Syllabus
Course Program

Software Quality, Testing and Support

Specialty
121 – Software Engineering

Institute
Institute of Computer Science and Information Technology

Educational program
Software Engineering

Department
Software Engineering and Management Intelligent Technologies (321)

Level of education
Bachelor's level

Course type
Special (professional), Mandatory

Semester
5

Language of instruction
Українська, англійська

Lecturers and course developers

Serhii Oriekhov

Serhii.Oriekhov@khpi.edu.ua
Doctor of Philosophy (Ph.D.), Associate Professor, Associate Professor of Software Engineering and Management Intelligent Technologies Department

Google Scholar: https://scholar.google.com/citations?user=OIJoUi4AAAAJ
ORCID: https://orcid.org/0000-0002-5040-5861
Scopus:
https://www2.scopus.com/authid/detail.uri?authorId=57210618400
More about the lecturer on the department's website

Natalia Fonta

Natalia.Fonta@khpi.edu.ua
PhD, Associate Professor, Associate Professor of SEMIT Department. Number of scientific and educational publications is more than 60.
(Google Scholar:
https://scholar.google.com.tw/citations?hl=ru&pli=1&user=we3S6nwAAAAJ;

Scopus:
https://www.scopus.com/authid/detail.uri?authorId=57215861869;
ORCID: https://orcid.org/0000-0001-5593-1409).

Leading lecturer in disciplines: "Probability theory and mathematical statistics"," Numerical Methods ".
Scientific directions: development of information systems for strategic company management; application of computer intelligence methods and models for solving problems of managing complex organizational systems; business analytics.
More about the lecturer on the department's website

Karina Melnyk

Karina.Melnyk@khpi.edu.ua
Ph.D., Associate Professor, Associate Professor of Software Engineering and Information Technology Management

Author (co-author) of more than 70 publications, 5 collective monographs, 10 articles in publications indexed in Scopus and Web of Science. (h-index = 5, i10-index = 1 in Google Scholar -https: //scholar.google.com/citations? user = xCU7GMgAAAAJ & hl = ru; ORCID ID https://orcid.org/0000-0001- 9642-5414; Scopus Author ID https://www.scopus.com/authid/detail.uri?authorId=57195074119).
Leading lecturer of the courses: Basics of Software Engineering (Bachelors) (in English), Methods of Empirical Information Processing (Bachelors) (in English and Ukrainian), Basics of Intelligent Systems Design (Masters) (in English and Ukrainian)
More about the lecturer on the department's website

General information

Summary

The task of the discipline is for students to acquire the knowledge and skills necessary to check the quality of software models and test the main artifact of the software development cycle - software code. In the course, students acquire the skills of creating test plans and choosing test strategies. They also consolidate these skills through the practical use of version control systems (e.g. GitHub), structural, functional and integration testing tools for programming languages Java, C#, Python, PHP.

Course objectives and goals

The formation of theoretical and practical knowledge in students, which is necessary for working with modern quality models, as well as test strategies and plans that implement a full cycle of testing, support and quality assessment of various artifacts of the software development life cycle.

Format of classes

Lectures, laboratory classes, self-study, consultations. Final control in the form of an exam.

Competencies

К01. Ability to think abstractly, analyze and synthesize.
К02. Ability to apply knowledge in practical situations.
К05. Ability to learn and master modern knowledge.
К06. Ability to search, process and analyze information from various sources.
К07. Ability to work in a team.
К16. Ability to formulate and ensure software quality requirements in accordance with customer requirements, terms of reference and standards.
К17. Ability to comply with specifications, standards, rules and guidelines in the professional field when implementing life cycle processes.
К22. Ability to accumulate, process and systematize professional knowledge of software development and maintenance and recognize the importance of lifelong learning.
К24. Ability to carry out the system integration process, apply change management standards and procedures to maintain the integrity, overall functionality and reliability of the software.
К25. Ability to reasonably choose and master the tools for software development and maintenance.
К26. Ability to think algorithmically and logically.

Learning outcomes

PLO01. Analyze, purposefully search and select information and reference resources and knowledge necessary for solving professional problems, taking into account modern achievements of science and technology.
PLO14. Apply in practice software tools for domain analysis, design, testing, visualization, measurement and documentation of software.
PLO15. Motivated to choose programming languages and development technologies to solve the problems of creating and maintaining software.
PLO19. To know and be able to apply methods of software verification and validation.
PLO20. Know approaches to assessing and ensuring software quality.

Student workload

The total volume of the course is 120 hours (4 ECTS credits): lectures – 16 hours, laboratory classes – 32 hours, self-study – 72 hours.

Course prerequisites

Fundamentals of Programming
Fundamentals of Software Engineering
Computer Architecture and Operating Systems

Features of the course, teaching and learning methods, and technologies

Teaching and learning methods:
interactive lectures with presentations, discussions, laboratory classes, teamwork, case method, student feedback, problem-based learning.
Forms of assessment:
written individual assignments for laboratory work (CAS), assessment of knowledge in laboratory classes (CAS), express surveys (CAS), online tests (CAS), final/semester control in the form of a semester exam, according to the schedule of the educational process (FAS).

