



COMPONENTS OF SOFTWARE ENGINEERING. PART 4. SOFTWARE QUALITY AND TESTING

The working program of the academic discipline (Syllabus)

Details of the academic discipline

Level of higher education *First (Bachelor)*

Branch of knowledge	<i>12 Information Technologies</i>
Specialty	<i>121 Software engineering</i>
Educational program	<i>Software Engineering of Multimedia and Information Retrieval Systems</i>
Discipline status	<i>Normative</i>
Form of education	<i>Daytime</i>
Year of training, semester	<i>3 rd year of training, 5th semester</i>
Scope of the discipline	<i>Lectures: 36 hours, computer workshop: 18 hours, self-study: 66 hours.</i>
Semester control/ control measures	<i>Examination, modular control work, calendar control</i>
Lessons schedule	<i>According to the schedule for the autumn semester of the current academic year (http://roz.kpi.ua/)</i>
Language of teaching	<i>English</i>
Information about the head of the course/teachers	<i>Lecturer: PhD, associate professor, Kostiantyn O. Tkachenko Computer Workshop: PhD, associate professor, Kostiantyn O. Tkachenko e-mail: tkachenko.kostiantyn@pzks.fpm.kpi.ua</i>
Placement of the course	<i>Google Classroom access is granted to registered students.</i>

Program of educational discipline

1. Description of the educational discipline, its purpose, subject of study and learning outcomes

Studying the discipline "Components of Software Engineering. Part 4. Software Quality and Testing" allows students to develop the competencies necessary for solving practical problems of professional activity related to software quality assessment and testing.

***The purpose of** studying the discipline "Components of Software Engineering. Part 4. Software Quality and Testing" is the formation of students' abilities to independently assess the quality and conduct testing of the developed software.*

***The subject of** the discipline is "Components of Software Engineering. Part 4. Software Quality and Testing" are the processes, principles and techniques used for software quality analysis and testing.*

*Studying the discipline "Components of Software Engineering. Part 4. Software Quality and Testing" forms **general competencies (GC)** necessary for solving practical tasks of professional activity related to quality assessment and testing of developed software:*

***GC02** Aptitude for abstract thinking, analysis and synthesis*

*Studying the discipline "Components of Software Engineering. Part 4. Software Quality and Testing" forms **speciality professional competencies (PC)** necessary for solving practical tasks of professional activity related to quality assessment and testing of developed software:*

***PC01** Ability to identify, classify and formulate software requirements.*

PC02 Ability to participate in software design, including its structure, behavior and functioning processes modeling (formal description).

PC03 Ability to develop software systems architectures, modules and components.

PC04 Ability to formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards.

PC05. Ability to follow specifications, standards, rules and recommendations in the professional field during the life cycle processes implementation.

PC07 Knowledge of information data models, the ability to create software for data storage, retrieval and processing.

PC10 Ability to accumulate, process and systematize professional knowledge about software creation and maintenance, and determination of the importance of lifelong learning.

PC11. Ability to implement phases and iterations of the life cycle of the software systems and information technology based on appropriate models and approaches to software development.

PC12 Ability to carry out the system integration process, apply change management standards and procedures to maintain software integrity, overall functionality and reliability.

PC13 Ability to reasonably select and master software development and maintenance tools.

PC16 Ability to develop software of information retrieval systems.

PC17 Ability to develop software of multimedia systems.

PC19 Ability to identify, analyze, and document software requirements for multimedia and information retrieval systems.

PC20 Ability to create innovative start-up projects, calculate the main technical and economic indicators and develop business models of innovative start-up projects for the development of software for multimedia and information retrieval systems that have commercial potential for investment.

Program learning outcomes (PLO) of the discipline "Components of Software Engineering. Part 4. Software Quality and Testing" according to the educational program:

PLO01 To analyze, purposefully search and select the necessary information and reference resources and knowledge to solve professional problems, taking into the account modern advances in science and technology.

PLO03 To know the software life cycle basic processes, phases and iterations.

PLO04 To know and apply professional standards and other regulatory documents in the field of software engineering.

PLO06 To know and apply relevant mathematical concepts, domain methods, system and object-oriented analysis and mathematical modeling for software development.

PLO07 To know and to apply in practice the fundamental concepts, paradigms and basic principles of the functioning of language, instrumental and computational tools of software engineering.

PLO08 To know and to be able to develop a human-machine interface.

PLO09 To be able to use collecting, formulating and analyzing software requirements methods and tools.

PLO10 To conduct a pre-project survey of the subject area, system analysis of the design object.

PLO11 To select initial data for design, guided by formal methods of describing requirements and modeling.

