National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"

Department of Computer Engineering

Software Engineering Components. Part 4. Software quality and testing.

The program of the academic discipline (Syllabus)

Details of the academic discipline	
Cycle of Higher Education	***First cycle of higher education (Bachelor's degree)***
Field of Study	*12 Information technologies*
Specialty	*121 Software Engineering*
Education Program	*Computer Systems Software Engineering*
Type of Course	*Normative*
Mode of Studies	*Full-time education*
Year of studies, semester	*3 year (5 semester)*
ECTS workload	*4 credits*
Testing and assessment	*Exam*
Course Schedule	*Lectures 18 (36 hours), Laboratory9 (18hours)*
Language of Instruction	*English*
Course Instructors	Lecturer: Prof., Dr.Sc. Bogdan Korniyenko,b.korniyenko@kpi.ua Laboratory: Prof., Dr.Sc. Bogdan Korniyenko,b.korniyenko@kpi.ua
Access to the course	https://classroom.google.com

Program of academic discipline

1 Course description, goals and objectives, and learning outcomes

Studying the discipline "Software Engineering Components. Part 4. Software quality and testing" occupies an important place in the structure of knowledge acquisition in the educational program "Computer systems software engineering". The main task of the discipline is the formation of a set of knowledge about the methods of manual software testing, the features of system, module and integration testing, models for evaluating the degree of testing of a software product, and the ability to evaluate the complexity of a software product using a mathematical model, to use methods of manual and automated software testing, to create a set of tests for testing simple and complex systems.

The purpose of studying the discipline "Software Engineering Components. Part 4. Software quality and testing" is theoretical and practical training of students, which should ensure that they acquire basic knowledge of methods of manual software testing, features of system, module and integration testing, models for evaluating the degree of testing of a software product, and the ability to evaluate the complexity of a software product using a mathematical model, use methods manual and automated software testing, create a set of tests for testing simple and complex systems.

The subject of study of the discipline is modern methods, tools and technologies of software development.

According to the requirements of the EP, the discipline "Software Engineering Components" should ensure that applicants acquire competencies and program learning outcomes: PC01-05, PC07, PC08, PC11-13, PLO01-04, PLO06-11, PLO13-20. In particular, after mastering the module "Software Engineering Components. Part 4. Software quality and testing" must demonstrate the following competencies and program learning outcomes:

- the ability to use instrumental means of designing and creating information technology systems, products and services;
- the ability to design and evaluate software, the ability to choose the optimal set of operating system tools and software tools for developing distributed applications, strategies for using cloud technologies;
- the ability to build effective computing algorithms, to justify methods of designing and using software, to determine the effectiveness of programs through testing, to carry out documentation in compliance with norms and standards;
- the ability to use the capabilities of software, operating systems, computer networks to build the architecture of information systems based on the interaction of various software platforms in distributed corporate information systems.

According to the results of studying the educational discipline "Software Engineering Components. Part 4. Software quality and testing", the following **knowledge** should be obtained:

- apply modern technologies and tools for the development of software systems at all stages of the life cycle;
- explain the difference between different programming paradigms, characterize types of programming, classify software development methods;
- identify, classify and formulate software requirements;
- formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards;
- the ability to evaluate and take into account economic, social, technological and environmental factors affecting the field of professional activity.

Skills that should be acquired as part of studying the academic discipline "Software Engineering Components. Part 4. Software quality and testing":

- to analyze, purposefully search for and choose information and reference resources and knowledge necessary for solving professional tasks, taking into account modern achievements of science and technology;
- know the code of professional ethics, understand the social significance and cultural aspects of software engineering and adhere to them in professional activities;
- know and apply professional standards and other legal documents in the field of software engineering;
- know and be able to apply software verification and validation methods;
- know approaches to evaluation and quality assurance of software.

Such a combination of general and special competences, theoretical and practical knowledge, skills and abilities helps to increase the professional level of bachelor's degree holders in order to carry out effective activities in the field of development of software engineering.

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

Necessary disciplines: "Algorithms and data structures", "Fundamentals of programming", "Software Engineering Components. Part 1. Introduction to software engineering", "Software

Engineering Components. Part 2. Software modeling. Analysis of requirements for software", "Software Engineering Components. Part 3. Software architecture", “Object-oriented programming”.

