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 Infromatică
1.3 Departamentul	Departamentul de Informatică
1.4 Domeniul de studii	Informatică
1.5 Ciclul de studii	Informatică 3 ani
1.6 Programul de studiu / Calificarea	Informatică

2. Date despre disciplină

2.1 Denumirea disciplinei	Fundamentele programării						
2.2 Titularul activităţilor de curs	Lect. Dr. Camelia Şerban						
2.3 Titularul activităţilor de seminar	Lect. Dr. Camelia Şerban						
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	• Să cunoască concepele de baza ale ingineriei software (proiectare, implementare și intreținere) și să invețe limbajul de programare Python.
7.2 Obiectivele specifice	• Să cunoască conceptele de baza ale programarii • Să cunoască conceptele 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 însușească un stil de programare conform celor mai bune recomandări 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 interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
2. Programare procedurala • Tipuri structurate: liste, tuple, dictionare	• Expunere interactivă	

• Functii: cazuri de testare, definire, cariabile, apel • Transmiterea parametrilor • Functii anonime • Cum scriem functii:programare dirijata de teste, refactorizari	• Explicație • 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 • Arhitecturi software stratificate • 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	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
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 interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
6. Programare orientata pe obiecte • Obiecte si clase • Diagrame UML • Mostenire • Exceptii	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
7. Proiectarea programelor • Top down and bottom up strategies: • Organizarea elementelor UI si relatia cu alte straturi	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
8. Testarea si inspectarea programelor • Black box testing, white box testing • Unit testing, integration testing • Program inspection: coding style, refactoring	• Expunere interactivă • Explicație • Conversație • Exemple	

	• Demosntrație didactică	
9. Recursivitate • Recursivitate directa si indirecta • Exemple	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
10. Complexitatea algoritmilor • Notatia asimptotica: big-o, little-o, big-omega, little-omega, theta • Comparatii algoritmi	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
11. Metoda Backtracking • Algoritmul Backtracking • Extensii ale algoritmului • Exemple	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
12. Metoda divizarii • Descriere Metoda • Exemple Algoritmi de cautare • catuatare secventiala • cautare binara • complexitatea algoritmilor	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
13 Algoritmi de sortare • BubbleSort • SelectionSort • InsertionSort • QuickSort • MergeSort • Cmplexitatea algoritmilor	• Expunere interactivă • Explicație • Conversație • Exemple • Demosntrație didactică	
14. Recapitulare	• Expunere interactivă • Conversație	

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

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	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
10. Complexitatea algoritmilor	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
11. Backtracking	• Expunere interactivă • Explicație • Conversație • Exemple • Demonstrație didactică	
12. Metoda injumătățirii. Algoritmi de căutare	• 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 conţinuturilor disciplinei cu aşteptările reprezentanţilor comunităţii epistemice, asociaţiilor profesionale şi angajatori reprezentativi din domeniul aferent programului

- Cursul respecta recomandările IEEE şi ACM legate de Curiculla 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 in limbajul Python	Examen scris	40%
10.5 Seminar/laborator	Abilitatea de a scrie și depana un program Python	Examen practic	30%
	Programele scrise în timpul semestrului	Documentație	30%
10.6 Standard minim de performanţă			
• Minimum 5 la fiecare proba.			

Data completării	Semnătura titularului de curs	Semnătura titularului de seminar
29.04.2014	Lect. Dr. Camelia Şerban	Lect. Dr. Camelia Şerban

Data avizării în departament

Semnătura directorului de departament

Prof. dr. Bazil Pârv