



FIȘA DISCIPLINEI COURSE DESCRIPTION

1. Date despre program

Program Information

1.1 Instituția de învățământ superior <i>University</i>	Universitatea “Alexandru Ioan Cuza” din Iași <i>“Alexandru Ioan Cuza” University of Iași</i>
1.2 Facultatea <i>Faculty</i>	Facultatea de Informatică <i>Faculty of Computer Science</i>
1.3 Departamentul <i>Department</i>	Informatică <i>Department of Computer Science</i>
1.4 Domeniul de studii <i>Study Domain</i>	Informatică <i>Computer Science</i>
1.5 Ciclul de studii <i>Study Cycle</i>	Master <i>Master studies</i>
1.6 Programul de studii / Calificarea <i>Study Program / Qualification</i>	Informatică/Studii universitare de masterat în Informatică <i>Computer Science/Computer Science Master Graduate</i>

2. Date despre disciplină

Course Information

2.1 Denumirea disciplinei <i>Course Name</i>	Tehnici Avansate de Ingineria programării <i>Advanced Software Engineering Techniques</i>						
2.2 Titularul activităților de curs <i>Course Teacher</i>	Iftene Adrian						
2.3 Titularul activităților de seminar <i>Seminary Teacher</i>	Iftene Adrian						
2.4 An de studiu <i>Year of study</i>	1	2.5 Semestru <i>Semester</i>	1	2.6 Tip de evaluare <i>Evaluation</i>	M	2.7 Regimul disciplinei* <i>Course status*</i>	OB/ OP

* OB – Obligatoriu / OP – Opțional

* OB – Compulsory / OP – Optional

3. Timpul total estimat (ore pe semestru și activități didactice)

Total estimated hours (hours per semester and didactic activities)

3.1 Număr de ore pe săptămână <i>Hours per week</i>	4	din care: 3.2 curs <i>in which: course</i>	2	3.3 seminar/laborator <i>seminary/laboratory</i>	2
3.4 Total ore din planul de învățământ <i>Hours in curriculum</i>	56	din care: 3.5 curs <i>in which: course</i>	28	3.6 seminar/laborator <i>seminary/laboratory</i>	28
Distribuția fondului de timp <i>Time Distribution</i>					Ore <i>hours</i>
Studiu după manual, suport de curs, bibliografie și altele <i>Manual study, Course support, Bibliography, and others</i>					20
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren <i>Supplementary Documentation in library, in electronic forums, and on the field</i>					20
Pregătire seminarii/laboratoare, teme, referate, portofolii și eseuri <i>Seminaries/laboratories preparation, homeworks, reports, portfolios and essays</i>					40
Tutoriat <i>Tutoring</i>					
Examinări <i>Evaluation</i>					4
Alte activități <i>Other activities (consultations per student)</i>					



3.7 Total ore studiu individual <i>Total hours individual study</i>	130/ 105
3.8 Total ore pe semestru <i>Total hours per semester</i>	200/ 175
3.9 Număr de credite <i>Credits</i>	8/7

4. Precondiții (dacă este cazul)

Preconditions (if necessary)

4.1 De curriculum <i>Of Curriculum</i>	Algoritmi și Programare. Programare orientată obiect. Ingineria programării <i>Algorithms and programming. Object oriented programming. Programming Engineering.</i>
4.2 De competențe <i>Of Skills</i>	Programarea în limbaje de nivel înalt. Dezvoltarea și întreținerea aplicațiilor informatice de dimensiuni mari. Proiectarea și gestiunea bazelor de date și a șabloanelor de proiectare. <i>Programming using high level languages. Development and maintenance of large applications. Designing and using databases and design patterns.</i>

5. Condiții (dacă este cazul)

Conditions (if necessary)

5.1 De desfășurare a cursului <i>For Course Operation</i>	Sala de curs trebuie să dispună de video-proiector, conexiune la Internet și tablă pentru exemplificări <i>Overhead projector, internet connection, blackboard</i>
5.2 De desfășurare a seminarului/ laboratorului <i>For Seminary/Laboratory Operation</i>	Sala de laborator trebuie să dispună de conexiune la Internet și tablă pentru exemplificări. Studenții au nevoie de calculatoare pe care să fie instalat un limbaj ce le permite să programeze orientat-obiect (Java, C#, Python, PhP, C++, etc.) <i>Internet connection, blackboard, computers with appropriate OO language development environment installed (Java, C#, Python, PhP, C++, etc.)</i>

