

FIȘA DISCIPLINEI

Metode avansate de programare funcțională

Anul universitar 2025-2026

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 Matematică și Informatică al Liniei Maghiare
1.4. Domeniul de studii	Informatică
1.5. Ciclu de studii	Licență
1.6. Programul de studii / Calificarea	Informatică (în limba maghiară)
1.7. Forma de învățământ	Învățământ cu frecvență

2. Date despre disciplină

2.1. Denumirea disciplinei		Metode avansate de programare funcțională / Haladó funkcionális programozás / Advanced methods of functional programming			Codul disciplinei	MLM5047	
2.2. Titularul activităților de curs		Horváth Zoltán					
2.3. Titularul activităților de seminar		Șuteu-Szöllősi Ștefan Lucian					
2.4. Anul de studiu	3	2.5. Semestrul	6	2.6. Tipul de evaluare	Examen (E)	2.7. Regimul disciplinei	Disciplină de specialitate (DS)

3. Timpul total estimat (ore pe semestru al activităților didactice)

3.1. Număr de ore pe săptămână	5	din care: 3.2. curs	2	3.3. seminar/ laborator/proiect	3
3.4. Total ore din planul de învățământ	60	din care: 3.5. curs	24	3.6 seminar/laborator/proiect	36
Distribuția fondului de timp pentru studiul individual (SI) și activități de autoinstruire (AI)					ore
Studiul după manual, suport de curs, bibliografie și notițe (AI)					36
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					20
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					14
Tutorat (consiliere profesională)					16
Examinări					4
Alte activități					
3.7. Total ore studiu individual (SI) și activități de autoinstruire (AI)				90	
3.8. Total ore pe semestru				150	
3.9. Numărul de credite				6	

4. Precondiții (acolo unde este cazul)

4.1. de curriculum	Nu e cazul
4.2. de competențe	Cunoștințe de bază de programare, cunoștințe de bază de logică matematică, cunoștințe elementare de programare funcțională

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Sală de curs cu tablă și videoproiector
5.2. de desfășurare a seminarului/ laboratorului	Sală de laborator cu tablă și videoproiector

6.1. Competențele specifice acumulate¹

Competențe profesionale/esențiale	- 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. - C1.5 Dezvoltarea de unități de program și elaborarea documentațiilor aferente - C4.1 Definirea conceptelor și principiilor de bază ale informaticii, precum și a teoriilor și modelelor matematice - C4.3 Identificarea modelelor și metodelor adecvate pentru rezolvarea unor probleme reale
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 capacităților de valorificare a cunoștințelor, de adaptare la cerințele unei societăți dinamice și de comunicare în limba română și într-o limbă de circulație internațională

6.2. Rezultatele învățării

Cunoștințe	Studentul cunoaște limbajul de programare Haskell și/sau Clean, noțiunea de curry-ing și deducția tipurilor.
Aptitudini	- Studentul utilizează paradigmele programării declarative/funcționale. - Studentul recunoaște elementele programării declarative (de exemplu construirea listelor cu „list comprehension”).
Responsabilități și autonomie	Studentul este capabil să lucreze independent într-unul dintre limbajele funcționale.

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	- Limbajele funcționale necesită un model de programare diferit de cel imperativ, iar scopul este de a învăța și de a stăpâni aceste strategii de programare. - Aplicarea modelului de programare funcțională în algoritmi moderni - Aprofundarea gândirii despre tipuri și conștientizarea utilității tipurilor
7.2 Obiectivele specifice	- Modelul de programare funcțională; algoritmi recursivi, rescrierea automată a algoritmilor recursivi, derivarea tipurilor. - Definiții ale sistemelor de tipuri în Haskell/Clean, mecanismul de derivare a tipurilor

¹ Se poate opta pentru competențe sau pentru rezultatele învățării, respectiv pentru ambele. În cazul în care se alege o singură variantă, se va șterge tabelul aferent celeilalte opțiuni, iar opțiunea păstrată va fi numerotată cu 6.

