

FIȘA DISCIPLINEI

Programarea calculatoarelor si limbaje de programare

Anul universitar 2025-2026

1. Date despre program

1.1. Instituția de învățământ superior	Universitatea Babeș-Bolyai Cluj-Napoca
1.2. Facultatea	Facultatea de Matematică și Informatică
1.3. Departamentul	Departamentul de Informatică
1.4. Domeniul de studii	Calculatoare și Tehnologia Informației
1.5. Ciclul de studii	Licență
1.6. Programul de studii / Calificarea	Ingineria Informației (în limba engleză)
1.7. Forma de învățământ	Cu frecvență

2. Date despre disciplină

2.1. Denumirea disciplinei		Programarea calculatoarelor si limbaje de programare				Codul disciplinei	MLE5171
2.2. Titularul activităților de curs		Asist. dr. Anamaria Briciu					
2.3. Titularul activităților de seminar		Asist. dr. Anamaria Briciu					
2.4. Anul de studiu	1	2.5. Semestrul	1	2.6. Tipul de evaluare	C	2.7. Regimul disciplinei	Obligatorie DF

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

3.1. Număr de ore pe săptămână	5	din care: 3.2. curs	2	3.3. seminar/ laborator/proiect	1 S 2 LP
3.4. Total ore din planul de învățământ	70	din care: 3.5. curs	28	3.6 seminar/laborator/proiect	42
Distribuția fondului de timp pentru studiul individual (SI) și activități de autoinstruire (AI)					ore
Studiul după manual, suport de curs, bibliografie și notițe (AI)					18
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					18
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					14
Tutoriat (consiliere profesională)					12
Examinări					18
Alte activități					
3.7. Total ore studiu individual (SI) și activități de autoinstruire (AI)				80	
3.8. Total ore pe semestru				150	
3.9. Numărul de credite				6	

4. Precondiții (acolo unde este cazul)

4.1. de curriculum	
4.2. de competențe	

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Proiector
5.2. de desfășurare a seminarului/ laboratorului	Calculatoare, limbajul și mediul de programare Python

6. Competențele specifice acumulate¹

¹ Se poate opta pentru competențe sau pentru rezultatele învățării, respectiv pentru ambele. În cazul în care se alege o singură variantă, se va șterge tabelul aferent celeilalte opțiuni, iar opțiunea păstrată va fi numerotată cu 6.

Competențe profesionale/esențiale	C1.1 Definirea și descrierea paradigmelor de programare și a mecanismelor specifice limbajului Python, precum și identificarea diferențelor sintactice și semantice. C1.2 Descrierea aplicațiilor software existente la diferite niveluri de abstractizare (arhitectură, clase, metode) utilizând cunoștințele de bază adecvate. C1.3 Elaborarea codului sursă și testarea componentelor într-un limbaj de programare bine cunoscut, pe baza unor specificații date. C1.4 Testarea aplicațiilor pe baza unor planuri de testare. C1.5 Dezvoltarea unităților de programe și a documentației corespunzătoare.
Competențe transversale	TC1 Aplicarea unor reguli de lucru eficiente și riguroase, manifestarea unor atitudini responsabile față de domeniile științific și didactic, valorificând potențialul individual și respectând principiile profesionale și etice. TC2 Utilizarea unor metode și tehnici eficiente pentru învățare, informare, cercetare și dezvoltarea abilităților de valorificare a cunoștințelor, pentru adaptarea la nevoile unei societăți dinamice și pentru comunicarea într-o limbă străină de largă circulație.

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	Cunoașterea conceptelor de bază în inginerie software (proiectare, implementare și mentenanță) și învățarea limbajului de programare Python
7.2 Obiectivele specifice	- Cunoașterea unor concepte cheie de programare - Cunoașterea unor concepte de bază de inginerie software - Înțelegerea uneltelor software de bază folosite în dezvoltarea de programe - Învățarea limbajului de programare Python și uneltelor folosite în dezvoltarea, execuția, testarea și depanarea programelor - Asimilarea și îmbunătățirea unui stil de programare pe baza unor recomandări practice

