FIŞA DISCIPLINEI

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	**Calculatoare şi Tehnologia Informaţiei**
1.5 Ciclul de studii	**Licenţă**
1.6 Programul de studiu / Calificarea	**Ingineria Informaţiei (în limba engleză)**

2. Date despre disciplină

2.1 Denumirea disciplinei (ro) (en)	**Programarea calculatoarelor si limbaje de programare** **Computer programming and programming languages**						
2.2 Titularul activităţilor de curs	**Prof. dr. Camelia Chira**						
2.3 Titularul activităţilor de seminar	**Prof. dr. Camelia Chira**						
2.4 Anul de studiu	**1**	2.5 Semestrul	**1**	2.6. Tipul de evaluare	**C**	2.7 Regimul disciplinei	Obligatorie DF
2.8 Codul disciplinei	MLE5171						

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	2 LP 1S
3.4 Total ore din planul de învăţământ	70	Din care: 3.5 curs	28	3.6 seminar/laborator	42
Distribuţia fondului de timp:					ore
Studiul după manual, suport de curs, bibliografie şi notiţe					18
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate şi pe teren					18
Pregătire seminarii/laboratoare, teme, referate, portofolii şi eseuri					14
Tutoriat					12
Examinări					18
Alte activităţi:					

3.7 Total ore studiu individual	80
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	• Proiector
5.2 De desfăşurare a seminarului/laboratorului	• Calculatoare, limbajul şi mediul de programare Python

6. Competenţele specifice acumulate

Competenţe profesionale	C1.1 Recunoașterea şi descrierea conceptelor proprii calculabilității, complexității, paradigmelor de programare şi modelării sistemelor de calcul şi comunicații C1.2 Utilizarea de teorii şi instrumente specifice (algoritmi, scheme, modele, protocoale etc.) pentru explicarea structurii şi funcționării sistemelor hardware, software şi de comunicații C2.1 Descrierea structurii şi funcționării componentelor hardware, software şi de comunicații C2.3 Construirea unor componente hardware, software şi de comunicații folosind metode de proiectare, limbaje, algoritmi, structuri de date, protocoale şi tehnologii C2.5 Implementarea componentelor sistemelor hardware, software şi de comunicație
Competenţe transversale	CT1 Comportarea onorabilă, responsabilă, etică, în spiritul legii pentru a asigura reputația profesiei CT3 Demonstrarea spiritului de inițiativă şi acțiune pentru actualizarea cunoştințelor profesionale, economice şi de cultură organizațională

7. Obiectivele disciplinei (reieşind din grila competenţelor acumulate)

7.1 Obiectivul general al disciplinei	• Cunoaşterea conceptelor de bază în inginerie software (proiectare, implementare şi mentenanţă) şi învăţarea limbajului de programare Python
7.2 Obiectivele specifice	• Cunoaşterea unor concepte cheie de programare • Cunoaşterea unor concepte de bază de inginerie software • Înţelegerea uneltelor software de bază folosite în dezvoltarea de programe • Învăţarea limbajului de programare Python şi uneltelor folosite în dezvoltarea, execuţia, testarea şi depanarea programelor • Asimilarea şi îmbunătăţirea unui stil de programare pe baza unor recomandări practice

8. Conţinuturi

8.1 Curs	Metode de predare	Observaţii
1. Introducere în procesele de dezvoltare software • Ce este programarea: algoritm, program, elemente de bază ale limbajului Python, interpretorul Python, roluri de bază în	• Expunerea interactivă • Explicaţia • Conversaţia	

ingineria software • Cum scriem programe: enunţul problemei, cerinţe, procese de dezvoltare ghidate de funcţionalităţi	• Exemple • Demonstraţia didactică	
2. Programare procedurală • Tipuri de date: liste, tuple, dicţionar • Funcţii: cazuri de test, definiţie, scopul variabilelor, transmiterea parametrilor • Dezvoltare ghidată de testare, refactorizare	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
3. Programare modulară • Ce este un modul: definirea unui modul Python, scopul variabilelo într-un modul, pachete, librării standard • Eclipse + PyDev	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
4. Tipuri definite de utilizator • Cum definim tipuri noi de date: încapsulare, ascunderea datelor în Python, sugestii • Introducere în programarea orientată obiect • Excepţii	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
5. Programare orientată obiect • Tipuri abstracte de date • Implementarea claselor în Python • Obiecte şi clase: clase, obiecte, atribute, metode, scop şi namespace în Python	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
6. Principii de proiectare software • Arhitectura stratificată: intefaţa utilizator, aplicaţie, domeniu, infrastructură • Cum organizăm codul sursă: responsabilităţi, principiul responsabilităţii unice, separarea grijilor, dependenţe, cuplaj, coeziune	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
7. Testarea şi inspecţia programelor • Metode de testare: testare exhaustivă, testare de tip black box, testare de tip white box • Testare automată, TDD • Operaţii cu fişiere în Python	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
8. Recursivitate • Noţiunea de recursivitate • Recursivitate directă şi indirectă • Exemple • Complexitate	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
9. Algoritmi de căutare • Definirea problemei • Metode de căutare: secvenţială, binară • Complexitatea algoritmilor	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
10. Algoritmi de sortare • Definirea problemei • Metode de sortare: Bubble Sort, Selection Sort, Insertion Sort, Quick Sort	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple	