6. Competențe specifice acumulate

Specific Skills Acquired

Competențe profesionale <i>Professional Skills</i>	La finalizarea cu succes a acestei discipline, studenții vor fi capabili să: C1. Programarea în limbaje de nivel înalt. C2. Dezvoltarea și întreținerea aplicațiilor informatice. C3. Proiectarea și gestiunea bazelor de date. C4. Modelarea proiectului și folosirea șabloanelor de proiectare <i>Upon successful completion of this discipline, students will be able to:</i> C1. <i>Programming in high level languages</i> C2. <i>Developing and maintaining IT applications</i> C3. <i>Designing and using databases</i> C4. <i>Designing and using design patterns</i>
Competențe transversale <i>Transversal Skills</i>	La finalizarea cu succes a acestei discipline, studenții vor fi capabili să: CT1. Desfășurarea eficientă a activităților organizate într-un grup inter-disciplinar și dezvoltarea capacităților empatice de comunicare inter-personala, de relaționare și colaborare cu grupuri diverse <i>Upon successful completion of this discipline, students will be able to:</i>



CT1. Working effectively within an interdisciplinary group and the development of interpersonal communication skills

7. Obiectivele disciplinei (din grila competențelor specifice acumulate)

Course Objectives (from the grid of specific skills acquired)

7.1 Obiectivul general <i>General Objective</i>	<p>Construirea unei viziuni profesionale asupra procesului de dezvoltare a programelor. Studenții vor învăța metode și tehnici avansate pentru realizarea unor programe complexe de o calitate superioară în contextul respectării cerințelor clienților privind funcționalitatea, costurile și termenul limită.</p> <p><i>Building a professional image with regards to the process of developing IT products. Students will learn advanced methods and techniques for creating complex, high quality programs while adapting to clients' requirements regarding function, costs and deadline.</i></p>
7.2 Obiectivele specifice <i>Specific Objectives</i>	<p>La finalizarea cu succes a acestei discipline, studenții vor fi capabili să:</p> <ul style="list-style-type: none"> • Explice cum pot fi folosite principiile programării orientate obiect în dezvoltarea proiectelor de dimensiuni mari • Descrie pașii necesari implementării cu succes a unui proiect pentru un client și vor putea descrie pașii necesari testării produsului construit • Utilizeze tehnicile orientate obiect în implementarea componentelor funcționale și de testare ale unui proiect dat • Analizeze cerințele clientului, analizeze erorile software care apar pe parcursul dezvoltării proiectului, analizeze testele manuale pe care le execută asupra unui sistem • Calculeze necesarul de timp, necesarul de personal și necesarul de bani pentru dezvoltarea proiectului cerut de client <p><i>On successful completion of this course, students will be able to:</i></p> <ul style="list-style-type: none"> • <i>Explain how to use object-oriented programming principles in large development projects</i> • <i>Describe the steps required to successfully implement a project for a client and describe the steps necessary for testing the project</i> • <i>Utilize object-oriented technologies to implement and test functional components of a given project</i> • <i>Analyze customer requirements, analyze software errors that occur during the development of the project, analyze the manual tests on a system</i> • <i>Calculation of time, staffing and money needed to develop the project requested by the client</i>

8. Conținut

Content

8.1	Curs <i>Course</i>	Metode de predare <i>Teaching methods</i>	Observații (ore și referințe bibliografice) <i>Notes</i>
1.	Conținutul cursului. Bibliografie. Recapitulare ingineria programării <i>Course content. Bibliography. Software engineering recapitulation.</i>	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [1], [2], [3], [4]
2.	Swebok. Model Driven Development (MDA, AMDD), Test Driven Development, Domain Specific Language (EMF)	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [2], [4], [7], [8], [9], [10]
3.	Modelare (EMF, IBM Rational Rose Data Modeler)	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune	2 ore, [2], [10], [12]