Module "Software Engineering Components. Part 4. Software quality and testing " is necessary for coursework of the Software Engineering Components, and can also be useful when studying the disciplines "Methodologies and technologies of software development", "Risk management and project quality".

3 Structure of the credit module

A list of the main topics included in the study program of the discipline "Software Engineering Components. Part 4. Software quality and testing":

Section1. Fundamentals of software quality assessment.

Topic1.1. Basic concepts of software testing.

Topic1.2. The concept of software quality assessment and testing, the purpose of testing.

Topic1.3. Levels of testing.

Topic1.4. Types, types of testing.

Topic1.5. Principles of testing.

Section2. Software testing and development processes.

Topic2.1. Software development models.

Topic2.2. Life cycle of testing.

Topic2.3. Types and areas of testing.

Topic2.4. Testing classifications.

Section3. Development of test cases using checklists.

Topic3.1. The concept of a check list.

Topic3.2. Test case and its life cycle.

Topic3.3. Test management tools.

Topic3.4. Properties of quality test cases.

Section4. Defect reports.

Topic4.1. Terminology: errors, defects, failures, failures.

Topic4.2. Defect report and its lifecycle.

Topic4.3. Defect report management tools.

Topic4.4. Properties of quality defect reports.

Topic4.5. The logic of creating effective defect reports.

Section5. Testing techniques.

Topic5.1. Positive and negative test cases.

Topic5.2. Equivalence classes and boundary conditions.

Topic5.3. Domain testing and combination of parameters.

Topic5.4. Pair testing and finding solutions.

Topic5.5. Exploratory testing.

4 Educational resources and materials

Basic:

1. Pressman, Roger (2010) *Software Engineering: A Practitioner's Approach*, McGraw Hill, New York, NY.
2. Sommerville, Ian (2011) *Software Engineering*, Addison-Wesley, Boston, MA.

3. Stephens, Rod (2015) *Beginning Software Engineering*, Wrox.
4. Tsui, Frank , Orlando Karam and Barbara Bernal (2013) *Essentials of Software Engineering*, Jones & Bartlett Learning , Sudbury, MA.
5. Pfleeger, Shari (2001) *Sofwtare Engineering: Theory and Practice*, Prentice Hall, Upper Saddle River, NJ.

Supplementary:

1 Larman, C. (2005) *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and iterative Development*, Pearson

2 Ambler, S. (2002) *Agile Modeling: Effective Practices for Extreme Programming and the Unified Process*, New York, John Wiley & Sons.

3 Bass, D.L., Clements, D.P. and Kazman, D.R. (2012) *Software Architecture in Practice*, 3rd edn, Upper Saddle River, NJ, Addison Wesley

4 Beck, K. (2004) *Extreme Programming Explained: Embrace Change*, Upper Saddle River, NJ, Addison Wesley

5 Clemens Szyperski (2002) Component Software: Beyond object-oriented programming, Addison-Wesley

6 John Cheesman & John Daniels (2000) UML Components: A simple process for specifying component-based software (The component software series) Addison-Wesley

7 Rob Pooley, Perdita Stevens (2006) *Using UML Software Engineering with Objects and Components*, second edition. Addison-Wesley

8 Christopher Fox (2006) *Introduction to Software Engineering Design*. Addison Wesley

Educational content

5 Methodology

Sections and topics	Hours			
	Total	including		
		Lectures	Practical work	Self-study
Section1. Fundamentals of software quality assessment. *Topic 1.1. Basic concepts of software testing.* *Topic 1.2. The concept of software quality assessment and testing, the purpose of testing.* *Topic 1.3. Levels of testing.* *Topic 1.4. Types, types of testing.* *Topic 1.5. Principles of testing.*	18	4	4	10
Section2. Software testing and development processes. *Topic 2.1. Software development models.* *Topic 2.2. Life cycle of testing.* *Topic 2.3. Types and areas of testing.* *Topic 2.4. Testing classifications.*	26	6	4	16