8. Conținuturi

8.1 Curs	Metode de predare	Observații
1. Introducere în procesele de dezvoltare software - Ce este programarea: algoritm, program, elemente de bază ale limbajului Python, interpretorul Python, roluri de bază în ingineria software - Cum scriem programe: enunțul problemei, cerințe, procese de dezvoltare ghidate de funcționalități	- Expunerea interactivă - Explicația - Conversația - Exemple - Demonstrația didactică	
2. Programare procedurală - Tipuri de date: liste, tuple, dicționar - Funcții: cazuri de test, definiție, scopul variabilelor, transmiterea parametrilor - Dezvoltare ghidată de testare, refactorizare	- Expunerea interactivă - Explicația - Conversația - Exemple - Demonstrația didactică	
3. Programare modulară - Ce este un modul: definirea unui modul Python, scopul variabilelor într-un modul, pachete, librării standard - PyCharm	- Expunerea interactivă - Explicația - Conversația - Exemple - Demonstrația didactică	
4. Tipuri definite de utilizator	Expunerea interactivă	

● Cum definim tipuri noi de date: încapsulare, ascunderea datelor în Python ● Introducere în programarea orientată obiect ● Excepții	● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
5. Programare orientată obiect ● Tipuri abstracte de date ● Implementarea claselor în Python ● Obiecte și clase: clase, obiecte, attribute, metode, scop și namespace în Python	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
6. Principii de proiectare software ● Arhitectura stratificată: interfața utilizator, aplicație, domeniu, infrastructură ● Cum organizăm codul sursă: responsabilități, principiul responsabilității unice, separarea grijilor, dependențe, cuplaj, coeziune	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
7. Testarea și inspecția programelor ● Metode de testare: testare exhaustivă, testare de tip black box, testare de tip white box ● Testare automată, TDD ● Operații cu fișiere în Python	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
8. Recursivitate ● Noțiunea de recursivitate ● Recursivitate directă și indirectă ● Exemple Complexitatea algoritmilor	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
9. Algoritmi de căutare ● Definirea problemei ● Metode de căutare: secvențială, binară ● Complexitatea algoritmilor	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
10. Algoritmi de sortare ● Definirea problemei ● Metode de sortare: Bubble Sort, Selection Sort, Insertion Sort, Quick Sort ● Complexitatea algoritmilor	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
11. Metode de rezolvare a problemelor (I) ● Prezentare generală metode Backtracking, Divide et impera ● Complexitatea algoritmilor ● Exemple	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
12. Metode de rezolvare a problemelor (II) ● Prezentare generală metode Greedy, Programare dinamică ● Complexitatea algoritmilor ● Exemple	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
13. Recapitulare ● Recapitularea celor mai importante concepte prezentate în curs	● Expunerea interactivă ● Explicația ● Conversația ● Exemple ● Demonstrația didactică	
14. Evaluare		
Bibliografie 1. M.L. Hetland, Beginning Python: From Novice to Professional, Apress, Third Edition, 2017. 2. M. Frentiu, H.F. Pop, Fundamentals of Programming, Cluj University Press, 2006. 3. K. Beck, Test Driven Development: By Example. Addison-Wesley Longman, 2002. http://en.wikipedia.org/wiki/Test-driven_development 4. M. Fowler, Refactoring. Improving the Design of Existing Code, Addison-Wesley, 1999.		

<http://refactoring.com/catalog/index.html>

5. The Python Programming Language - <https://www.python.org/>

6. The Python Standard Library - <https://docs.python.org/3/library/>

7. The Python Tutorial - <https://docs.python.org/3/tutorial/>

8.2 Laborator	Metode de predare	Observații
1. Programe Python simple	● Expunerea interactivă ● Explicația ● Conversația ● Demonstrația didactică	
2. Programare procedurală		
3. Programare modulară		
4. Dezvoltare software bazată pe funcționalități		
5. Tipuri abstracte de date		
6. Principii de proiectare		
7. Programare orientată obiect		
8. Proiectarea programelor. Arhitectura stratificată		
9. Inspecție și testare		
10. Recursivitate. Complexitatea algoritmilor		
11. Algoritmi de căutare și sortare		
12. Metode de rezolvare a problemelor: Backtracking		
13. Metode de rezolvare a problemelor: Greedy		
14. Test practic		