8. Conținuturi

8.1 Curs	Metode de predare	Observații
1. Introducere în paradigma funcțională, diferite limbaje de programare funcțională, comparații, caracteristici funcționale în alte limbaje de programare.	Prelegere, explicații, exemplificare	
2. Elemente ale limbajului Haskell/Clean, conceptul de tipuri, mediul de programare, recursivitate.	Prelegere, explicații, exemplificare	
3. Manipularea listelor; reprezentarea listelor, construirea lor, operații cu liste;	Prelegere, explicații, exemplificare	
4. Funcții de ordin superior; funcțiile map, filter, until, etc.	Prelegere, explicații, exemplificare	
5. Operatori, tipuri de funcții, derivarea tipurilor, funcții lambda.	Prelegere, explicații, exemplificare	
6. Modelul calculului lambda pentru evaluarea unei funcții/program. Exemple de lambda-calculus, perspectivă teoretică.	Prelegere, explicații, exemplificare	
7. Programe de înaltă performanță, reducerea complexității cu zip, zipwith	Prelegere, explicații, exemplificare	
8. Funcțiile foldr și foldl, paradigma map+fold = map-reduce. Modelul de programare pipeline și aplicațiile sale.	Prelegere, explicații, exemplificare	
9. Tipuri existențiale, tipuri de ordin superior	Prelegere, explicații, exemplificare	
10. Clase de construcție ale tipurilor	Prelegere, explicații, exemplificare	
11. Programare generativă	Prelegere, explicații, exemplificare	
12. Recapitulare, Discuții despre exercițiile de verificare.	Explicații, exemplificare	

Bibliografie

- [1]. Bird R (2011) Pearls of Functional Algorithm Design,
- [2]. Bird R. (2015) Thinking Functionally with Haskell, Cambridge University Press
- [3]. Reede, C. (1989) Elements of Functional Programming, Addison Wesley.
- [4]. Petricek T, Skeet J (2009) Real-World Functional Programming, With examples in F# and C#, Manning Publications.
- [5]. Field A. (1988) Functional Programming, Addison Wesley, New York.
- [6]. Horváth Zoltán (ELTE programnyelvek tanszék) Funkcionális programozás előadása.
- [7]. Graham Hutton (2007) Programming in Haskell, Cambridge University Press.
- [8]. Miran Lipovaca (2011) Learn you a Haskell for Great Good, No Starch Press, San Francisco.

8.2 Seminar / laborator	Metode de predare	Observații
1. Elemente ale limbajului Haskell/Clean, mediul de programare	Explicații, exemple	
2. Operații aritmetice, liste, constructori de liste	Explicații, exemple, stabilirea și rezolvarea problemelor, implementare	
3. Exersarea stilului funcțional, utilizând funcții precum map, fold, filter etc.	Explicații, exemple, stabilirea și rezolvarea problemelor, implementare	
4. Operatori, variabile de tip, polimorfism	Explicații, exemple, stabilirea și rezolvarea problemelor, implementare	
5. Tipurile record, tablou, noțiunea de „uniqueness type” (Clean)	Explicații, exemple, stabilirea și rezolvarea problemelor, implementare	
6. Tipuri de date algebrice, tratarea erorilor	Explicații, exemple, stabilirea și rezolvarea problemelor, implementare	

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

- Cursul urmează structura cursurilor de la universități de top din Anglia (University College London, University of St Andrews) și SUA (MIT, Stanford).
- Exercițiile au fost elaborate folosind materialele universităților menționate mai sus – Stanford, MIT, UCL.

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 Metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	Cunoașterea conceptelor de bază ale paradigmei programării funcționale	Examen scris (suplimentat cu întrebări orale, dacă este necesar)	50%
10.5 Seminar/laborator	Abilități de programare în Haskell/Clean	Prezentarea temelor (25%) + examen de laborator (25%)	50%
10.6 Standard minim de performanță			
• Nota de trecere este 5 • Cerințele minime trebuie îndeplinite atât la examenul scris, cât și la examenul practic (prezentarea temelor și examenul de laborator, fiecare în parte) pentru a promova.			

11. Etichete ODD (Obiective de Dezvoltare Durabilă / Sustainable Development Goals)²



Data completării:
19.09.2025

Semnătura titularului de curs
Prof. dr. Horváth Zoltán

Semnătura titularului de seminar
Conf. dr. Șuteu-Szöllősi Ștefan Lucian

Data avizării în departament:

Semnătura directorului de departament
Conf. dr. András Szilárd Károly

² Păstrați doar etichetele care, în conformitate cu [Procedura de aplicare a etichetelor ODD în procesul academic](#), se potrivesc disciplinei și ștergeți-le pe celelalte, inclusiv eticheta generală pentru Dezvoltare durabilă - dacă nu se aplică. Dacă nicio etichetă nu descrie disciplina, ștergeți-le pe toate și scrieți "Nu se aplică".