FIŞA DISCIPLINEI

1. Date despre program

1.1 Instituţia de învăţământ superior	**Universitatea Babeş-Bolyai din Cluj-Napoca**
1.2 Facultatea	**Facultatea de Matematică şi Infrormatică**
1.3 Departamentul	**Departamentul de Informatică**
1.4 Domeniul de studii	**Informatică**
1.5 Ciclul de studii	**Licenţă**
1.6 Programul de studiu / Calificarea	**Informatică română**

2. Date despre disciplină

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

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 sem 2 lab
3.4 Total ore din planul de învăţământ	84	Din care: 3.5 curs	28	3.6 seminar/laborator	56

Distribuţia fondului de timp:	ore
Studiul după manual, suport de curs, bibliografie şi notiţe	14
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate şi pe teren	12
Pregătire seminarii/laboratoare, teme, referate, portofolii şi eseuri	14
Tutoriat	8
Examinări	18
Alte activităţi:	-

3.7 Total ore studiu individual	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. Competenţele specifice acumulate

Competenţe profesionale	C1.1 Descrierea adecvată a paradigmelor de programare şi a mecanismelor de limbaj specifice, precum şi identificarea diferenţei dintre aspectele de ordin semantic şi sintactic. C1.2 Explicarea unor aplicaţii soft existente, pe niveluri de abstractizare (arhitectură, pachete, clase, metode) utilizând in mod adecvat cunoştinţele de bază C1.3 Elaborarea codurilor sursă adecvate şi testarea unitară a unor componente într-un limbaj de programare cunoscut, pe baza unor specificaţii 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
Competenţe transversale	CT1 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 CT3 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 și de comunicare în limba română și într-o limbă de circulație internațională

7. Obiectivele disciplinei (reieşind din grila competenţelor acumulate)

7.1 Obiectivul general al disciplinei	• Sa introduca conceptele de baza ale ingineriei software (proiectare, implementare si intretinere) si sa prezinte limbajul de programare Python.
7.2 Obiectivele specifice	• Sa introduca conceptele de baza ale programarii • Sa introduca conceptele de baza ale ingineriei software • Sa foloseasca instrumente de baza pentru construirea programelor • Sa prezinte limbajul Python si instrumente de dezvoltare pentru programarea, executia si depanarea programelor Python. • Sa promoveze un stil de programare conform celor mai bune recomandari practice.

8. Conţinuturi

8.1 Curs	Metode de predare	Observaţii
1. Introducere in procese de dezvoltare software • Ce este programarea: algoritm, program, elemente de baza Python, interpretor Python, roluri in ingineria software • Cum scriem programe: enunt problema, cerinte, proces de dezvoltare dirijat de functionalitati (FDD) • Exemple: calculator	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
2. Programare procedurala • Tipuri structurate: liste, tuple, dictionare • Functii: cazuri de testare, definire, cariabile, apel • Transmiterea parametrilor • Functii anonime	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	

• Cum scriem functii:programare dirijata de teste, refactorizari		
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 • Arhitecturi software stratificate • Eclipse+PyDev	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
4. Tipuri definite de utilizator • Cum definim tipuri noi • Incapsulare, ascunderea informatiei, tipuri abstracte de date	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
5. Principii de proiectare si programare • Problema: program cu operatii CRUD pe entitati de un tip dat • Arhitectura stratificata: UI, Domeniu, Infrastructura • Sabloane GRASP • Sabloane DDD: entity, validator, repository, controller • Principii: Information Expert, Low Coupling, High Cohesion, Protected Variation, Single responsibility, Dependency Injection	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
6. Programare orientata pe obiecte • Obiecte si clase • Diagrame UML • Mostenire • Exceptii	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
7. Proiectarea programelor • Top down and bottom up strategies: • Organizarea elementelor UI si relatia cu alte straturi	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
8. Testarea si inspectarea programelor • Black box testing, white box testing • Unit testing, integration testing • Program inspection: coding style, refactoring	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
9. Recursivitate • Recursivitate directa si indirecta • Exemple **Complexitatea algoritmilor** • Notatia asimptotica: big-o, little-o, big-omega, little-omega, theta • Comparatii algoritmi	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
10. Algoritmi de cautare • cautare secventiala	• Expunere interactiva • Explicatie	

cautare binara **Algoritmi de sortare** • BubbleSort • SelectionSort • InsertionSort • QuickSort • MergeSort • Complexitatea algoritmilor	• Conversatie • Exemple • Demonstratie didactica	
11. Backtracking • Algoritmul Backtracking • Extensii ale algoritmului • Exemple	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
12. Divide et Impera • Descriere Metoda • Exemple **Greedy** • Descriere Metoda • Exemple	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
13 Programare dinamica • Descriere Metoda • Exemple	• Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica	
14. Recapitulare	• Expunere interactiva • Conversatie	

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	

Bibliography

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 și ACM pentru domeniul Informatică.
- Cursul există în programele de studiu ale universităților importante din România și din străinătate.
- Conținutul disciplinei este considerat de majoritatea companiilor software ca fiind deosebit de important pentru obținerea unor 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	Cunoștintele acumulate	Examen scris	30%

10.5 Seminar/laborator	Scrierea unui program	Simulare examen practic	10%
	Scrierea unui program	Examen practic	30%
	Programele scrise in timpul semestrului	Documentatie	30%

10.6 Standard minim de performanță

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

Data completării	Semnătura titularului de curs	Semnătura titularului de seminar
20.04.2022	Prof. dr. Istvan Gergely Czibula	Prof. dr. Istvan Gergely Czibula

Data avizării în departament

Semnătura directorului de departament

Prof. dr. Dioșan Laura