FIŞA DISCIPLINEI

Fundamentele Programării

Anul universitar 2025-2026

1. Date despre program

1.1. Instituţia de învăţământ superior	Universitatea Babeş-Bolyai Cluj-Napoca
1.2. Facultatea	Facultatea de Matematică şi Informatică
1.3. Departamentul	Departamentul de informatică
1.4. Domeniul de studii	Informatică
1.5. Ciclul de studii	Licenţă
1.6. Programul de studii / Calificarea	Informatică
1.7. Forma de învăţământ	Cu frecvenţă

2. Date despre disciplină

2.1. Denumirea disciplinei	Fundamentele programării					Codul disciplinei	MLR5005
2.2. Titularul activităţilor de curs	Prof. dr. CZIBULA Istvan Gergely						
2.3. Titularul activităţilor de seminar	Prof. dr. CZIBULA Istvan Gergely						
2.4. Anul de studiu	1	2.5. Semestrul	1	2.6. Tipul de evaluare	E	2.7. Regimul disciplinei	Obligatoriu

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

3.1. Număr de ore pe săptămână	6	din care: 3.2. curs	2	3.3. seminar/ laborator/proiect	2 sem 2 lab
3.4. Total ore din planul de învăţământ	82	din care: 3.5. curs	28	3.6 seminar/laborator/proiect	56
Distribuţia fondului de timp pentru studiul individual (SI) şi activităţi de autoinstruire (AI)					**ore**
Studiul după manual, suport de curs, bibliografie şi notiţe (AI)					14
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate şi pe teren					12
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii şi eseuri					14
Tutoriat (consiliere profesională)					8
Examinări					18
Alte activităţi					
3.7. Total ore studiu individual (SI) şi activităţi de autoinstruire (AI)				66	
3.8. Total ore pe semestru				150	
3.9. Numărul de credite				6	

4. Precondiţii (acolo unde este cazul)

4.1. de curriculum	-
4.2. de competenţe	-

5. Condiţii (acolo unde este cazul)

5.1. de desfăşurare a cursului	Sală, plus proiector
5.2. de desfăşurare a seminarului/ laboratorului	Laboratoare echipate cu Python

6.1. Competenţele specifice acumulate[1]

Competenţe profesionale/esenţiale	• utilizarea bazelor teoretice ale informaticii şi a modelelor formale • utilizarea instrumentelor informatice în context interdisciplinar • programarea în limbaje de nivel înalt
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 • 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	Studentul cunoaşte: • metodele, algoritmii, paradigmele si tehnicile folosite în diferite ramuri ale informaticii. • utilizarea calculatoarelor, dezvoltarea programelor şi aplicaţiilor software, procesarea informaţiilor.
Aptitudini	Studentul este capabil să: • prezinte şi să explice metodele, algoritmii, paradigmele si tehnicile folosite în diferite ramuri ale informaticii. • folosească paradigme de programare (procedural, orientat obiect, funcţional) pentru realizarea de aplicaţii software adecvate specificului domeniului aplicaţiei dezvoltate. • înţeleagă şi să comunice eficient informaţiile.
Responsabilităţi şi autonomie	Studentul are capacitatea de a lucra independent pentru: • a dezvolta, proiecta şi crea noi aplicaţii, sisteme sau produse folosind bunele practici din domeniu. • conceperea programelor de calculator şi analiza sistemelor software.

7. Obiectivele disciplinei (reieşind din grila competenţelor acumulate)

7.1 Obiectivul general al disciplinei	Să introducă conceptele de baza ale ingineriei software (proiectare, implementare şi întreţinere) şi să prezinte limbajul de programare Python.

[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.

7.2 Obiectivele specifice	• Să introducă conceptele de bază ale programării. • Să introducă conceptele de bază ale ingineriei software. • Să folosească instrumente de bază pentru construirea programelor. • Să prezinte limbajul Python și instrumente de dezvoltare pentru programarea, execuția și depanarea programelor Python. • Să promoveze un stil de programare conform celor mai bune recomandări practice.

8. Conținuturi

8.1 Curs	Metode de predare	Observații
1. Introducere in procese de dezvoltare software • Ce este programarea: algoritm, program, elemente de bază Python, interpretor Python, roluri în ingineria software • Cum scriem programe: enunț problemă, cerințe, proces de dezvoltare dirijat de functionalități (FDD) • Exemple: calculator	• Expunerea interactivă • Explicația • Conversația • Demonstrația didactică	
2. Programare procedurală • Tipuri structurate: liste, tuple, dicționare • Funcții: cazuri de testare, definire, cariabile, apel • Transmiterea parametrilor • Funcții anonime • Cum scriem funcții:programare dirijată de teste, refactorizări	• Expunerea interactivă • Explicația • Conversația • Demonstrația didactică	
3. Programare modulară • Ce este un modul: modul Python, domeniul variabilelor, pachete, module standard, distribuire module • Cum organizăm codul sursă: responsabilități, single responsibility principle, separation of concerns, dependency, coupling, cohesion • Arhitecturi software stratificate • Eclipse+PyDev	• Expunerea interactivă • Explicația • Conversația • Demonstrația didactică	
4. Tipuri definite de utilizator • Cum definim tipuri noi • Încapsulare, ascunderea informației, tipuri abstracte de date	• Expunerea interactivă • Explicația • Conversația • Demonstrația didactică	
5. Principii de proiectare și programare • Problema: program cu operații CRUD pe entități de un tip dat • Arhitectura stratificată: UI, Domeniu, Infrastructura • Șabloane GRASP • Șabloane DDD: entity, validator, repository, controller • Principii: Information Expert, Low Coupling, High Cohesion, Protected Variation, Single responsibility, Dependency Injection	• Expunerea interactivă • Explicația • Conversația • Demonstrația didactică	
6. Programare orientată pe obiecte • Obiecte și clase • Diagrame UML	• Expunerea interactivă • Explicația • Conversația	

