UNIVERSITATEA „ALEXANDRU IOAN CUZA" din IAŞI PER LIBERTATEM AD VERITATEM

www.uaic.ro

COURSE DESCRIPTION

1. Program Information

1.1 University	**"Alexandru Ioan Cuza" University of Iaşi**
1.2 Faculty	**Computer Science**
1.3 Department	**Computer Science**
1.4 Study Domain	**Computer Science**
1.5 Study Cycle	**Master**
1.6 Study Program / Qualification	**Computational Optimization**

2. Course Information

2.1 Course Name	**Software Quality**
2.2 Course Teacher	**Lect. dr. Vlad Rădulescu**
2.3 Seminary Teacher	**Lect. dr. Vlad Rădulescu**

2.4 Study Year	**I-II**	**2.5** Semester	**I**	**2.6** Evaluation	**E**	**2.7** Course Status*	**OP**

** OB – Compulsory / OP – Optional*

3. Total estimated hours (hours per semester and didactic activities)

3.1 Hours per week	**4**	in which: **3.2** course	**2**	**3.3** seminary/laboratory	**2**
3.4 Hours in curriculum	**56**	in which: **3.5** course	**28**	**3.6** seminary/laboratory	**28**

Time Distribution	hours
Manual study, Course support, Bibliography, and others	**38**
Supplementary Documentation in library, in electronic forums, and on the field	**38**
Seminaries/laboratories preparation, homeworks, reports, portfolios and essays	**38**
Tutoring	**-**
Evaluation	**4**
Other activities (consultations per student)	**10**

3.7 Total hours individual study	**114**
3.8 Total hours per semester	**184**
3.9 Credits	**8**

4. Preconditions (if necessary)

4.1 Of Curriculum	**-**
4.2 Of Skills	**-**

5. Conditions (if necessary)

5.1 For Course Operation	**-**
5.2 For Seminary/Laboratory Operation	**-**

6. Specific Skills Acquired

Professional Skills	**C1. The identification of proper methodologies for developing software systems.** **C2. The identification of proper models and methods for solving real-life problems.** **C3. The development of dedicated informatic projects.**
Transversal Skills	**CT1. The use of efficient methods and techniques for learning, acquiring information, research and development of the capabilities to capitalize the knowledge, to adapt to the requirements of a dinamic society and to communicate in Romanian and in an international language.**

7. Course Objectives (from the grid of specific skills acquired)

7.1 General Objectives	Understanding the main elements that define the quality of software systems. Getting acquainted with the methods used in program testing and analysis.
7.2 Specific Objectives	Upon the successful completion of this course, the students will be able to: ▪ Describe the main concepts related to software testing, risk analysis, test planning, software quality measurement. ▪ Use software testing tools. ▪ Analyze software projects and the risk of defect arrival. ▪ Plan the testing of software systems. ▪ Decide the actions to be taken for improving the development process of a software project.

8. General Description

8.1	**Course**	**Teaching Methods**	**Observations** (hours and bibliographic references)
1	Introduction	exposition, debate, case studies, problem solving	-
2	Program testing	exposition, debate, case studies, problem solving	-
3	Defects in software systems. Code inspection	exposition, debate, case studies, problem solving	-
4	Risk analysis. Test planning	exposition, debate, case studies, problem solving	-
5	Testing levels: unit testing, integration testing, system testing, acceptance testing	exposition, debate, case studies, problem solving	-

6	Extreme testing. Regressive testing	exposition, debate, case studies, problem solving	-
7	Assertions. Debugging	exposition, debate, case studies, problem solving	-
8	Recapitulation	exposition, debate, case studies, problem solving	-
9-10	Measuring the software quality. Metrics for software quality. Defect removal	exposition, debate, case studies, problem solving	-
11	Software reliability models	exposition, debate, case studies, problem solving	-
12	Process metrics for testing	exposition, debate, case studies, problem solving	-
13	Complexity metrics	exposition, debate, case studies, problem solving	-
14	Recapitulation	exposition, debate, case studies, problem solving	-

Bibliography

Main references:
R. D. Craig, S. P. Jaskiel, *Systematic Software Testing*, SQE Publishing, 2007.
S. H. Kahn, *Metrics and Models in Software Quality Engineering*, Second Edition, Addison-Wesley, 2003.
Robert V. Binder, *Testing Object-Oriented Systems: Models, Patterns, and Tools*, Addison-Wesley, 2000.

Supplementary references:
G. J. Myers, *The Art of Software Testing*, Second Edition, Wiley, 2004.

8.2	**Seminary / Laboratory**	**Teaching methods**	**Observations** (hours and bibliographic references)
1	Programe testing; possible defects	debate, case studies, problem solving	-
2	Equivalence classes	debate, case studies, problem solving	-
3-5	Unit testing; usign the NUnit program and the NMock library	debate, case studies, problem solving	-
6	Load testing. Stress testing	debate, case studies, problem solving	-
7	Using assertions in the testing process	debate, case studies, problem solving	-
8	Recapitulation	debate, case studies, problem solving	-
9	Project work - specifications	debate, case studies, problem solving	-
10-11	Project work - application development	debate, case studies, problem solving	-
12	Project work - unit testing	debate, case studies, problem solving	-

13	Project work - assertions	debate, case studies, problem solving	-
14	Project work - documentation	debate, case studies, problem solving	-
Bibliography G. J. Myers, *The Art of Software Testing*, Second Edition, Wiley, 2004.			

9. Course content synchronization with the expectations of the community representatives, professional associations and employers from the program domain

The development of large software projects is an inherently error-prone activity. That is why knowing the testing techniques and methodologies is mandatory for the project managers. Beyond error tracking, the efficiency of program writing must be assessed and appropriate measures must be taken to improve the process; to achieve that, risk analysis and statistical models must be used in order to predict/improve the evolution of software projects.

10. Evaluation

Activity Type	**10.1 Evaluation criteria**	**10.2 Evaluation methods**	**10.3 The weight of each evaluation form (%)**
10.4 Course	understanding the concepts related to software quality	written test	50%
10.5 Seminary/ Laboratory	the ability to handle the development of large software projects	project	50%
10.6 Minimal performance standards			
- understanding the phases of a software project and the corresponding testing phases - the ability to develop a test plan for a simple software system - the ability to use testing techniques and tools (unit testing, assertions, code inspection)			