

## SYLLABUS

### 1. Information regarding the programme

1.1 Higher education institution	<b>Babes-Bolyai University</b>
1.2 Faculty	<b>Faculty of Mathematics and Computer Science</b>
1.3 Department	<b>Department of Computer Science</b>
1.4 Field of study	<b>Computer Science</b>
1.5 Study cycle	<b>Bachelor</b>
1.6 Study programme / Qualification	<b>Computer Science (in Romanian)</b>

### 2. Information regarding the discipline

2.1 Name of the discipline	<b>Test Design Techniques</b>						
2.2 Course coordinator	<b>Lecturer PhD Camelia Chisăliță-Crețu</b>						
2.3 Seminar coordinator	<b>Lecturer PhD Camelia Chisăliță-Crețu</b>						
2.4. Year of study	<b>3</b>	2.5 Semester	<b>6</b>	2.6. Type of evaluation	<b>C</b>	2.7 Type of discipline	<b>Optional</b>
2.8 Discipline Code	<b>MLE5110</b>						

### 3. Total estimated time (hours/semester of didactic activities)

3.1 Hours per week	5	Of which: 3.2 course	2	3.3 seminar/laboratory	1 lab + 2 project
3.4 Total hours in the curriculum	60	Of which: 3.5 course	24	3.6 seminar/laboratory	36
Time allotment:					Hours
Learning using manual, course support, bibliography, course notes					30
Additional documentation (in libraries, on electronic platforms, field documentation)					35
Preparation for seminars/labs, homework, papers, portfolios and essays					30
Tutorship					10
Evaluations					10
Other activities: .....					-
3.7 Total individual study hours	115				
3.8 Total hours per semester	175				
3.9 Number of ECTS credits	7				

### 4. Prerequisites (if necessary)

4.1. curriculum	<ul style="list-style-type: none"> <li>• OOP, Programming Fundamentals, Advanced Programming Methods</li> </ul>
4.2. competencies	<ul style="list-style-type: none"> <li>• Good programming skills in at least one of the programming languages Java, C#</li> </ul>

### 5. Conditions (if necessary)

5.1. for the course	<ul style="list-style-type: none"> <li>• Course hall with projector</li> </ul>
5.2. for the seminar /lab activities	<ul style="list-style-type: none"> <li>• Laboratory: computers and use of a programming language environment</li> </ul>

## 6. Specific competencies acquired

<b>Professional competencies</b>	<ul style="list-style-type: none"> <li>• <b>C2.1</b> Identify adequate software systems development methodologies</li> <li>• <b>C1.2</b> Identify and explain specific test design techniques that correspond to a testing level.</li> <li>• <b>C1.3</b> Source code and goal oriented test elaboration in a well-known programming language.</li> <li>• <b>C4.3</b> Identify models and methods adequate to real life problem solving.</li> </ul>
<b>Transversal competencies</b>	<ul style="list-style-type: none"> <li>• <b>CT1</b> Apply rules to organized and efficient work, responsibilities of didactical and scientific activities and creative capitalization of own potential, while respecting principles and rules for professional ethics.</li> <li>• <b>CT3</b> Use efficient methods and techniques for learning, knowledge gaining, and research and develop capabilities for capitalization of knowledge, accommodation to society requirements and communication in English.</li> </ul>

## 7. Objectives of the discipline (outcome of the acquired competencies)

7.1 General objective of the discipline	<ul style="list-style-type: none"> <li>• Enhance the students understanding of testing and test design techniques.</li> <li>• Provide the students with an environment in which they can explore the usage and usefulness of software testing and test design concepts in various business scenarios.</li> <li>• Induce a realistic and industry driven view of software testing concepts and their inherent benefits.</li> </ul>
7.2 Specific objective of the discipline	<ul style="list-style-type: none"> <li>• Give students the ability to explore various test design techniques applied to different levels of testing.</li> <li>• Improve the students' abilities to tackle on goal driven testing.</li> <li>• Enhance the students understanding of test design techniques value in business.</li> <li>• Students will be able to use various tools for the testing process (i.e., test management, test running, test reporting and bug reporting).</li> <li>• Students will be able to design test cases according to an established testing goal and using specific test design technique in order to investigate the software.</li> </ul>

## 8. Content

8.1 Course	Teaching methods	Remarks
<b>1. Software Testing. Test Design Techniques</b> 1.1. Software Testing. Goals. Scope 1.2. Test Design Technique. Attributes 1.3. Taxonomy of Test Design Techniques	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>2. Coverage-based Techniques I</b> 2.1. Focus. Objectives 2.2. Tours. Logical Expressions	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>3. Coverage-based Techniques II</b> 3.1. Specification-based Testing; 3.2. Requirements-based Testing;	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>4. Tester-based Techniques I</b>	<ul style="list-style-type: none"> <li>• Interactive exposure</li> </ul>	

