

FIȘA DISCIPLINEI

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	Matematică-Informatică
1.5 Ciclul de studii	Licență
1.6 Programul de studiu / Calificarea	Matematică Informatică – limba de studiu română

2. Date despre disciplină

2.1 Denumirea disciplinei (ro)	Algoritmi și Programare						
(en)							
2.2 Titularul activităților de curs	Lect. Dr. Găceanu Radu Dan						
2.3 Titularul activităților de seminar	Lect. Dr. Găceanu Radu Dan						
2.4 Anul de studiu	1	2.5 Semestrul	1	2.6. Tipul de evaluare	C	2.7 Regimul disciplinei	Obligatorie
2.8 Codul disciplinei	MLR5005						

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

3.1 Număr de ore pe săptămână	6	Din care: 3.2 curs	2	3.3 seminar/laborator	2 sem 2 lab
3.4 Total ore din planul de învățământ	84	Din care: 3.5 curs	28	3.6 seminar/laborator	56
Distribuția fondului de timp:					ore
Studiul după manual, suport de curs, bibliografie și notițe					14
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					12
Pregătire seminarii/laboratoare, teme, referate, portofolii și eseuri					14
Tutoriat					8
Examinări					18
Alte activități:					
3.7 Total ore studiu individual	66				
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	• Sală, plus proiector
5.2 De desfășurare a seminarului/laboratorului	• Laboratoare echipate cu medii de dezvoltare Python; sala cu proiector (seminar)

6. Competențele specifice acumulate

Competențe profesionale	- C1.1 Descrierea adecvată a paradigmelor de programare și a mecanismelor de limbaj specifice, precum și identificarea diferenței dintre aspectele de ordin semantic și sintactic. - C1.2 Explicarea unor aplicații soft existente, pe niveluri de abstractizare (arhitectură, pachete, clase, metode) utilizând în mod adecvat cunoștințele de bază - C1.3 Elaborarea codurilor sursă adecvate și testarea unitară a unor componente într-un limbaj de programare cunoscut, pe baza unor specificații de proiectare date - C1.4 Testarea unor aplicații pe baza unor planuri de test - C1.5 Dezvoltarea de unități de program și elaborarea documentațiilor aferente
Competențe transversale	- CT1 Aplicarea regulilor de muncă organizată și eficientă, a unor atitudini responsabile față de domeniul didactic-științific, pentru valorificarea creativă a propriului potențial, cu respectarea principiilor și a normelor de etică profesională - CT3 Utilizarea unor metode și tehnici eficiente de învățare, informare, cercetare și dezvoltare a capacităților de valorificare a cunoștințelor, de adaptare la cerințele unei societăți dinamice și de comunicare în limba română și într-o limbă de circulație internațională

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	Să cunoască conștepele de baza ale ingineriei software (proiectare, implementare și întreținere) și să învețe limbajul de programare Python.
7.2 Obiectivele specifice	- Să cunoască conștepele de baza ale programarii - Să cunoască conștepele de baza ale ingineriei software - Să folosească instrumente de baza pentru construirea programelor - Să învețe limbajul Python si instrumente de dezvoltare pentru programarea, executia si depanarea programelor Python. - Să-și însușeasca un stil de programare conform celor mai bune recomandări practice.

8. Conținuturi

8.1 Curs	Metode de predare	Observații
1. Introducere în 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 funcționalități - Exemple: calculator	- Expunere interactivă - Explicație - Conversație - Exemple - Demonstrație didactică	
2. Programare procedurală - Tipuri structurate: liste, tuple, dicționare - Functii: cazuri de testare, definire, variabile, apel, transmiterea parametrilor, functii anonime - Cum scriem functii:programare dirijata de teste, refactorizari	- Expunere interactivă - Conversație - Exemple - Demosntrație didactică	