• Complexitatea algoritmilor	• Demonstraţia didactică	
11. Metode de rezolvare a problemelor I • Prezentare generală metode Backtracking, Divide et impera • Complexitatea algoritmilor • Exemple	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
12. Metode de rezolvare a problemelor II • Prezentare generală metode Greedy, Programare dinamică • Complexitatea algoritmilor • Exemple	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia didactică	
13. Recapitulare • Recapitularea celor mai importante concepte prezentate în curs	• Expunerea interactivă • Explicaţia • Conversaţia • Exemple • Demonstraţia 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/index.html
7. The Python Tutorial - https://docs.python.org/3/tutorial/

8.2 Llaborator	Metode de predare	Observaţii
1. Programe Python simple	• Expunerea interactivă • Explicaţia • Conversaţia • Demonstraţia didactică	
2. Programare procedurală		
3. Programare modulară		
4. Dezvoltare software bazată pe funcţionalităţi		
5. Tipuri abstracte de date		
6. Principii de proiectare		
7. Programare orientată obiect		
8. Proiectarea programelor. Arhitectura stratificată		
9. Inspecţie şi testare		
10. Recursivitate. Complexitatea algoritmilor		
11. Algoritmi de căutare şi 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.

<table>
<tr><td colspan="3">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/index.html
7. The Python Tutorial - https://docs.python.org/3/tutorial/</td></tr>
<tr><td>8.3 Seminar</td><td>Metode de predare</td><td>Observaţii</td></tr>
<tr><td>1. Programe Python simple. Programare procedurală</td><td rowspan="8">• Expunerea interactivă
• Explicaţia
• Conversaţia
• Demonstraţia didactică</td><td></td></tr>
<tr><td>2. Programare modulară. Dezvoltare software bazată pe funcţionalităţi.</td><td></td></tr>
<tr><td>3. Tipuri abstracte de date. Principii de proiectare.</td><td></td></tr>
<tr><td>4. Programare orientată obiect. Proiectarea programelor. Arhitectura stratificată.</td><td></td></tr>
<tr><td>5. Inspecţie şi testare. Recursivitate. Complexitatea algoritmilor.</td><td></td></tr>
<tr><td>6. Algoritmi de căutare şi sortare. Metode de rezolvare a problemelor: Backtracking</td><td></td></tr>
<tr><td>7. Metode de rezolvare a problemelor: Greedy</td><td></td></tr>
<tr><td colspan="3">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/index.html
7. The Python Tutorial - https://docs.python.org/3/tutorial/</td></tr>
</table>

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

- Cursul respectă recomandările curiculare IEEE şi ACM pentru studii în informatică
- Cursul există în programele de studiu ale marilor universităţi din România şi străinătate
- Conţinutul cursului este considerat de companiile software ca fiind imporant pentru 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	Corectitutdinea şi completitudinea cunoştinţelor acumulate şi capacitatea de a proiecta şi implementa corect programe Python	Examen scris	40%
10.5 Seminar/laborator	Abilitatea de a proiecta, implementa şi testa un programe Python	Examen practic	30%

	Corectitudinea temelor de laborator şi documentaţiă livrată în timpul semestrului	Program şi documentaţie	30%

10.6 Standard minim de performanţă

- Fiecare student trebuie să demonstreze că a atins un nivel acceptabil de cunoaştere şi înţelegere a domeniului, că este capabil să exprime cunoştinţele într-o formă coerentă şi să le utilizeze în rezolvarea unor probleme.
- Fiecare student trebuie să obţină minim nota 5 pentru examenul practic, examenul scris şi nota finală.

Data completării	Semnătura titularului de curs	Semnătura titularului de seminar
9.05.2022	Prof. dr. Camelia Chira	Prof. dr. Camelia Chira

Data avizării în departament

Semnătura directorului de departament

Prof. dr. Laura Dioşan

24.05.2022