FIŞA DISCIPLINEI

Limbaje de programare multiparadigma

Anul universitar 2025-2016

1. Date despre program

1.1. Instituţia de învăţământ superior	Universitatea Babes-Bolyai Cluj-Napoca	
1.2. Facultatea	Facultatea de Matematica si Informatica	
1.3. Departamentul	Departamentul de Informatica	
1.4. Domeniul de studii	Informatica	
1.5. Ciclul de studii	Licenta	
1.6. Programul de studii / Calificarea	Informatica - romana	
1.7. Forma de învăţământ	Cu frecventa	

2. Date despre disciplină

2.1. Denumirea disciplinei	Limbaje de programare multiparadigma					Codul disciplinei	**MLR5163**
2.2. Titularul activităţilor de curs	Conf. Dr. Niculescu Virginia						
2.3. Titularul activităţilor de seminar	Conf. Dr. Niculescu Virginia						
2.4. Anul de studiu	3	2.5. Semestrul	6	2.6. Tipul de evaluare	C	2.7. Regimul disciplinei	Optional

3. Timpul total estimat (ore pe semestru al activităţilor didactice)

3.1. Număr de ore pe săptămână	**4**	din care: 3.2. curs	**2**	3.3. seminar/ laborator/proiect	**2**
3.4. Total ore din planul de învăţământ	48	din care: 3.5. curs	24	3.6 seminar/laborator/proiect	**24**
Distribuţia fondului de timp pentru studiul individual (SI) şi activităţi de autoinstruire (AI)					**ore**
Studiul după manual, suport de curs, bibliografie şi notiţe (AI)					15
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate şi pe teren					15
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii şi eseuri					30
Tutoriat (consiliere profesională)					4
Examinări					10
Alte activităţi [comunicare bidirecţională cu titularul de disciplină / tutorele]					3
3.7. Total ore studiu individual (SI) şi activităţi de autoinstruire (AI)				**77**	
3.8. Total ore pe semestru				**125**	
3.9. Numărul de credite				**5**	

4. Precondiţii (acolo unde este cazul)

4.1. de curriculum	Fundamentele programarii, Programare Orientata Obiect, Programare functionala
4.2. de competenţe	Abilitati de implementare programe si abilitati de abstractizare

5. Condiţii (acolo unde este cazul)

5.1. de desfăşurare a cursului	Sala cu proiector
5.2. de desfăşurare a seminarului/ laboratorului	Laborator cu statii de lucru si proiector

6.1. Competenţele specifice acumulate[1]

[1] Se poate opta pentru competenţe sau pentru rezultatele învăţării, respectiv pentru ambele. În cazul în care se alege o singură variantă, se va şterge tabelul aferent celeilalte opţiuni, iar opţiunea păstrată va fi numerotată cu 6.

Competenţe profesionale/ esenţiale	• programarea în limbaje de nivel înalt • dezvoltarea şi întreţinerea aplicaţiilor informatice • utilizarea instrumentelor informatice în context interdisciplinar • utilizarea bazelor teoretice ale informaticii şi a modelelor formale
Competenţe transversale	• aplicarea regulilor de muncă organizată şi eficientă, a unor atitudini responsabile faţă de domeniul didactic-ştiinţific, pentru valorificarea creativă a propriului potenţial, cu respectarea principiilor şi a normelor de etică profesională • desfăşurarea eficientă a activităţilor organizate într-un grup interdisciplinar şi dezvoltarea capacităţilor empatice de comunicare interpersonală, de relaţionare şi colaborare cu grupuri diverse • utilizarea unor metode şi tehnici eficiente de învăţare, informare, cercetare şi dezvoltare a capacităţilor de valorificare a cunoştinţelor, de adaptare la cerinţele unei societăţi dinamice şi de comunicare în limba română şi într-o limbă de circulaţie internaţională

6.2. Rezultatele învăţării

Cunoştinţe	-Absolventul cunoaşte multiple limbaje de programare si este capabil sa scrie aplicaţii in limbaje compilate, interpretate sau dinamice având capacitatea de a alege limbajul de programare potrivit pentru specificul aplicaţiei de dezvoltat.
Aptitudini	-Absolventul este capabil să identifice probleme complexe şi să examineze probleme conexe pentru a dezvolta opţiuni de rezolvare şi implementa soluţii. -Absolventul are capacitatea de a evalua diferite arhitecturi si soluţii posibile pentru o problema si a alege pe cel potrivit pentru cerinţele si constrângerile specifice aplicaţiei de dezvoltat. -Absolventul are abilitatea de a înţelege şi comunica eficient informaţiile.
Responsabilităţi şi autonomie	- Absolventul este capabil de a redacta un raport ştiinţific. - Absolventul are abilitatea de a înţelege şi comunica eficient informaţiile.

7. Obiectivele disciplinei (reieşind din grila competenţelor acumulate)

7.1 Obiectivul general al disciplinei	• Aprofundarea de catre studenti a intelegerii diferitelor paradigme de programare . • Intelegerea si aprofundarea relatiilor dintre teoria paradigmelor de programare si aplicarea lor concreta in diferite limbaje de programare.
7.2 Obiectivele specifice	• Aprofundarea cunostintelor de programare in limbajele: C++, Java si Python. • Invatarea unor limbaje noi de programare (Kotlin, Scala, Go, Rust, Ruby, Julia…).

