

FIȘA DISCIPLINEI

1. Date despre program

1.1 Instituția de învățământ superior	Universitatea Tehnică din Cluj-Napoca
1.2 Facultatea	Automatică și Calculatoare
1.3 Departamentul	Calculatoare
1.4 Domeniul de studii	Calculatoare și Tehnologia Informației
1.5 Ciclul de studii	Licență
1.6 Programul de studii / Calificarea	Calculatoare și Tehnologia Informației/ Inginer
1.7 Forma de învățământ	IF – învățământ cu frecvență
1.8 Codul disciplinei	32.

2. Date despre disciplină

2.1 Denumirea disciplinei	Programare funcțională				
2.2 Titularii de curs	Conf.dr.ing. Radu Razvan Slavescu - Radu.Razvan.Slavescu@cs.utcluj.ro				
2.3 Titularul/Titularii activităților de seminar/laborator/proiect	Prof.dr.ing. Camelia Pinte ing. Vanessa Mercea				
2.4 Anul de studiu	3	2.5 Semestrul	5	2.6 Tipul de evaluare (E – examen, C – colocviu, V – verificare)	E
2.7 Regimul disciplinei	DF – fundamentală, DD – în domeniu, DS – de specialitate, DC – complementară				DD
	DI – Impusă, DOp – opțională, DFac – facultativă				DI

3. Timpul total estimat

3.1 Număr de ore pe săptămână	4	din care:	Curs	2	Seminar		Laborator	2	Proiect	
3.2 Număr de ore pe semestru	56	din care:	Curs	28	Seminar		Laborator	28	Proiect	
3.3 Distribuția fondului de timp (ore pe semestru) pentru:										
(a) Studiul după manual, suport de curs, bibliografie și notițe										15
(b) Documentare suplimentară în bibliotecă, pe platforme electronice de specialitate și pe teren										14
(c) Pregătire seminarii / laboratoare, teme, referate, portofolii și eseuri										10
(d) Tutoriat										2
(e) Examinări										3
(f) Alte activități:										
3.4 Total ore studiu individual (suma (3.3(a)...3.3(f)))										44
3.5 Total ore pe semestru (3.2+3.4)										100
3.6 Numărul de credite										4

4. Precondiții (acolo unde este cazul)

4.1 de curriculum	
4.2 de competențe	Notiuni fundamentale de algoritmi. Recursivitate

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Tabla, proiector, calculator
5.2. de desfășurare a laboratorului	Calculatoare, interpretoare/compileratoare specifice limbajelor studiate (LISP, ML, Haskell), Prezență obligatorie 100% pentru admiterea la examenul final

6. Competențele specifice acumulate

6.1 Competențe profesionale	<p>C2 - Proiectarea componentelor hardware, software și de comunicații</p> <p>C2.1 - Descrierea structurii și funcționării componentelor hardware, software și de comunicații</p> <p>C2.2 - Explicarea rolului, interacțiunii și funcționării componentelor sistemelor hardware, software și de comunicații</p> <p>C2.3 - Construirea unor componente hardware, software și de comunicații folosind metode de proiectare, limbaje, algoritmi, structuri de date, protocoale și tehnologii</p> <p>C2.4 - Evaluarea caracteristicilor funcționale și nefuncționale ale componentelor hardware, software și de comunicații, pe baza unor metrici</p>
-----------------------------	--

	C2.5 - Implementarea componentelor hardware, software și de comunicație
6.2 Competențe transversale	N/A

7. Obiectivele disciplinei

7.1 Obiectivul general al disciplinei	Obiectivul general este cunoașterea de către studenți a elementelor de baza ale paradigmei functionale de programare, a avantajelor acestei paradigme (imutabilitatea variabilelor, posibilitatea demonstrării formale a corectitudinii unui program, posibilitatea paralelizării facile a codului), precum și a fundamentelor sale teoretice (calculul lambda)
7.2 Obiectivele specifice	Pentru atingerea acestor obiective generale, studenții vor învăța: - să scrie cod în maniera funcțională, cu eliminarea variabilelor de stare - să identifice avantajele și dezavantajele diferitelor stiluri de programare - să utilizeze recursivitatea și să structureze codul pentru optimizarea sa - să demonstreze formal corectitudinea unui program - să manipuleze expresii lambda de baza