PLO12 To apply efficient approaches to software design in practice.

PLO13 To know and apply methods of developing algorithms, designing software and data and knowledge structures.

PLO14 To apply in practice instrumental software tools for domain analysis, design, testing, visualization, measurement and documentation of software.

PLO15 To choose programming languages and development technologies to solve the problems of creating and maintaining software.

PLO16 To have the software development, design approval and all types of software documentation release skills.

PLO17 To be able to apply methods of component software development.

PLO18 To know and be able to apply information technology of processing, storage and transmission of data.

PLO19 To know and be able to apply software verification and validation methods.

PLO20 To know approaches to evaluation and quality assurance of software.

PLO22 To know and be able to apply project management methods and tools.

PLO23 To be able to document and present the software development results.

PLO28 To be able to develop business models and create innovative start-up projects for the development of software for multimedia and information retrieval systems, which have commercial potential for investment

PLO29 To know and be able to manage software creation and implementation projects according to SWEBOK, PMBOK, BPMCBOK international standards.

2. Pre-requisites and post-requisites of the discipline (place in the structural and logical scheme of training according to the relevant educational program)

The successful study of the discipline "Components of Software Engineering. Part 4. Software Quality and Testing" precedes the study of the disciplines "Components of Software Engineering. Part 1. Introduction to Software Engineering", "Components of Software Engineering. Part 2. Software Modeling. Analysis of Software Requirements", "Components of Software Engineering. Part 3. Software Architecture" of the curriculum for bachelor's training in the specialty 121 Software engineering.

Received during the assimilation of the discipline "Software engineering components. Part 4. Quality and testing of software" theoretical knowledge and practical skills ensure mastery of the discipline "Software of information and search systems", successful completion of pre-diploma practice, completion of diploma projects in the specialty 121 Software engineering.

3. Content of the academic discipline

Discipline "Components of Software Engineering. Part 4. Software Quality and Testing" involves the study of the following topics:

Topic 1. Fundamentals of software testing

Topic 2. Organization of the software testing process

Topic 3. Evaluation of software quality

Topic 4. Means of optimization of the testing and quality control process

Exam

4. Educational materials and resources

Basic References:

1. The materials of the discipline "Components of Software Engineering. Part 4. Software Quality and Testing" are in Google classroom, access is granted to registered students.

2. Pezze M., Young M. *Software Testing and Analysis: Process, Principles and Techniques*. John Wiley & Sons Inc., 2008. 488 p.

3. Myers G.J., Badgett T., Sandler C. *The art of software testing: Third Edition*. John Wiley & Sons Inc., 2012. 254 p.

4. Winter A., Meaney R. *Team guide to software testability*. Conflux, 2021. 178 p.

5. Sandwich Testing. [Electronic resource]. URL: <https://www.geeksforgeeks.org/sandwich-testing-software-testing/>

6. Automation Testing Tutorial: What is Automated Testing? [Electronic resource]. URL: <https://www.guru99.com/automation-testing.html>

Additional References:

1. Software Testing – Documentation. [Electronic resource]. URL: https://www.tutorialspoint.com/software_testing/software_testing_documentation.htm

2. Test Deliverables in Software Testing – Detailed Explanation. [Electronic resource]. URL: <https://www.softwaretestingmaterial.com/test-deliverables/>