Bibliografie

1. M.L. Hetland, Beginning Python: From Novice to Professional, Apress, Third Edition, 2017.

2. M. Frentiu, H.F. Pop, Fundamentals of Programming, Cluj University Press, 2006.

3. K. Beck, Test Driven Development: By Example. Addison-Wesley Longman, 2002. http://en.wikipedia.org/wiki/Test-driven_development

4. M. Fowler, Refactoring. Improving the Design of Existing Code, Addison-Wesley, 1999.

<http://refactoring.com/catalog/index.html>

5. The Python Programming Language - <https://www.python.org/>

6. The Python Standard Library - <https://docs.python.org/3/library/>

7. The Python Tutorial - <https://docs.python.org/3/tutorial/>

8.3. Seminar	Metode de predare	Observații
1. Programe Python simple. Programare procedurală	● Expunerea interactivă ● Explicația ● Conversația ● Demonstrația didactică	
2. Programare modulară. Dezvoltare software bazată pe funcționalități.		
3. Tipuri abstracte de date. Principii de proiectare.		
4. Programare orientată obiect. Proiectarea programelor. Arhitectura stratificată.		
5. Inspecție și testare. Recursivitate. Complexitatea algoritmilor.		
6. Algoritmi de căutare și sortare. Metode de rezolvare a problemelor: Backtracking		
7. Metode de rezolvare a problemelor: Greedy		

Bibliografie

1. M.L. Hetland, Beginning Python: From Novice to Professional, Apress, Third Edition, 2017.

2. M. Frentiu, H.F. Pop, Fundamentals of Programming, Cluj University Press, 2006.

3. K. Beck, Test Driven Development: By Example. Addison-Wesley Longman, 2002. http://en.wikipedia.org/wiki/Test-driven_development

4. M. Fowler, Refactoring. Improving the Design of Existing Code, Addison-Wesley, 1999.

<http://refactoring.com/catalog/index.html>

5. The Python Programming Language - <https://www.python.org/>

6. The Python Standard Library - <https://docs.python.org/3/library/>

7. The Python Tutorial - <https://docs.python.org/3/tutorial/>

9. Coroborarea conținuturilor disciplinei cu așteptările reprezentanților comunității epistemice, asociațiilor profesionale și angajatori reprezentativi din domeniul aferent programului

- Cursul respectă recomandările curriculare IEEE și ACM pentru studii în informatică
- Cursul există în programele de studiu ale marilor universități din România și străinătate
- Conținutul cursului este considerat de companiile software ca fiind important pentru abilități 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	Corectitudinea și completitudinea cunoștințelor acumulate și capacitatea de a proiecta și implementa corect programe Python	Examen scris	40%	
10.5 Laborator	Abilitatea de a proiecta, implementa și testa programe Python	Examen practic	30%	
	Corectitudinea temelor de laborator și documentația livrată în timpul semestrului	Program și documentație	30%	
10.6. Seminar	Activitate de seminar	Participare activă la discuțiile din timpul seminariilor (punerea de întrebări, răspuns la întrebările cadrului didactic, rezolvarea de probleme, etc)	Maximum 0.5 puncte bonus, adăugate notei finale	
10.6 Standard minim de performanță				
● Fiecare student trebuie să demonstreze un nivel acceptabil de cunoștințe și înțelegere a domeniului, capacitatea de a prezenta cunoștințele într-o manieră coerentă și abilitatea de a stabili conexiuni și de a folosi aceste cunoștințe pentru a rezolva diverse probleme în Python. ● Fiecare student este obligat să participe la minimum 5 seminarii și 12 laboratoare. ● Trebuie obținută o notă minimă de 5 la activitatea de laborator, testul practic și examenul scris.				

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

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".

Nu se aplică.

Data completării:
22.09.2025

Semnătura titularului de curs

Asist. dr. Anamaria Briciu

Semnătura titularului de seminar

Asist. dr. Anamaria Briciu

Data avizării în departament:
...

Semnătura directorului de departament

Conf.dr. Adrian STERCA