

FIŞA DISCIPLINEI

1. Date despre program

1.1 Instituția de învățământ superior	Universitatea Babeș-Bolyai din Cluj-Napoca
1.2 Facultatea	Facultatea de Matematică și Informatică
1.3 Departamentul	Departamentul de Informatică
1.4 Domeniul de studii	Informatică
1.5 Ciclul de studii	Licență
1.6 Programul de studiu / Calificarea	Informatică română

2. Date despre disciplină

2.1 Denumirea disciplinei	Fundamentele programării					
2.2 Titularul activităților de curs	Prof. dr. Istvan Gergely Czibula					
2.3 Titularul activităților 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
2.8 Cod disciplină	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	<ul style="list-style-type: none"> • Sală, plus proiectoare
5.2 De desfășurare a seminarului/laboratorului	<ul style="list-style-type: none"> • Laboratoare echipate cu Python

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 capacitațiilor 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 (reiesind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	<ul style="list-style-type: none"> • Sa introduca concepțile de baza ale ingineriei software (proiectare, implementare și întreținere) și să prezinte limbajul de programare Python.
7.2 Obiectivele specifice	<ul style="list-style-type: none"> • Sa introduca concepțile de baza ale programării • Sa introduca concepțile de baza ale ingineriei software • Sa foloseasca instrumente de baza pentru construirea programelor • Sa prezinte limbajul Python și instrumente de dezvoltare pentru programarea, execuția și depanarea programelor Python. • Sa promoveze un stil de programare conform celor mai bune recomandari practice.

8. Conținuturi

8.1 Curs	Metode de predare	Observații
1. Introducere în procese de dezvoltare software <ul style="list-style-type: none"> • Ce este programarea: algoritm, program, elemente de baza Python, interpretor Python, roluri în ingineria software • Cum scriem programe: enunț problema, cerințe, proces de dezvoltare dirijat de funcționalități (FDD) • Exemple: calculator 	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
2. Programare procedurală <ul style="list-style-type: none"> • Tipuri structurate: liste, tuple, dictionare 	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie 	

<ul style="list-style-type: none"> • Functii: cazuri de testare, definire, variabile, apel • Transmiterea parametrilor • Functii anonte • Cum scriem functii: programare dirijata de teste, refactorizari 	<ul style="list-style-type: none"> • Conversatie • Exemple • Demonstratie didactica 	
3. Programare modulara <ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
4. Tipuri definite de utilizator <ul style="list-style-type: none"> • Cum definim tipuri noi • Incapsulare, ascunderea informatiei, tipuri abstracte de date 	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
5. Principii de proiectare si programare <ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
6. Programare orientata pe obiecte <ul style="list-style-type: none"> • Obiecte si clase • Diagrame UML • Mostenire • Exceptii 	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
7. Proiectarea programelor <ul style="list-style-type: none"> • Top down and bottom up strategies: • Organizarea elementelor UI si relatia cu alte straturi 	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
8. Testarea si inspectarea programelor <ul style="list-style-type: none"> • Black box testing, white box testing • Unit testing, integration testing • Program inspection: coding style, refactoring 	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
9. Recursivitate <ul style="list-style-type: none"> • Recursivitate directa si indirecta • Exemple Complexitatea algoritmilor <ul style="list-style-type: none"> • Notatia asymptotica: big-o, little-o, big-omega, little-omega, theta 	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	

<ul style="list-style-type: none"> Comparatii algoritmi <p>10. Algoritmi de cautare</p> <ul style="list-style-type: none"> cautare seventionala cautare binara <p>Algoritmi de sortare</p> <ul style="list-style-type: none"> BubbleSort SelectionSort InsertionSort QuickSort MergeSort Complexitatea algoritmilor 	<ul style="list-style-type: none"> Expunere interactiva Explicatie Conversatie Exemple Demonstratie didactica 	
<p>11. Backtracking</p> <ul style="list-style-type: none"> Algoritmul Backtracking Extensii ale algoritmului Exemple 	<ul style="list-style-type: none"> Expunere interactiva Explicatie Conversatie Exemple Demonstratie didactica 	
<p>12. Divide et Impera</p> <ul style="list-style-type: none"> Descriere Metoda Exemple <p>Greedy</p> <ul style="list-style-type: none"> Descriere Metoda Exemple 	<ul style="list-style-type: none"> Expunere interactiva Explicatie Conversatie Exemple Demonstratie didactica 	
<p>13 Programare dinamica</p> <ul style="list-style-type: none"> Descriere Metoda Exemple 	<ul style="list-style-type: none"> Expunere interactiva Explicatie Conversatie Exemple Demonstratie didactica 	
<p>14. Recapitulare</p>	<ul style="list-style-type: none"> Expunere interactiva Conversatie 	

Bibliografie

- Kent Beck. *Test Driven Development: By Example*. Addison-Wesley Longman, 2002. See also Test-driven development. http://en.wikipedia.org/wiki/Test-driven_development
- Martin Fowler. *Refactoring. Improving the Design of Existing Code*. Addison-Wesley, 1999. See also <http://refactoring.com/catalog/index.html>
- Frentiu, M., H.F. Pop, Serban G., Programming Fundamentals, Cluj University Press, 2006
- The Python language reference*. <http://docs.python.org/py3k/reference/index.html>
- 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	Observatii
1. Programe Python	<ul style="list-style-type: none"> Expunere interactiva Explicatie Conversatie Exemple Demonstratie didactica 	
2. Programare procedurala	<ul style="list-style-type: none"> Expunere interactiva Explicatie Conversatie Exemple Demonstratie didactica 	

3. Programare modulară	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
4. Tipuri definite de utilizator	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
5. Principii de proiectare	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
6. POO	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
7. Proiectare	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
8. Testare si inspectare	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
9. Recursivitate	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
10. Complexitatea algoritmilor	<ul style="list-style-type: none"> • Expunere interactivă • Explicatie • Conversatie 	

	<ul style="list-style-type: none"> • Exemple • Demonstratie didactica 	
11. Backtracking	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
12. Metoda injumatatirii. Algoritmi de cautare	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
13. Pregatirea examenului practic	<ul style="list-style-type: none"> • Expunere interactiva • Explicatie • Conversatie • Exemple • Demonstratie didactica 	
14: Pregatirea examenului scris	<ul style="list-style-type: none"> • 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 conținuturilor disciplinei cu așteptările reprezentanților comunității epistemice, asociațiilor profesionale și angajatorii 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	Cunoștințele acumulate	Examen scris	40%
10.5 Seminar/laborator	Scrierea unui program	Examen practic	30%
	Programele scrise în timpul semestrului	Documentatie	30%
10.6 Standard minim de performanță			
<ul style="list-style-type: none"> • 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 cel puțin 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

Prof. dr. Czibula Istvan Gergely

Prof. dr. Czibula Istvan Gergely

Data avizării în departament

Semnătura directorului de departament

Conf. dr. Sterca Adrian