3. Programare modulara • Ce este un modul: modul Python, domeniul variabilelor, pachete, module standard, distribuire module • Cum organizam codul sursa: responsabilitati, single responsibility principle, separation of concerns, dependency, coupling, cohesion • Eclipse+PyDev	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
4. Tipuri definite de utilizator • Cum definim tipuri noi • Incapsulare, ascunderea informatiei, tipuri abstracte de date • Introducere programarea orientata obiect –clasa implementarea unui TAD	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
5. Programarea orientata obiect • Mecanisme programarii orientate obiect • Dezideratele proiectarii obiectuale ○ Cuplare redusa ○ Coeziune ridicata ○ Abstractizare corespunzatoare ○ Complexitate gestionabila	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
6. Programarea orientata obiect • Principii si sabloane de proiectare: ○ SOLID: single responsibility, open-closed.... ○ DDD: entity, value object, repository, service ○ IoC, app coordinator, dependency injection ○ GOF: singleton, strategy • Problema: program cu operatii CRUD pe entitati de un tip dat • Arhitectura stratificata: UI, Domeniu, Infrastructura	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
7. Proiectarea programelor • Diagrame UML • Sabloane Grasp • Exceptii	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
8. Persistenta și procesarea datelor • DTO • Filter, lambda	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
9. Testarea si inspectarea programelor • Black box testing, white box testing • Unit testing, integration testing	• Expunere interactivă • Explicație	

Program inspection: coding style, refactoring	- Conversație - Exemple - Demonstrație didactică	
10. Complexitatea Algoritmilor, recursivitate, căutare - Recursivitate directă și indirectă - Exemple - Căutare secvențială - Căutare binară - Complexitatea algoritmilor	- Expunere interactivă - Explicație - Conversație - Exemple - Demonstrație didactică	
11 Algoritmi de sortare - BubbleSort - SelectionSort - InsertionSort - QuickSort - MergeSort - Complexitatea algoritmilor	- Expunere interactivă - Explicație - Conversație - Exemple - Demonstrație didactică	
12. Metode de rezolvare a problemelor Metoda divizării - Descriere Metoda - Exemple Metoda Backtracking - Algoritmul Backtracking - Extensii ale algoritmului	- Expunere interactivă - Explicație - Conversație - Exemple - Demonstrație didactică	
13. Programare dinamică - Descrierea metodei - Exemple	- Expunere interactivă - Conversație	
14. Colocviu - Evaluare		
Bibliografie - Kent Beck. <i>Test Driven Development: By Example</i>. Addison-Wesley Longman, 2002. See also Test-driven development. http://en.wikipedia.org/wiki/Test-driven_development - Martin Fowler. <i>Refactoring. Improving the Design of Existing Code</i>. Addison-Wesley, 1999. See also http://refactoring.com/catalog/index.html - Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. <i>Introduction to Algorithms 3rd ed.</i>, 2009. - Craig Larman. <i>Applying UML and Patterns. An Introduction to Object Oriented Analysis and Design</i>, 2004. <i>The Python language reference</i>. http://docs.python.org/py3k/reference/index.html <i>The Python standard library</i>. http://docs.python.org/py3k/library/index.html <i>The Python tutorial</i> . http://docs.python.org/tutorial/index.html		
8.2 Seminar / laborator	Metode de predare	Observații
1. Programe Python	- Expunere interactivă - Explicație - Conversație - Exemple - Demonstrație didactică	
2. Programare procedurală	- Expunere interactivă - Explicație - Conversație	

	• Exemple • Demonstrație didactică	
3. Programare modulară	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
4. Tipuri definite de utilizator	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
5. Principii de proiectare	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
6. POO	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
7. Proiectare	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
8. Testare și inspectare	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
9. Recursivitate. Complexitatea algoritmilor	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
10. Metoda înjumătățirii. Algoritmi de căutare	• Expunere	

	interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
11. Algoritmi de sortare	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
12. Backtracking	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
13. Pregătirea examenului practic	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
14: Pregătirea examenului scris	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	

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>
- The Python tutorial*. <http://docs.python.org/tutorial/index.html>

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

- Cursul respecta recomandările IEEE și ACM legate de Curricula pentru specializarea Informatică.
- Cursul face parte din programul de studiu de la majoritatea universităților importante din România și din străinătate.
- Conținutul cursului este considerat de companiile soft ca fiind important pentru un nivel mediu de cunoștințe în 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. Capacitatea de a proiecta și implementa programe scrise în limbajul Python	Examen scris	20%
10.5 Seminar/laborator	Abilitatea de a scrie și depana un program Python	Examen practic	60%
	Programele scrise în timpul semestrului	Documentație	20%
10.6 Standard minim de performanță			
Nota finală minim 5.			

Data completării

06.05.2023

Semnătura titularului de curs

Lect. Dr. Găceanu Radu Dan

Semnătura titularului de seminar

Lect. Găceanu Radu Dan

Data avizării în departament

Semnătura directorului de departament

Prof. dr. Laura Dioșan