3. What is Software Testing Life Cycle (STLC)? [Electronic resource]. URL:

Educational content

5. Methods of mastering an educational discipline (educational component)

No	Type of training session	Description of the training session
<i>Topic 1. Introduction to search engines and services</i>		
1	<i>Lecture 1. Introduction to software testing</i>	<i>Definition of software testing. Purpose and goals of software testing. Types of software testing. The software development life cycle and the role of testing at each stage. The importance of software testing. Advantages of software testing. Task on self-study No. 1.</i>
2	<i>Lecture 2. Inspection in testing. Basic concepts and characteristics of inspection.</i>	<i>Definition of Software Inspection. Benefits of Software Inspection. Inspection process and methodology. Roles and responsibilities in Software Inspection. Types of software inspection. Tools and techniques used for Software Inspection Task on self-study No. 2.</i>
3	<i>Computer workshop 1. Software inspection and analysis (Part 1)</i>	<i>Task: Using object-oriented programming software to develop a module for software inspection Tasks on self-study No. 3.</i>
4	<i>Lecture 3. Static analysis. Basic concepts and characteristics, means of static analysis.</i>	<i>Definition of static analysis. Advantages of static analysis. Types of static analysis (source code analysis, data flow analysis, etc.). Tools and methods used in static analysis. Integration of static analysis into the software development life cycle. Limitations of static analysis. Task on self-study No. 4.</i>
5	<i>Lecture 4. Modular testing. Tasks, input data, main testing steps, unit testing completion conditions.</i>	<i>Definition of unit testing. Advantages of unit testing. Module testing process and methodology. Writing practical unit tests. Test Driven Development (TDD). Tools and techniques used in unit testing. Task on self-study No. 5.</i>
6	<i>Computer workshop 1. Software inspection and analysis (Part 2)</i>	<i>Task: Using software to implement a module for implementing static analysis of software Tasks on self-study No. 6</i>
7	<i>Lecture 5. Integration testing and system testing.</i>	<i>Definition of integration and system testing. Advantages of integration and system testing. Process and methodology of integration testing. System testing process and procedure. Types of integration and system testing. Tools and methods used in integration and system testing. Task on self-study No. 7.</i>
8	<i>Lecture 6. Regression testing, basic concepts and characteristics, key stages.</i>	<i>Definition of regression testing. Advantages of regression testing. Process and methodology of regression testing. Types of regression testing (manual and automatic regression testing). Strategies for effective regression testing. Tools and techniques used in regression testing. Task on self-study No. 8.</i>
9	<i>Computer workshop 1. Software inspection and analysis (Part 3)</i>	<i>Task: Using software to develop a module for testing, the types of which are discussed in the lectures. Tasks on self-study No. 9</i>
10	<i>Lecture 7. Functional testing.</i>	<i>Definition of functional testing. Advantages of functional testing. Process and methodology of functional testing. Types of functional testing (black box testing, white box testing, etc.). Writing practical functional tests. Tools and techniques used in functional testing. Task on self-study No. 10</i>

11	Computer workshop 2. Software testing (Part 1)	Task: Using software to develop a module for implementing functional testing. Tasks on self-study No. 11
12	Lecture 8. Structural testing	Definition of structural testing. Advantages of structural testing. Process and methodology of structural testing. Types of structural tests (branch testing, state testing, etc.). Writing practical structural tests. Tools and methods used in structural testing. Task on self-study No. 12.
13	Lecture 9 . Testing object-oriented software	Definition of object-oriented software. Strategies for testing object-oriented software (imitation, polymorphism, encapsulation). Writing practical tests for object-oriented software. Tools and methods used in testing object-oriented software. Task on self-study No. 13.
14	Computer workshop 2. Software testing (Part 2)	Task: With the help of software tools, develop a module for the implementation of structural testing Tasks on self-study No. 14
<i>Topic 2. Organization of the software testing process</i>		
15	Lecture 10. Choosing a test case	Determination of test case selection. The process and methodology of selecting test cases. Factors to consider when selecting test cases (coverage, risk, etc.). Types of test case selection methods (equivalence distribution, analysis of boundary values, etc.). Tools and techniques used to select test cases. Task on self-study No. 15.
16	Lecture 11. Performance testing	Definition of performance testing. Benefits of performance testing. Performance testing process and methodology. Types of performance testing (load testing, stress testing, etc.). Performance testing criteria (response time, throughput, etc.). Tools and techniques used in performance testing Task on self-study No. 16.
17	Lecture 12. Security testing	Definition of security testing. Benefits of security testing. Security testing process and methodology. Types of security testing (penetration testing, vulnerability scanning, etc.). Security check criteria (confidentiality, integrity, availability). Tools and techniques used in security testing. Task on self-study No. 17.
18	Lecture 13. Testing web applications	Definition of web application testing. Benefits of web application testing. The process and methodology of testing web applications. Writing practical tests for web applications. Tools and techniques used to test web applications. Task on self-study No. 18.
19	Computer workshop 2. Organization of the software testing process (Part 3)	Task: With the help of software tools, develop a module for testing web applications. Task on self-study No. 19.
<i>Topic 3. Evaluation of software quality</i>		

