grading scale and the ECTS grading scale is shown below in the **Correspondence Table**.
The final examination grade is recorded if the total number of points accumulated by the student in the course as a result of ongoing assessment falls within the range of 60 to 100 points. In this case, a grade of *Excellent/Good/Satisfactory* is assigned according to the institutional scale, and a letter grade is assigned according to the ECTS scale, corresponding to the total number of points earned by the student as specified in the **Correspondence Table**.

Table – Correspondence between the Institutional Grading Scale and the ECTS Grading Scale

<table>
<tr><th rowspan="2">ECTS Grade</th><th rowspan="2">Rating Scale (Points)</th><th colspan="2">Institutional Grade(Level of Achievement of the Intended Learning Outcomes in the Course)</th></tr>
<tr><th>Pass/ Fail</th><th>Exam / Graded Credit</th></tr>
<tr><td>A</td><td>90-100</td><td rowspan="5">Pass</td><td>Excellent – a high level of achievement of the intended learning outcomes in the course, indicating the learner's full readiness for further study and/or professional activity in the field.</td></tr>
<tr><td>B</td><td>83-89</td><td rowspan="4">Good – an average (maximally sufficient) level of achievement of the intended learning outcomes in the course and readiness for further study and/or professional activity in the field.

Satisfactory – the student has demonstrated a minimally sufficient level of achievement of the learning outcomes required for further study and/or professional activity in the field.</td></tr>
<tr><td>C</td><td>73-82</td></tr>
<tr><td>D</td><td>66-72</td></tr>
<tr><td>E</td><td>60-65</td></tr>
<tr><td>FX</td><td>40-59</td><td>Fail</td><td>Fail – several intended learning outcomes in the course have not been achieved. The level of acquired learning outcomes is insufficient for further study and/or professional activity in the field.</td></tr>
</table>

F	0-39		**Fail** – no learning outcomes have been achieved.

10 SELF-ASSESSMENT QUESTIONS ON LEARNING OUTCOMES

1. Concept of program, software tool, software. Types of software by types of licenses.
2. Reasons and prerequisites for the emergence of the "Software Engineering" discipline. Definition of software engineering.
3. Software engineering as an engineering and scientific discipline .
4. Standardization in software engineering.
5. Main areas of SWEBOK knowledge .
6. Software life cycle.
7. General characteristics of the cascade model of the software life cycle; its advantages and disadvantages.
8. General characteristics of the spiral model of the software life cycle and its advantages and disadvantages.
9. General characteristics of the iterative model of the software life cycle and its advantages and disadvantages.
10. General characteristics of the incremental (step-by-step) software life cycle model and its advantages and disadvantages.
11. General characteristics of the rapid application development methodology RAD (Rapid Application Development) and its advantages and disadvantages.
12. RUP methodology (Rational Unified Process). General characteristics of the software development process.
13. MSF (Microsoft Solutions Framework) project team model .
14. MSF (Microsoft Solutions Framework) process model .
15. Loud software development methodologies. General characteristics.
16. Extreme Programming (XP).
17. Scrum methodology.
18. International and national standards of the software life cycle.
19. The essence and methods of structural analysis and modeling.
20. General characteristics of structural analysis and modeling methodologies.
21. Basic principles of structural analysis and modeling.
22. IDEF0 notation for graphical representation of the SADT model.
23. Syntax and semantics of DFD diagrams.
24. Basic concepts of "entity-relationship" information modeling (ERD) .
25. Overview of notations used in the construction of ER diagrams.
26. Methodology for building ER-diagrams.
27. The essence of object-oriented analysis and modeling; its advantages and disadvantages.
28. Basic principles of object-oriented analysis and modeling.
29. General characteristics of the UML language. UML diagrams.
30. The general process of developing software requirements.
31. Properties of software requirements.
32. Sources and strategies for identifying software requirements.
33. Analysis, systematization and verification of software requirements.
34. Specification and formalization of software requirements.
35. Software specification.
36. The concept of software architecture. Main classes of architectures.
37. Basic principles of software architecture design.
38. The modular structure of the software product. The main characteristics of the modules.
39. Types of software modules and their structure.
40. Top-down and bottom-up software design.

41. Development of the software structure using an object-oriented approach.
42. Principles, rules and stages of software interface design.
43. Programming paradigms.
44. Concept of programming language. Development stages of programming languages.
45. Classification of programming languages.
46. Tools and integrated programming environments. Translators.
47. Basics of software design. Standards in construction. Design management.
48. Classification of software errors. Software debugging methods. General software debugging technique.
49. The main types of testing.
50. Levels of testing (module, integration and system testing).
51. Tests. Classification of tests. Validation tests. Measurement of test results.
52. Features of object-oriented testing.
53. Peculiarities of testing object-oriented modules.
54. Software verification and attestation.
55. General principles of drafting and design of software documentation.
56. The main types of software documentation.
57. Characteristics of the documentation intended for the user of the software .
58. Electronic reference system.
59. Software configuration management processes.
60. Stages and procedures in software configuration management.
61. Technological support for software configuration management.
62. The main stages of software product readiness. Alpha version. Beta version. Release
63. Organization and methods of software maintenance. Stages and procedures for software maintenance.
64. Escort techniques. Reengineering, reverse reengineering.
65. Code of ethics for software engineering professionals (IEEE-CS/ACM) .
66. Corporate ethics and culture.