Program of the course

Topics of the lectures

Topic 1: Concept of software quality and reliability
Definition of the concept of software quality. An overview of the most common software reliability models and an analysis of the possibility of their implementation.
Topic 2: Software quality models at the level of IT company, process and IT product
ISO quality model. CMMI IT Quality Model. Review of SWEBOK's IT product quality standard.
Topic 3: Characteristics of software quality
Fundamentals of the metric theory of programs. Halstead and McCabe quality metrics.
Topic 4: The place of verification and testing processes in the software life cycle
Concept of program verification and validation. Program verification process. Roles in the verification process. Documentation in the process of verification.
Topic 5: Testing criteria
Stages of the testing process. Definition of a good test. Structural criteria of tests. Functional test criteria. Mutational test criteria. Random test criteria.
Topic 6: Classification of software errors
Definition of the term software error. Classes of software errors. Report a software bug. Software error registration systems.
Topic 7: Test plan
Structure and purpose. Algorithm for creating a test plan. Version control and testing systems.

Topics of the workshops

Workshops are not provided within the discipline.

Topics of the laboratory classes

Topic 1. Creating a test software system for a further testing experiment in the programming language Java, C#, Javascript, Python, or PHP
Topic 2. Preparation of documentation for creating a test plan in the UML language and using the IDEF methodology
Topic 3. Calculations of values of typical quality metrics based on developed software artifacts
Topic 4. Use of the metric theory of programs for conducting test experiments
Topic 5. Application of structural testing criteria
Topic 6. Peculiarities of the application of testing criteria within the OOP paradigm
Topic 7. Using components to automate testing in Java, C#, Javascript, Python, or PHP programming environments
Topic 8. Creation of test plans. Control of the elimination of software errors with the help of special software

Self-study

Individual assignments are not provided in the curriculum.
Students are recommended with additional materials (videos, articles) for self-study and processing.

Course materials and recommended reading

Key literature

1. S. B. Vardeman, J. M. Jobe. Statistical Methods for Quality Assurance. Basics, Measurement, Control, Capability, and Improvement. Springer, 2016.
2. R.J. Leach. Introduction to Software Engineering. CRC Press, 2016.
3. P. Ammann, J. Offutt. Introduction to Software Testing. Cambridge University Press. 2008.
4. D. Graham, E. Veenendaal, I. Evans, R. Black. Foundations of Software Testing. Istqb Certification. Thomson. 2018.
5. M. Pezzè, M. Young. Software Testing and Analysis: Process, Principles, and Techniques. 2008.
6. K. Naik, P. Tripathy. Software Testing and Quality Assurance. Theory and Practice. John Wiley & Sons, Inc. 2008.
7. B. Rumpe. Agile Modeling with UML. Code Generation, Testing, Refactoring. Springer, 2017.
8 A. Takanen, J. DeMott, C. Miller, A. Kettunen. Fuzzing for Software Security Testing and Quality Assurance. Artech House, 2018.
9. O. Filipova, R. Vilao. Software Development From A to Z. A Deep Dive into all the Roles Involved in the Creation of Software. Apress, 2018.

Additional literature

1. D. Winkler, S. Biffl, J. Bergsmann. Software Quality. The Complexity and Challenges of Software Engineering and Software Quality in the Cloud. Springer, 2019.
2. J.F. Dooley. Software Development, Design and Coding. With Patterns, Debugging, Unit Testing, and Refactoring. Apress, 2017.
3. A. Mili, F. Tchier. Software Testing Concepts and Operations John Wiley & Sons, Inc. 2015.
4. Quality Assurance. Software Quality Assurance Made Easy. Solis Tech, 2018.
5. McCabe T.J. A Complexity Measure // IEEE Transactions on Software Engineering. – V.2, № 4, 1976. – pp.308 – 320.
6. Y. Singh. Software Testing. Cambridge University Press. 2012.
7. A. Pajankar Python Unit Test Automation. Practical Techniques for Python Developers and Testers. Apress, 2018.
8. Standard for Software Verification and Validation Plans (ANSI / IEEE standard 1012). 19. M. Hutcheson Software Testing Fundamentals. Methods and Metrics. Wiley Publishing Inc. 2003.

Assessment and grading

Criteria for assessment of student performance, and the final score structure

100% of the final grade consists of the results of the assessment in the form of an exam (40%) and current assessment (60%):
- 6 laboratory works (6% each);
- 2 tests (12% each).

Grading scale

Total points	National	ECTS
90–100	Excellent	A
82–89	Good	B
75–81	Good	C
64–74	Satisfactory	D
60–63	Satisfactory	E
35–59	Unsatisfactory (requires additional learning)	FX
1–34	Unsatisfactory (requires repetition of the course)	F

Norms of academic integrity and course policy

The student must adhere to the Code of Ethics of Academic Relations and Integrity of NTU "KhPI": to demonstrate discipline, good manners, kindness, honesty, and responsibility. Conflict situations should be openly discussed in academic groups with a lecturer, and if it is impossible to resolve the conflict, they should be brought to the attention of the Institute's management.
Regulatory and legal documents related to the implementation of the principles of academic integrity at NTU "KhPI" are available on the website: http://blogs.kpi.kharkov.ua/v2/nv/akademichna-dobrochesnist/

Approval

Approved by	08.06.2023	Head of the department Ihor HAMAIUN
	08.06.2023	Guarantor of the educational program Uliya LITVINOVA