20	Lecture 14. Graphical User Interface (GUI) Testing	Definition of GUI testing. GUI testing process and methodology. GUI testing criteria (usability, accessibility, consistency, etc.). Tools and techniques used for GUI testing. Task on self-study No. 20.
21	Lecture 15. Usability testing	Definition, benefits, process and methodology of usability testing. Types (expert review, heuristic evaluation, etc.). Usability testing criteria. Tools and methods used for usability testing. Task on self-study No. 21.
22	Computer workshop 3. Software quality assessment and the use of JUnit for software testing. (Part 1)	Task: With the help of software tools, implement a module that performs software testing Task on self-study No. 22.
23	Lecture 16. Error-based testing	Definition of error-based testing. Advantages of error-based testing. Error-based testing process and methodology. Types of error-based testing (fault injection, mutation testing, etc.). Fault-based testing criteria (detection rate, false alarm rate, etc.). Tools and techniques used in error-based testing. Task on self-study No. 18.
<i>Topic 4. Means of optimization of the testing and quality control process</i>		
24	Lecture 17. Automation and testing tools	Defining test automation and tools. Advantages of test automation. Test automation process and methodology. Types of test automation. Criteria for choosing test automation tools (coverage, reliability, etc.). Means and methods used for test automation. Task on self-study No. 24.
25	Computer workshop 3. Software quality assessment and the use of JUnit for software testing. (Part 2)	Task: With the help of software tools, implement a module that performs software testing Task on self-study No. 25.
26	Lecture 18. Planning and monitoring of the software quality process	Defining the process of software quality planning and monitoring. Planning and monitoring process and methodology. Types of software quality indicators (reliability, efficiency, etc.). Methods of improving the software quality process (Six Sigma, Total Quality Management, etc.). Tools and techniques used to plan and monitor the software quality process. Task on self-study No. 26.
27	Computer workshop 3. Software quality assessment and the use of JUnit for software testing. (Part 3)	Task: With the help of software tools, implement a module that performs software testing Task on self-study No. 27.
<i>Modular control work</i>		

6. Independent work of a student/graduate student

The discipline "Components of Software Engineering. Part 4. Software Quality and Testing" is based on independent preparations for classroom classes on theoretical and practical topics.

No. z/p	The name of the topic submitted for independent processing	Number of hours	References
---------	--	-----------------	------------

1	<i>Preparation for lecture 1</i>	1	<i>Basic: 1-3 Additional: 1, 3</i>
2	<i>Preparation for lecture 2</i>	1	<i>Basic: 1-3 Additional: 1, 3</i>
3	<i>Preparation for the computer workshop 1 (Part 1)</i>	1	<i>Basic: 1-3 Additional: 1, 3</i>
4	<i>Preparation for lecture 3</i>	1	<i>Basic: 1-3 Additional: 1, 3</i>
5	<i>Preparation for lecture 4</i>	1	<i>Basic: 1-3 Additional: 1, 3</i>
6	<i>Preparation for the computer workshop 1 (Part 2)</i>	1	<i>Basic: 1-3 Additional: 1, 3</i>
7	<i>Preparation for lecture 5</i>	1	<i>Basic: 1-4 Additional: 1-3</i>
8	<i>Preparation for lecture 6</i>	1	<i>Basic: 1-4 Additional: 1-3</i>
9	<i>Preparation for the computer workshop 1 (Part 3)</i>	1	<i>Basic: 1-4 Additional: 1-3</i>
10	<i>Preparation for lecture 7</i>	1	<i>Basic: 1-3, 5, 6 Additional: 1-3</i>
11	<i>Preparation for the computer workshop 2 (Part 1)</i>	1	<i>Basic: 1-3, 5, 6 Additional: 1-3</i>
12	<i>Preparation for lecture 8</i>	1	<i>Basic: 1-3, 5, 6 Additional: 1-3</i>
13	<i>Preparation for lecture 9</i>	1	<i>Basic: 1-3, 5, 6 Additional: 1-3</i>
14	<i>Preparation for the computer workshop 2 (Part 2)</i>	1	<i>Basic: 1-3, 5, 6 Additional: 1-3</i>
15	<i>Preparation for lecture 10</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
16	<i>Preparation for lecture 11</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
17	<i>Preparation for lecture 12</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
18	<i>Preparation for lecture 13</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
19	<i>Preparation for the computer workshop 2 (Part 3)</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
20	<i>Preparation for lecture 14</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
21	<i>Preparation for lecture 15</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
22	<i>Preparation for the computer workshop 3 (Part 1)</i>	1	<i>Basic: 1-6 Additional: 1-3</i>
23	<i>Preparation for lecture 16</i>	1	<i>Basic: 1, 4, 6 Additional: 1-3</i>
24	<i>Preparation for lecture 17</i>	1	<i>Basic: 1, 4, 6 Additional: 1-3</i>
25	<i>Preparation for the computer workshop 3 (Part 2)</i>	1	<i>Basic: 1, 4, 6 Additional: 1-3</i>

26	Preparation for lecture 18	1	Basic: 1-6 Additional: 1-3
27	Preparation for the computer workshop 3 (Part 3)	1	Basic: 1-6 Additional: 1-3
28	Preparation for modular control work	9	Basic: 1-6 Additional: 1-3
29	Preparation for the exam	30	Basic: 1-6 Additional: 1-3

Policy and control

7. The policy of academic discipline (educational component)

The policy and principles of academic integrity are defined in Chapter 3 of the Code of Honor of the National Technical University of Ukraine "Ihor Sikorsky Kyiv Polytechnic Institute". More details: <https://kpi.ua/code>.