8. Conţinuturi

8.1 Curs	Metode de predare	Observaţii
C1 Abstractizare la nivelul limbajelor de programare. Generatii de limbaje de programare	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	
C2-C3 Paradigme de programarea: imperativa, declaratica (functionala si logica), orientare-obiect, concurenta, reactiva.	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	

C4 Paradigme de programare in Python	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	
C5 Paradigme de programare in Java	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	
C6 Paradigme de programare in C++	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	
C7 Scala -limbaj de programare multiparadigma		
C8 Kotlin -limbaj de programare multiparadigma	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	
C10 Mecanisme specifice pentru asigurarea concurentei: co-rutine. Exemplificari	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	
C11 Mecanisme specifice pentru asigurarea concurentei: model Actor. Exemplificari	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	
C12 Prezentari referate studenti si analiza concluziva	Expunere, descriere, explicatie, exemple, discutii ale unor studii de caz.	

Bibliografie

1. Michael Scott. Programming Language Pragmatics. 4th ed. Morgan Kaufmann, 2015
2. Martin Odersky, Lex Spoon, Bill Venners.Programming in Scala: A Comprehensive Step-by-Step Guide, 2nd Edition
3. Paul Chiusano and Runar Bjarnason Functional Programming in Scala, Mannon, 2014.
4. Bjarne Stroustrup: The C++ Programming Language Special Edition, Addison-Wesley, 2000
5. Andrei Alexandrescu. Modern C++ Design: Generic Programming and Design Patterns Applied, Addison-Wesley Professional.2001
6. Georgy Pashkov Multi-Paradigm Programming with Modern C++. Packt Publishing.2020
7. David Vandevoorde, Douglas Gregor, Nicolai M. Josuttis. C++ Templates: The Complete Guide, 2nd Edition**.** Addison-Wesley Professional. 2017 Chapter: C++ Metaprogramming.
8. C.D. Marlin. Coroutines. A Programming Methodology, a Language Design and an Implementation. Springer-Verlag Berlin Heidelberg
9. Kotlin Docs | Kotlin Documentation [https://kotlinlang.org/docs/home.html]

8.2 Seminar / laborator	Metode de predare	Observaţii
1. Prezentarea cerintelor corepunzatoare referatului stiintific si alegerea cazurilor de utilizare (preobleme)	Implementari, dezbatere, explicatie,exemple.	Seminarul se desfasoara la 2 saptamani cate 2 ore
2. Exemplificare folosirea diferitelor paradigme de programare in Python – analiza lizibilitatii, eficientei si performantei.	Implementari, dezbatere, explicatie,exemple.	
3. Exemplificare folosirea diferitelor paradigme de programare in Java analiza lizibilitatii, eficientei si performantei.	Implementari, dezbatere, explicatie,exemple.	
4. Exemplificare folosirea diferitelor paradigme de programare in C++ analiza lizibilitatii, eficientei si performantei.	Implementari, dezbatere, explicatie,exemple.	
5. Prezentari referate studenti	Dialog, dezbatere, explicatie,exemple.	
6. Prezentari referate studenti	Dialog, dezbatere, explicatie,exemple.	

Bibliografie

***, The Python Language Reference — Python 3.13.3 documentation [https://docs.python.org/3/reference/index.html]
***, Scala Tutorial. https://www.tutorialspoint.com/scala/index.htm
***, Tutoriale Java http://download.oracle.com/javase/tutorial/

***, Tutorial C++ [https://en.cppreference.com/] ***, Kotlin Docs | Kotlin Documentation [https://kotlinlang.org/docs/home.html]

9. Coroborarea conţinuturilor disciplinei cu aşteptările reprezentanţilor comunităţii epistemice, asociaţiilor profesionale şi angajatori reprezentativi din domeniul aferent programului

- Cursul respecta Recomandarile IEEE and ACM Curricula pentru studii in Computer Science;
- Cursuri cu tematica similara exista in programele de studii ale majoritatii universitatilor din tara si strainatate.
- Cursul defineste dobandirea unor abilitati care constituie avantaje evidentiate de potentialele firme angajatoare din domeniu.

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 Metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	• Referat care prezinta un limbaj multiparadigma • Mini-proiect in limbajul prezentat	Colocviu	90%
10.5 Seminar/laborator	• Corectitudine si performanta	Analiza cod, testare	10%
10.6 Standard minim de performanţă			
Minim nota 5 (pe o scara intre 1 si 10).			

11. Etichete ODD (Obiective de Dezvoltare Durabilă / Sustainable Development Goals)[2]

Nu se aplică.

Data completării:
15.04.2025

Semnătura titularului de curs

Semnătura titularului de seminar

Data avizării în departament:
...

Semnătura directorului de departament

Conf.dr. Adrian STERCA

[2] Păstraţi doar etichetele care, în conformitate cu *Procedura de aplicare a etichetelor ODD în procesul academic*, se potrivesc disciplinei şi ştergeţi-le pe celelalte, inclusiv eticheta generală pentru *Dezvoltare durabilă* - dacă nu se aplică. Dacă nicio etichetă nu descrie disciplina, ştergeţi-le pe toate şi scrieţi "*Nu se aplică.*".