11. EDUCATIONAL AND METHODOLOGICAL SUPPORT

The educational process for the course *"*Software Engineering Basics*"* is supported with all necessary instructional and methodological materials, which are available in the Modular Learning Environment MOODLE:

1. Course *"*Software Engineering Basics*"*: https://msn.khmnu.edu.ua/course/view.php?id=9732
2. Methodological Guidelines for Laboratory Sessions: https://msn.khmnu.edu.ua/course/view.php?id=9732

12 RECOMMENDED LITERATURE

Primary

1. Rod Stephens. Beginning Software Engineering : Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada and the United Kingdom. 2022. 685 P.

2. Elvis C. Foster. Software Engineering. A Methodical Approach: Second Edition. Published 2022 by CRC Press, Taylor & Francis Group. Boca Raton-London-New York. 579 p.

3. Mr. Rohit Manglik, Principles of Software Engineering. (2024). EduGorilla Publication, 296 p.

4. O'Regan, Gerard. Concise Guide to Software Engineering: From Fundamentals to Application Methods. , Springer International Publishing, 2022, 444 p.

5. Bose, D. R., Roy, D. S., Sutradhar, M. S. (2024). Comprehensive Guide to Software Engineering: Principles, Processes, and Practices. Authors Click Publishing. 639 p.

Additional

6. Electronic versions of methodological instructions for laboratory works.
7. Ronald J. Leach. Introduction to Software Engineering: Second Edition. Howard University, Washington, DC, USA. CRC Press, Taylor & Francis Group. 2018. 420 p.
8. Automated design of information systems. URL: https://posibniki.com.ua/catalog-avtomatizovane-proektuvannya-informaciynih-sistem---denisova-oo.
9. Levus E. V., Marusenkova T. A. Introduction to software engineering: 1024 tasks for preparation for control measures. Lviv: Lviv Polytechnic, 2021. 188 p .
10. Martin Robert. Pure Agile: Back to Basics; trans. from English V. Lunenko. Kharkiv: «Ranok» Publishing House: Fabula, 2021. 224 p.
11. Software Engineering Body of Knowledge (SWEBOK). URL: https://www.computer.org/education/bodies-of-knowledge/software-engineering
12. Introduction to the Microsoft Solutions Framework. URL: https://learn.microsoft.com/en-us/previous-versions/tn-archive/bb497060 (v=technet.10)?redirectedfrom=MSDN
13. Manifesto for Agile Software Development. URL: http://agilemanifesto.org/
14. The Software Engineering Code of Ethics and Professional Practice. URL: https://ethics.acm.org/code-of-ethics/software-engineering-code/
15. ACM Code of Ethics and Professional Conduct. URL: https://ethics.acm.org/code-of-ethics/
16. DSTU ISO/IEC/IEEE 24765:2018 (ISO/IEC/IEEE 24765:2017, IDT). Systems and software engineering. Dictionary of terms.
17. DSTU ISO/IEC/IEEE 12207:2018. Systems and software engineering. Software life cycle processes (ISO/IEC/IEEE 12207:2017, IDT).
18. DSTU ISO/IEC TR 24774:2016 (ISO/IEC TR 24774:2010, IDT). Systems and software engineering. Life cycle management. Guidelines for describing the process.
19. DSTU ISO/IEC TS 24748-1:2018 (ISO/IEC TS 24748-1:2016, IDT). Systems and software engineering. Life cycle management. Part 1. Guidelines for life cycle management.
20. DSTU ISO/IEC TR 24748-3:2016 (ISO/IEC TR 24748-3:2011, IDT). Systems and software engineering. Life cycle management. Part 3. Guidance on the application of ISO/IEC 12207 (Software life cycle processes).
21. ISO/IEC/IEEE 29148-2018. Systems and software engineering – Life cycle processes – Requirements engineering.
22. DSTU ISO/IEC TR 24766:2016 (ISO/IEC 24766:2009, IDT). Information Technology. Systems and software engineering. Guidelines for the development of technical requirements for software tools.
23. DSTU ISO/IEC/IEEE 42010:2018 (ISO/IEC/IEEE 42010:2011, IDT). Systems and software engineering. Description of the architecture.
24. DSTU ISO/IEC/IEEE 15289:2019 (ISO/IEC/IEEE 15289:2017, IDT). Systems and software engineering. Content of life cycle information objects (documentation).
25. DSTU ISO/IEC/IEEE 26515:2018 (ISO/IEC/IEEE 26515:2011, IDT). Systems and software engineering. Development of user documentation in a flexible environment.
26. Borodkina I., Borodkin G. Software engineering: manual for students higher education institutions. K.: Publishing House "Center for Educational Literature", 2018. 204 p.
27. Levus E., Melnyk N.. Introduction to software engineering. L.: Lviv Polytechnic, 2018. 248 p.
28. Business process modeling and IT project management: training. manual / E. M. Kryzhanovskyi, A. R. Yascholt, S. O. Zhukov, O. M. Kozachko. Vinnytsia: VNTU, 2018. 91 p.

29. Postil S. D. UML. Unified language of modeling of information systems: training. Manual. Irpin: UDFSU, 2019. 322 p.

30. Праворська Н.І., Яшина О.М., Нетреба І.В. Доміна А.Р. Кириченко О.М. Метод конструювання програмного забезпечення згідно аналізу помилок SQL-запитів. Вісник Хмельницького національного університету, серія Технічні науки. №3, 2023 (321) – с. 302-307

13. INFORMATION RESOURCES

1. Electronic Library of the University. [Electronic resource]. – Access: http://library.khmnu.edu.ua/
2. Institutional Repository of Khmelnytskyi National University. [Electronic resource]. – Access: http://elar.khmnu.edu.ua/jspui/?locale=uk
3. Modular Learning Environment. [Electronic resource]. – Access: https://msn.khmnu.edu.ua/