8. Conținuturi

8.1 Curs	Nr.ore	Metode de predare	Observații
Noțiuni fundamentale de programare în Haskell și ML: funcții, constante, identificatori.	2	(Fizic/online) Slideuri, Demonstrații și reprezentare modele pe tabla, exerciții rapide pentru creșterea interacțiunii	
Noțiuni fundamentale: tipuri primitive de date, recursivitate, liste, tupluri, operatori infix, evaluare.	2		
Liste: construire listă, operații fundamentale pe liste. Noțiuni fundamentale: declarații locale, tipuri polimorfice.	2		
Liste: operatori de liste (generatori, filtre, expresie listă).	2		
Date alternative, potrivire forme, excepții	2		
Arbori: arbori binari (conversii liste-arbori, arbori binari de căutare, verificare proprietate arbori echilibrați AVL, printare).	2		
Arbori: arbori Huffman, implementare operații pe mulțimi. Rationator propozitional.	2		
Funcții de ordin superior: funcții anonime, aplicare parțială, relația dintre funcții și date, combinatori	2		
Funcționale pentru liste. Compile, linkeditare, rulare	2		
Exemplificare funcționale pentru liste (map, filter, foldr, foldl)	2		
Date infinite: evaluare leneșă, obiecte infinite, structuri circulare.	2		
Calcul Lambda: notație lambda, conversii, combinatori.	2		
Raționare asupra corectitudinii programelor: inducție structurală, echivalența funcțiilor, inducție pe numărul de noduri.	2		
Elemente de programare paralela în Haskell. Exemple de aplicații și cazuri de utilizare	2		
Bibliografie (<i>bibliografia minimală a disciplinei conținând cel puțin o lucrare bibliografică de referință a disciplinei, care există la dispoziția studenților într-un număr de exemplare corespunzător</i>)			
1. G. Hutton. Programming in Haskell, 2nd edition Cambridge University Press, 2016			
2. M. Lipovaca. Learn You a Haskell for Great Good . No Starch Press, 2011.			
3. I.A. Letia, L.A. Negrescu, L. Negrescu. Programare funcțională, vol. I . Ed. Albastra, Cluj-Napoca, 2006			
8.2 Aplicații (seminar/laborator/proiect)	Nr.ore	Metode de predare	Observații
Obiecte Lisp, evaluarea formelor, funcții Lisp primitive.	2	(Fizic/online) Rezolvare de exerciții și probleme, implementare de funcții pe calculator, trasare de algoritmi	
Reprezentare internă, controlul evaluării, definirea funcțiilor. Recursivitate și iterație.	2		
Expresii LAMBDA, funcții de ordin superior, mapare.	2		
Liste de asociație, proprietăți	2		
Potrivirea șabloanelor. Prelucrări simbolice.	2		
Recapitulare programare în Lisp pentru colocviul de laborator.	2		
Colocviu de laborator (Programare în Lisp).	2		
Definirea funcțiilor Haskell, ML. Recursivitate.	2		
Operații pe liste.	2		

Operații pe arbori, grafuri.	2		
Funcții de ordin superior Haskell, ML	2		
Evaluare leneșă.	2		
Calcul lambda	2		
Colocviu de laborator (Programare ML, Haskell).	2		
Bibliografie (<i>bibliografia minimală pentru aplicații conținând cel puțin o lucrare bibliografică de referință a disciplinei care există la dispoziția studenților într-un număr de exemplare corespunzător</i>)			
1. I.A. Leția, E.Șt. Chifu, C. Cenan. Programare funcțională . Îndrumător de laborator, Casa cărții de știință, 1999.			
2. A. Cumming A gentle introduction to ML (tutorial online)			

*Se vor preciza, după caz: tematica seminariilor, lucrările de laborator, tematica și etapele proiectului.

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

În concordanță cu obiectivele de business ale companiilor de a dezvolta produse software robuste, scalabile, cu număr minim de erori și timp de dezvoltare cât mai redus, cursul evidențiază tehnicile specifice programării funcționale care au pătruns și pătrund în limbajele (multiparadigma) moderne, precum și pe corectitudinea dezvoltării programelor. Sunt prezentate metode formale pentru verificarea corectitudinii programelor. Conținutul disciplinei este în concordanță cu conținutul unor cursuri similare ale altor universități (limbaje de programare, Haskell paralele și concurrent).

10. Evaluare

Tip activitate	Criterii de evaluare	Metode de evaluare	Pondere din nota finală
Curs	Înțelegerea conceptelor specifice paradigmei de programare funcțională și bazei sale teoretice. Capacitatea de a demonstra formal corectitudinea programelor	Examen final: - Fizic: examen scris - Online: Test Moodle	50%
Laborator	Abilitatea de a dezvolta cod Abilitatea de a identifica și corecta erori de programare.	Teste de laborator	50%
Standard minim de performanță: Abilitatea de a dezvolta cod în limbaje funcționale. Condiții de promovare: Nota ≥ 5			

Data completării:	Titulari	Titlu Prenume NUME	Semnătura
	Curs	Conf.dr.ing. Radu Razvan Slavescu	
	Aplicații	Prof.dr.ing. Camelia Pinte Ing. Vanessa Mercea	
Data avizării în Consiliul Departamentului		Director Departament Prof.dr.ing. Rodica Potolea	
Data aprobării în Consiliul Facultății de Automatică și Calculatoare		Decan Prof.dr.ing. Liviu Miclea	