	<i>Modelling (EMF, IBM Rational Rose Data Modeler)</i>	electronică. Desenarea la tablă a soluțiilor pentru o problemă dată. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	
4.	Business Process Modeling (BPMN: Visio BPMN Modeler, Intalio, Jadex, JBoss jBPM), Aspect Oriented Programming (Introduction, Basic Elements)	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. Problematizare. <i>Slide presentation. Course notes and tutorials available in electronic format. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [2], [12], [13]
5.	Domain Driven Design	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore [5], [6]
6.	AOP (Details, AspectJ, NKalore)	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. Problematizare și lucrul pe exemple. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [13]
7.	AOP, Runtime Verification, Monitoring-Oriented Programming (MOP)	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [13], [14]
8.	Recapitulare. Swebok, MDD, TDD, BPMN, AOP. <i>Revision of previous courses. Swebok, MDD, TDD, BPMN, AOP.</i>	Problematizare. Desenarea la tablă a soluțiilor pentru o problemă dată. <i>Case studies. Solutions for a given problem will be drawn on the blackboard. Problematization and work on examples.</i>	2 ore, [2-14]
9.	Runtime Verification, Java MOP Exemple, Service Oriented Architecture (SOA) <i>Runtime Verification, Java MOP Examples, Service Oriented Architecture (SOA)</i>	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [3], [15], [16]
10.	SOA, Serverless, Quality of Service (QoS), Testare funcțională non-funcțională <i>SOA, Serverless, Quality of Service (QoS), Functional and Non-Functional Testing</i>	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [3], [16], [17], [23]
11.	Arhitecturi degradate, Refactorizare <i>Rotting Design, Refactoring</i>	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [3], [18], [19], [20]
12.	Release, Deployment și Maintenance. Reutilizare de cod, Free software licenses	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [1], [21]



	<i>Release, Deployment and Maintenance, Code reuse, Free software licenses</i>		
13.	Selenium IDE, Etică <i>Selenium IDE, Ethics</i>	Prezentare de slide-uri. Note de curs și tutoriale disponibile în versiune electronică. <i>Slide presentation. Course notes and tutorials available in electronic format.</i>	2 ore, [22]
14.	Prezentări ale celor mai bune proiecte <i>Best projects presentations</i>	Prezentare de slide-uri. Problematizare și lucru pe exemple. <i>Slide presentation. Problematization and work on examples.</i>	2 ore

Bibliografie*Bibliography***Referințe principale:***Main references:*

- [1] Ian Sommerville: Software Engineering, Addison Wesley, 2001
- [2] Craig Larman: Applying UML and Patterns, Addison Wesley, 2002
- [3] Erich Gamma, Richard Helm, Ralph Johnson, John Vissides: Design Patterns, Elements of Reusable Object-Oriented Software, Addison Wesley, 1998
- [4] Alain Abran, James W. Moore: Software Engineering Body of Knowledge, IEEE Computer Science, 2004.

Referințe suplimentare:*Additional references:*

- [5] Eric Evans. 2003. Domain-Driven Design. Tackling Complexity in the Heart of Software. Addison Wesley.
- [6] Abel Avram and Floyd Marinescu. 2006. Domain-Driven Design Quickly. InfoQ Enterprise Software Development Series
- [7] CA Gen: <http://www.ca.com/us/products/detail/ca-gen.aspx>
- [8] AMDD: <http://www.agilemodeling.com/essays/amdd.htm>
- [9] TDD: <http://c2.com/cgi/wiki?TestDrivenDevelopment>,
http://en.wikipedia.org/wiki/Test-driven_development
- [10] EMF: <http://www.eclipse.org/modeling/emf/>,
<http://www.devx.com/Java/Article/29093>
- [11] IBM Rational Rose Data Modeler: <http://www01.ibm.com/software/awdtools/developer/datamodeler/>
- [12] BPMN: <http://www.bpmn.org/>
- [13] AOP: <http://onjava.com/pub/a/onjava/2004/01/14/aop.html>,
http://en.wikipedia.org/wiki/Aspect-oriented_programming
- [14] Chen, F., Roșu, G. MOP: An Efficient and Generic Runtime Verification Framework. OOPSLA'07, ACM press, pp 569-588, 2007
- [15] Patrick Meredith and Grigore Rosu. 2010. Runtime Verification with the RV System, First International Conference on Runtime Verification (RV'10), Springer, Lecture Notes in Computer Science, 6418, 136-152
- [16] SOA Examples: <http://www.ibm.com/developerworks/webservices/library/ws-soa-composite7/index.html>,
<http://itransform.abstraction.com/2009/09/sample-viewof-services-in-system.html>,
<http://www.infoq.com/articles/soa-healthcare>
- [17] Siegel, E. D.: Designing QoS solutions for the enterprise, 1999,
<http://www.wiley.com/legacy/compbooks/siegel/>
- [18] Robert Cecil Martin: Design Principles and Design Patterns. 2000.
http://www.objectmentor.com/resources/articles/Principles_and_Patterns.pdf
- [19] Robert Cecil Martin. Agile Development. Principles, Patterns and Practices, Prentice-Hall, 2003
- [20] Martin Fowler. Refactoring: Improving the Design of Existing Code. 1997.
<http://jaoo.dk/jaoo1999/schedule/MartinFowlerRefactoring.pdf>
- [21] Software release life cycle: http://en.wikipedia.org/wiki/Software_release_life_cycle
- [22] Software Engineering Ethics: <http://www.acm.org/about/se-code>