4.1. Focus. Objectives 4.2. User Testing. Alpha Testing. Beta Testing	<ul style="list-style-type: none"> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>5. Tester-based Techniques II</b> 5.1. Bug Bashes. Paired Testing. 5.2. Coverage-based Techniques vs Tester-based Techniques	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>6. Activity-based Techniques</b> 6.1. Focus. Objectives 6.2. Guerilla Testing. All-pairs Testing 6.3. Use Cases Testing. Scenario Testing 6.4. Coverage-based Techniques vs Activity-based Techniques	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation</li> <li>• Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>7. Evaluation-based Techniques</b> 7.1. Focus. Objectives 7.2. Function Equivalence Testing. Self-verifying data	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>8. Desired result-based Techniques</b> 8.1. Focus. Objectives 8.2. Confirmation Testing. User Acceptance Testing 8.3. Desired-based Techniques vs Evaluation-based Techniques	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>9. Risk-based Techniques</b> 9.1. Focus. Objectives 9.2. Quick-tests. History-based Testing. Usability Testing 9.3. HTSM. Failure modes	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>10. Test Design Techniques Analysis</b> 10.1. Tester-based Techniques vs Activity-based Techniques 10.2. Risk-based Techniques vs Coverage-based Techniques 10.3. Desired result-based Techniques vs Risk-based Techniques	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>11. Essay Presentations</b>	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Explanation. Conversation</li> <li>• Didactical demonstration</li> </ul>	
<b>12. Essay Presentations</b>	<ul style="list-style-type: none"> <li>• Interactive exposure</li> <li>• Conversation</li> </ul>	

## Bibliography

[Pres10] R. S. Pressman, Software engineering: a practitioner's approach, seventh edition, Higher Education, 2010.

[Crs09] L. Crispin, J. Gregory, Agile testing: a practical guide for testers and agile teams, Addison-Wesley, 2009.

[You08] M. Pezzand, M. Young, Software Testing and Analysis: Process, Principles and Techniques, John Wiley & Sons, 2008.

[Nai08] K. Naik, P. Tripathy, Software testing and quality assurance. Theory and Practice, A John Wiley & Sons, Inc., 2008.

[Kat08] J. P. Katoen, Principles of Model Checking, MIT Press, May 2008.

[Pat05] R. Patton, Software Testing, Sams Publishing, 2005.

[Mye04] Glenford J. Myers, The Art of Software Testing, John Wiley & Sons, Inc., 2004.  
 [Brn02] I. Burnstein, Practical Software Testing, Springer, 2002.  
 [Kaner99] C. Kaner, J. Falk, H.Q. Nguyen, Testing Computer Software, Wiley, 1999.  
 [Perry97] W.E.Perry, R.W. Rice, Surviving the Top Ten Challenges of Software Testing – A People Oriented Approach, Dorset House Publishing, 1997.  
 [Kaner02] C. Kaner, J. Bach, B. Pettichord, Lesson Learned in Software Testing, Wiley, 2002.  
 [Page08] A. Page, K. Johnston, B. Rollison, Microsoft, How We Test Software at Microsoft, 2008.  
 [Whitt2012] J. Whittaker, J. Arbon J. Carollo, How Google Tests Software, Google, Pearson Education, 2012.

8.2 Seminar / laboratory	Teaching methods	Remarks
<b>1. Laboratory 1</b> Testing tools and platforms. Testing Project Setup	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
<b>2. Laboratory 2</b> Test Automation Tools	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
<b>3. Laboratory 3</b> Coverage-based Techniques OR Tester-based Techniques	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
<b>4. Laboratory 4</b> Risk-based Techniques	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
<b>5. Laboratory 5</b> Activity-based Techniques OR Desired result-based Techniques	Presentation, Conversation, Problematizations, Discovery, Other methods – individual study, exercises	
<b>6. Laboratory 6</b> Project turn-in	Evaluation	

**References:**  
See references from Lectures.

### 9. Corroborating the content of the discipline with the expectations of the epistemic community, professional associations and representative employers within the field of the program

- Students will know how to apply test design techniques for a software product, in a similar way they are used in industry.
- Students will be able to understand the differences between the goals and scope of the various test techniques applied to a software system.

### 10. Evaluation

Type of activity	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 Share in the grade (%)
10.4 Course	Design and develop a testing solution (project) for a software product with focus on test design techniques. The corresponding grade is denoted by <b>P</b> .	Oral Examination	70%
10.5 Seminar/laboratory activities	Each lab activity will be graded. The arithmetic average of the grades is denoted by <b>L</b> .	Laboratory Activity	30%

**Remark:**

- Laboratory assignments will be achieved in groups of 2-3 students.

- Testing project will be achieved in groups of 4-5 students.

#### 10.6 Minimum performance standards

- Students will be able to apply test design techniques according to established goals for a software system.
- Students will be able to understand the differences between software testing goal, scope, and test design technique concepts.
- The final grade (M) is computed as follows:  $M = 30\%L + 70\%P$ .
- At least  $M \geq 5.00$  is favourable to pass this course exam.

Date

Signature of course coordinator

Signature of seminar coordinator

**09.04.2021**

**Lect. PhD. Camelia Chisăliță-Crețu,**

**Lect. PhD. Camelia Chisăliță-Crețu,**

Date of approval

Signature of the head of department

**Prof. PhD. Laura Diosan**