8. Types of control and rating system for evaluating learning outcomes (RSO)

During the semester, students perform three computer workshops. **The maximum number of points for each computer workshop: 10 points.**

Points are awarded for:

- the quality of performance of the computer workshop: 0-6 points;
- answer during the defence of the computer workshop: 0-2 points;
- timely presentation of work for defence: 0-2 points.

Performance evaluation criteria:

- 6 points - the work is done qualitatively, in full;
- 3-5 points - the work is done qualitatively, in full, but has shortcomings;
- 1-2 points - the work is completed in full but contains significant errors;
- 0 points - the work is not completed in full.

Answer evaluation criteria:

- 2 points - the answer is complete, and well-argued;
- 1 point - there are significant errors in the answer;
- 0 points - there is no answer, or the answer is incorrect.

Criteria for evaluating the timeliness of work submission for defence:

- 2 points - the work is presented for defence no later than the specified deadline;
- 0 points - the work is submitted for defence later than the specified deadline.

The maximum number of points for performing and defending computer practicals:

10 points x 3 comp. practice = 30 points.

During the semester, students complete a test. The task consists of 3 theoretical and 2 practical questions. The answer to each question is evaluated by 4 points.

Evaluation criteria for each test question:

- 3 points - the answer is correct, complete, and well-argued;
- 1-3 points - in general, the answer is correct but has flaws;
- 1 point - there are significant errors in the answer;
- 0 points - there is no answer, or the answer is incorrect.

The maximum number of points for a modular control work:

4 points x 5 questions = 20 points.

The rating scale for the discipline is equal to the following:

$R = R_C = R_{com.workshop} + R_{MKR} + R_{exam} = 30 \text{ points} + 20 \text{ points} + 50 \text{ points} = 100 \text{ points}.$

Calendar control: is conducted twice a semester as a monitoring of the current state of fulfilment of the

syllabus requirements.

At the first certification (8th week), the student receives "credited" if his current rating is at least 10 points (50% of the maximum number of points a student can receive before the first certification).

At the second certification (14th week), the student receives "passed" if his current rating is at least 20 points (50% of the maximum number of points a student can receive before the second certification).

Semester control: exam

Conditions for admission to semester control:

With a semester rating (R_c) of at least 30 points and the admission of all the work of the computer workshop, the student is admitted to the exam. After passing the exam, a grade is assigned according to the table (Table of correspondence of rating points to grades on the university scale).

Completion and defence of a computer workshop is a necessary condition for admission to the exam.

Table of correspondence of rating points to grades on the university scale :

<i>Scores</i>	<i>Rating</i>
100-95	Perfectly
94-85	Very good
84-75	Fine
74-65	Satisfactorily
64-60	Enough
Less than 60	Unsatisfactorily
Admission conditions not met	Not allowed

9. Additional information on the discipline (educational component)

The list of questions submitted for semester control is given in Appendix 1.

The working program of the academic discipline (syllabus):

Compiled by PhD, associate professor, Kostiantyn O. Tkachenko

Adopted by Computer Systems Software Department (protocol № 14 from 15.05.24)

Approved by the Faculty Board of Methodology (protocol № 12 from 21.06.24)

Appendix 1. List of issues that are submitted to the semester control

- 1. What is software testing and why is it an important stage of software development?*
- 2. What is the difference between Inspection testing and static testing?*
- 3. What is unit testing, and how does it differ from integration and system testing?*
- 4. What is regression testing, and why is it performed?*
- 5. What is functional testing, and how is it performed?*
- 6. What is structural testing, and what methods are used?*
- 7. Selection of test cases for the software system.*
- 8. How does object-oriented software testing differ from other types of software?*
- 9. What is performance testing, and how is it performed?*
- 10. What is security testing, and what methods are used?*
- 11. What is web application testing, and what methods are used?*
- 12. What is Graphical User Interface (GUI) testing, and what methods are used?*
- 13. What is usability testing, and what methods are used?*
- 14. What is error-based testing, and what methods are used?*
- 15. What is test automation, and what tools are used?*
- 16. What is the role of planning and monitoring in the software quality process?*
- 17. Software quality indicators.*
- 18. What is the difference between unit testing and integration testing?*
- 19. What is the purpose of regression testing, and how is it performed?*
- 20. What is the difference between security testing and performance testing?*