[23] AWS Amazon. 2017. Serverless Architectures with AWS Lambda. Overview and Best Practices. Amazon Web Services.			
8.2	Laborator <i>Laboratory</i>	Metode de predare <i>Teaching methods</i>	Observații (ore și referințe bibliografice) <i>Notes</i>
1.	Proiect software pe tematica de master a studentului: Introducere <i>Software project on student master thematic: Introduction</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case studies. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [1], [2], [3], [4]
2.	Proiect software pe tematica de master a studentului: State-of-the-art <i>Software project on student master thematic: State-of-the-art</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [2], [4], [5], [6]
3.	Proiect software pe tematica de master a studentului: Analiza cerințelor, diagrame UML <i>Software project on student master thematic: Requirements analysis, UML diagram</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [2], [7], [8], [9], [10]
4.	Proiect software pe tematica de master a studentului: șabloane de proiectare, modelare <i>Software project on student master thematic: Design patterns, modeling</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [2], [10], [12]
5.	Proiect software pe tematica de master a studentului: MDD (AMDD), TDD, DDD, BPMN <i>Software project on student master thematic: MDD (AMDD), TDD, DDD, BPMN</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [2], [12], [13]
6.	Proiect software pe tematica de master a studentului: AOP, MOP <i>Software project on student master thematic: AOP, MOP</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [13]
7.	Proiect software pe tematica de master a studentului: Implementare <i>Software project on student master thematic: Implementation</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [13], [14]
8.	Proiect software pe tematica de master a studentului: Revizuire <i>Software project on student master thematic: Revision</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [2-14]
9.	Proiect software pe tematica de master a studentului: Java MOP	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată.	2 ore, [3], [15], [16]



	<i>Software project on student master thematic: Java MOP</i>	<i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	
10.	Proiect software pe tematica de master a studentului: Integrare, Testare funcțională <i>Software project on student master thematic: Integration, Functional testing</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [3], [16], [17]
11.	Proiect software pe tematica de master a studentului: Refactorizare <i>Software project on student master thematic: Refactoring</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [3], [18], [19], [20]
12.	Proiect software pe tematica de master a studentului: Statistici <i>Software project on student master thematic: Statistics</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [1], [21]
13.	Proiect software pe tematica de master a studentului: Comparație cu soluțiile similar <i>Software project on student master thematic: Comparison with similar solutions</i>	Problematizare. Discuții. Schițarea la tablă a soluțiilor pentru o problemă dată. <i>Case study. Discussions. Solutions for a given problem will be drawn on the blackboard.</i>	2 ore, [22]
14.	Proiect software pe tematica de master a studentului: Evaluare finală <i>Software project on student master thematic: Final evaluation</i>	Prezentare folosind video-proiectorul. Problematizare. Discuții. <i>Video presentation. Case study. Discussion.</i>	2 ore

Bibliografie:*Bibliography***Referințe principale:***Main references:*

- [1] Ian Sommerville: Software Engineering, Addison Wesley, 2001
- [2] Craig Larman: Applying UML and Patterns, Addison Wesley, 2002
- [3] Erich Gamma, Richard Helm, Ralph Johnson, John Vissides: Design Patterns, Elements of Reusable Object-Oriented Software, Addison Wesley, 1998
- [4] Alain Abran, James W. Moore: Software Engineering Body of Knowledge, IEEE Computer Science, 2004.