Section3. Development of test cases using checklists. *Topic 3.1. The concept of a check list.* *Topic 3.2. Test case and its life cycle.* *Topic 3.3. Test management tools.* *Topic 3.4. Properties of quality test cases.*	28	8	4	16
Section4. Defect reports. *Topic 4.1. Terminology: errors, defects, failures, failures.* *Topic 4.2. Defect report and its lifecycle.* *Topic 4.3. Defect report management tools.* *Topic 4.4. Properties of quality defect reports.* *Topic 4.5. The logic of creating effective defect reports.*	24	6	4	14
Section5. Testing techniques. *Topic 5.1. Positive and negative test cases.* *Topic 5.2. Equivalence classes and boundary conditions.* *Topic 5.3. Domain testing and combination of parameters.* *Topic 5.4. Pair testing and finding solutions.* *Topic 5.5. Exploratory testing.*	26	12	4	10
Total hours in semester	120	36	20	64

Laboratory works:

The purpose of conducting laboratory classes is for students to consolidate theoretical knowledge and acquire the necessary practical skills for working with modern technologies for software engineering.

- Laboratory work #1: Software testing: development of tests;
- Laboratory work #2: Documentation of test results;
- Laboratory work #3: Terms of reference for the project;
- Laboratory work #4: Information systems reengineering;
- Laboratory work # 5: Project conveyor.

6 Self-study

- preparation for lectures by studying the previous lecture material;
- preparation for laboratory work with the study of the theory of laboratory work with an oral answer to the given questions of the section;
- preparation of results of laboratory work in the form of a protocol.

Attendance Policy and Assessment

7 Attendance Policy

During classes in an academic discipline, students must adhere to certain disciplinary rules:
- extraneous conversations or other noise that interferes with classes are not allowed;
- the use of mobile phones and other technical means is not allowed without the teacher's permission.

Laboratory works are submitted personally with a preliminary check of theoretical knowledge, which is necessary for the performance of laboratory work. Validation of practical results includes code review and execution of test tasks.

8 Monitoring and grading policy

Current control: survey on the subject of the lesson
Calendar control: conducted twice a semester as a monitoring of the current status of meeting the syllabus requirements.
Semester control: exam
Conditions for admission to semester control: enrollment of all laboratory work

System of rating points and evaluation criteria

The student's rating in the discipline consists of the points he receives for:
1. performance and defense of 5 laboratory works;
2. execution of 2 modular control works (MCW).

Laboratory works:
"excellent", a complete answer to the questions during the defense (at least 90% of the required information) and a properly prepared protocol for laboratory work - 6 points;
"good", a sufficiently complete answer to the questions during the defense (at least 75% of the required information) and a properly prepared protocol for laboratory work - 5 points;
"satisfactory", incomplete answer to the questions during the defense (at least 60% of the required information), minor errors and a properly prepared protocol for laboratory work - 4 points;
"unsatisfactory", an unsatisfactory answer and/or an improperly prepared protocol for laboratory work - 0 points.

Modular Control Works:
"Excellent", full answer (not less than 90% of the information you need) - 10 points;
"Good", a full answer (not less than 75% of the information you need), or a complete answer with minor mistakes - 8 points;
"Satisfactory", incomplete answer (but not less than 60% of the information you need) and minor mistakes - 6 points;
"Unsatisfactory", unsatisfactory response (incorrect problem solution), requires mandatory re - writing at the end of the semester - 0 points.

The maximum sum of weighted points of control measures during the semester is:

$$R=5*Rlab+2*Rmcw=5*6+2*10=50.$$

Exam:
Admission to the exam is subject to passing all laboratory work, writing both modular test papers, and a starting rating of at least 17 points.
At the exam, students perform a written test. Each ticket contains two theoretical questions (tasks). Each question (task) is valued at 25 points.

Table1 — Correspondence of rating points to grades on the university scale

Score	*Grade*
100-95	Exellent
94-85	Very good
84-75	Good
74-65	Satisfactory
64-60	Sufficient
below 60	Fail
Course requirements are not met	Not graded

Syllabus of the course:

designed by Professor of the Department of Information Systems and Technologies, Bogdan Korniyenko

adopted by Department of Computer Engineering (protocol № 10, 25.05.2022)

approved by the methodical commission of FICT (protocol № 10, 09.06.2022)