• Moştenire • Excepţii	• Demonstraţia didactică	
7. Proiectarea programelor • Top down and bottom up strategies • Organizarea elementelor UI şi relaţia cu alte straturi	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
8. Testarea si inspectarea programelor • Black box testing, white box testing • Unit testing, integration testing • Program inspection: coding style, refactoring	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
9. Recursivitate • Recursivitate directă şi indirectă • Exemple **Complexitatea algoritmilor** • Notaţia asimptotică: big-o, little-o, big-omega, little-omega, theta • Comparaţii algoritmi	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
10. Algoritmi de căutare • căutare secvenţială • căutare binară **Algoritmi de sortare** • BubbleSort • SelectionSort • InsertionSort • QuickSort • MergeSort • Complexitatea algoritmilor	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
11. Backtracking • Algoritmul Backtracking • Extensii ale algoritmului • Exemple	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
12. Divide et Impera • Descriere Metodă • Exemple **Greedy** • Descriere Metoda Exemple	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
13 Programare dinamică • Descriere Metodă • Exemple	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
14. Recapitulare	• Expunerea interactivă • Conversaţia	
Bibliografie 1. Kent Beck.Test Driven Development: By Example. Addison-Wesley Longman, 2002. See also Test-driven development. http://en.wikipedia.org/wiki/Test-driven_development 2. Martin Fowler. Refactoring. Improving the Design of Existing Code. Addison-Wesley, 1999. See also http://refactoring.com/catalog/index.html 3. Frentiu, M., H.F. Pop, Serban G., Programming Fundamentals, Cluj University Press, 2006 4. The Python language reference. http://docs.python.org/py3k/reference/index.html 5. The Python standard library. http://docs.python.org/py3k/library/index.html 6. The Python tutorial. http://docs.python.org/tutorial/index.html		
8.2 Seminar/laborator	Metode de predare	Observaţii
1. Programe Python	• Expunere interactiva	

	• Explicatie • Conversatie • Exemple • Demonstratie didactica	
2. Programare procedurala	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
3. Programare modulara	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
4. Tipuri definite de utilizator	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
5. Principii de proiectare	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
6. POO	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
7. Proiectare	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
8. Testare si inspectare	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
9. Recursivitate	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
10. Complexitatea algoritmilor	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
11. Backtracking	• Expunere interactiva • Explicatie	

	• Conversatie • Exemple • Demonstratie didactica	
12. Metoda injumatatirii. Algoritmi de cautare	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
13. Pregatirea examenului practic	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
14: Pregatirea examenului scris	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	

Bibliografie

1. Kent Beck.Test Driven Development: By Example. Addison-Wesley Longman, 2002. See also Test-driven development. http://en.wikipedia.org/wiki/Test-driven_development
2. Martin Fowler. Refactoring. Improving the Design of Existing Code. Addison-Wesley, 1999. See also http://refactoring.com/catalog/index.html
3. Frentiu, M., H.F. Pop, Serban G., Programming Fundamentals, Cluj University Press, 2006
4. The Python language reference. http://docs.python.org/py3k/reference/index.html
5. The Python standard library. http://docs.python.org/py3k/library/index.html
6. The Python tutorial. http://docs.python.org/tutorial/index.html

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

- Cursul respectă curricula IEEE şi ACM pentru domeniul Informatică.
- Cursul există în programele de studiu ale universităţilor importante din România şi din străinătate.
- Conţinutul disciplinei este considerat de majoritatea companiilor software ca fiind deosebit de important pentru obţinerea unor abilităţi medii de programare

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 Metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	• Cunoştintele acumulate	Examen scris	40%
10.5 Seminar/laborator	• Scrierea unui program	Examen practic	30%
	• Programele scrise in timpul semestrului	Documentatie	30%
10.7 Standard minim de performanţă			

- Fiecare student trebuie să demonstreze că a atins un nivel acceptabil de cunoştinţe şi înţelegere a domeniului, că este capabil să prezinte aceste cunoştinţe într-o manieră coerentă şi că are abilitatea de a stabili anumite conexiuni şi de a folosi aceste cunoştinţe în rezolvarea diferitelor probleme în limbajul de programare Python.
- Pentru promovare, este obligatorie prezenţa la cel puţin 10 seminarii şi 12 laboratoare .
- Promovarea este condiţionată de nota minimă 5 la activitatea de laborator, proba practică şi examenul scris.

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

Nu se aplică.

Data completării:	Semnătura titularului de curs	Semnătura titularului de seminar
.		
10.04.2025	Prof. dr. CZIBULA Istvan Gergely	Prof. dr. CZIBULA Istvan Gergely

Data avizării în departament:	Semnătura directorului de departament
	Conf. dr. Adrian STERCA

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