Referințe suplimentare:*Additional references:*

- [5] SOLID (The First 5 Principles of Object Oriented Design):
<https://scotch.io/bar-talk/s-o-l-i-d-the-first-fiveprinciples-of-object-oriented-design>
- [6] SOLID principles with real world examples:
<http://blog.gauffin.org/2012/05/solid-principles-withreal-world-examples/>
- MDA: http://en.wikipedia.org/wiki/Model-driven_architecture
- [7] CA Gen: <http://www.ca.com/us/products/detail/ca-gen.aspx>
- [8] AMDD: <http://www.agilemodeling.com/essays/amdd.htm>
- [9] TDD: <http://c2.com/cgi/wiki?TestDrivenDevelopment>,
http://en.wikipedia.org/wiki/Test-driven_development
- [10] EMF: <http://www.eclipse.org/modeling/emf/>,
<http://www.devx.com/Java/Article/29093>



- [11] IBM Rational Rose Data Modeler: <http://www01.ibm.com/software/awdtools/developer/datamodeler/>
- [12] BPMN: <http://www.bpmn.org/>
- [13] AOP: <http://onjava.com/pub/a/onjava/2004/01/14/aop.html>,
http://en.wikipedia.org/wiki/Aspect-oriented_programming
- [14] Chen, F., Roșu, G. MOP: An Efficient and Generic Runtime Verification Framework. OOPSLA'07, ACM press, pp 569-588, 2007
- [15] Patrick Meredith and Grigore Rosu. 2010. Runtime Verification with the RV System, First International Conference on Runtime Verification (RV'10), Springer, Lecture Notes in Computer Science, 6418, 136-152
- [16] SOA Examples: <http://www.ibm.com/developerworks/webservices/library/ws-soa-composite7/index.html>,
<http://itransform.abstraction.com/2009/09/sample-viewof-services-in-system.html>,
<http://www.infoq.com/articles/soa-healthcare>
- [17] Siegel, E. D.: Designing QoS solutions for the enterprise, 1999,
<http://www.wiley.com/legacy/compbooks/siegel/>
- [18] Robert Cecil Martin: Design Principles and Design Patterns. 2000.
http://www.objectmentor.com/resources/articles/Principles_and_Patterns.pdf
- [19] Robert Cecil Martin. Agile Development. Principles, Patterns and Practices, Prentice-Hall, 2003
- [20] Martin Fowler. Refactoring: Improving the Design of Existing Code. 1997.
<http://jaoo.dk/jaoo1999/schedule/MartinFowlerRefactoring.pdf>
- [21] Software release life cycle: http://en.wikipedia.org/wiki/Software_release_life_cycle
- [22] Software Engineering Ethics: <http://www.acm.org/about/se-code>

9. Coroborarea conținutului disciplinei cu așteptările reprezentanților comunității, asociațiilor profesionale și angajatorilor reprezentativi din domeniul aferent programului

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

Companiile de IT din Iași și din țară folosesc în dezvoltarea de proiecte de dimensiuni mari noțiuni de inginerie software. Etapele necesare dezvoltării aplicațiilor de dimensiuni mari (ingineria cerințelor, modelare, implementare, integrare, testare, deployment) sunt discutate pe larg pe parcursul cursului, iar studenții învață noțiunile de bază și parcurg efectiv aceste etape. Pe parcursul verii studenții participă la sesiuni de stagii în cadrul firmelor, fiind implicați în proiecte reale, unde aplică practic pe proiecte reale noțiunile teoretice învățate la acest curs. În urma discuțiilor cu principalii angajatori, acest curs se actualizează de la an la an, adaptându-se la cerințele actuale ale pieței de IT și nu numai.

IT companies from Iași and elsewhere in the country use software engineering techniques in the development of large-scale projects. The steps necessary for developing large applications (requirements engineering, modeling, implementation, integration, testing, and deployment) are widely discussed throughout the course, and students learn the basics and actually go through these stages. During the summer students participate in internships in companies where they are involved in real projects where apply the theoretical concepts learned during the course in real projects. Following discussions with major employers, this course is updated from year to year, adapting to the current needs of the IT market and beyond.

10. Evaluare

Evaluation

Tip activitate <i>Activity Type</i>	10.1 Criterii de evaluare <i>Evaluation criteria</i>	10.2 Metode de evaluare <i>Evaluation methods</i>	10.3 Pondere în nota finală (%) <i>The weight of each evaluation form (%)</i>
10.4 Curs <i>Course</i>	Punctaj minim la examen: Examenul nu este obligatoriu, dar cine susține examenul trebuie să obțină cel puțin 40% din punctajul acestuia, altfel va fi considerat picat.	Test scris + Bonus pentru activitatea de la curs <i>Written test + bonuses for course activity</i>	20 %



	<i>Minimum exam score: The exam is not mandatory, but whoever takes the exam must obtain at least 40% of its score, otherwise it will be considered failed.</i>		
10.5 Seminar/ Laborator <i>Seminary/ Laboratory</i>	Punctaj minim laborator: 60% din punctajul maxim ce poate fi obținut la laborator fără bonusuri (minim 60 % din punctajele temelor din primele 6 laboratoare și minim 60 % din punctajul proiectului) <i>Passing grade: 60% of the maximum grade obtainable for the laboratory work without bonuses (at least 60 % of the points from the activities during the first 6 laboratories and at least 60 % of the maximum project points)</i>	Prezentare de soluții la temele propuse săptămânal în primele 6 săptămâni + Lucrul la proiect + Bonus pentru activități suplimentare ce au legătură cu cursul de IP <i>Solutions for the activities given during the first 6 weeks + project work + bonuses for extra activities related to the course material.</i>	80 %
10.6 Standard minim de performanță <i>Minimal performance standards</i>			
<p>Studentii vor fi capabili să identifice principalele cerințe în urma discuției cu clientul și vor fi capabili să modeleze soluția pe care acesta o așteaptă. Implementarea se va reduce la construirea claselor principale și a atributelor acestora, fără implementarea metodelor de bază. Testarea proiectului o vor putea face din punct de vedere al testării automate și manuale.</p> <p>Punctaj minim laborator: 60% din punctajul maxim ce poate fi obținut la laborator fără bonusuri (minim 60 % din temele de laborator și minim 60 % proiect)</p> <p>Punctaj minim la examen: Examenul nu este obligatoriu, dar cine susține examenul trebuie să obțină cel puțin 40% din punctajul acestuia, altfel va fi considerat picat.</p> <p>Pentru studenții care îndeplinesc criteriile de promovare nota finală se stabilește împărțind punctajul obținut la maximul punctajelor fără bonus, iar rezultatul este înmulțit cu 10.</p> <p>Studentul care participă la examen, va primi o notă, altfel va fi considerat absent. Dacă unul din criteriile de promovare nu este îndeplinit, studentul va obține o nota mai mică sau egală cu 4.</p> <p><i>Students will be able to identify the main requirements after discussion with the client and be able to provide the solution which he expects. Implementation requires the building of main classes and their attributes without implementing basic methods. Testing of the project will be done in terms of manual testing.</i></p> <p><i>Laboratory Minimum score: 60% of the maximum score that can be obtained in the laboratory (minimum 60% of laboratory subjects and minimum 60% project)</i></p> <p><i>Minimum exam score: The exam is not mandatory, but whoever takes the exam must obtain at least 40% of its score, otherwise it will be considered failed.</i></p> <p><i>For students who meet the promotion criteria, the final mark is determined by dividing the score obtained to the maximum points without a bonus, and the result is multiplied by 10.</i></p> <p><i>Students participating in the exam will receive a grade, otherwise they will be considered absent. If one of the criteria for promotion is not met, the student will get a grade less than or equal to 4.</i></p>			

Data completării
Date

Titular de curs și seminar
*Course and Seminary/Laboratory
Teacher*

Prof.dr. Iftene Adrian

24.09.2025

Data avizării în departament
Department Date of Approval

Director de departament
Director of the Department
Conf.dr. Andrei Arusoaie