

FIȘA DISCIPLINEI

Limbaje formale și tehnici de compilare

Anul universitar 2025-2026

1. Date despre program

1.1. Instituția de învățământ superior	Universitatea Babeș-Bolyai Cluj-Napoca
1.2. Facultatea	Facultatea de Matematică și Informatică
1.3. Departamentul	Departamentul de Informatică
1.4. Domeniul de studii	Informatică
1.5. Ciclu de studii	Licență
1.6. Programul de studii / Calificarea	Informatică în limba engleză
1.7. Forma de învățământ	Cu frecvență

2. Date despre disciplină

2.1. Denumirea disciplinei	Limbaje formale și tehnici de compilare			Codul disciplinei	MLE5023
2.2. Titularul activităților de curs	Prof.dr. Simona Motogna				
2.3. Titularul activităților de seminar	Prof.dr. Simona Motogna				
2.4. Anul de studiu	3	2.5. Semestrul	5	2.6. Tipul de evaluare	E
				2.7. Regimul disciplinei	Obligatorie

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/proiect	2+
3.4. Total ore din planul de învățământ	84	din care: 3.5. curs	28	3.6 seminar/laborator/proiect	56
Distribuția fondului de timp pentru studiul individual (SI) și activități de autoinstruire (AI)					Ore
Studiul după manual, suport de curs, bibliografie și notițe (AI)					15
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					8
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					10
Tutoriat (consiliere profesională)					3
Examinări					5
Alte activități					-
3.7. Total ore studiu individual (SI) și activități de autoinstruire (AI)				41	
3.8. Total ore pe semestru				125	
3.9. Numărul de credite				5	

4. Precondiții (acolo unde este cazul)

4.1. de curriculum	- Structuri de date și algoritmi
4.2. de competențe	- Abilități medii de programare într-un limbaj de nivel înalt

5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Sala de curs cu proiector
5.2. de desfășurare a seminarului/ laboratorului	Sală de laborator dotată cu calculatoare; medii de programare (.NET, Java, Python etc.)

6.1. Competențele specifice acumulate¹

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.

Competențe profesionale/esențiale	- programarea în limbaje de nivel înalt - utilizarea bazelor teoretice ale informaticii și a modelelor formale
Competențe transversale	- 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 - 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	- Absolventul are cunoștințe legate de programare, matematică, inginerie și tehnologie și are abilitățile necesare pentru a le folosi în crearea de sisteme informatice complexe. - Absolventul are cunoștințele necesare pentru selectarea și utilizarea procedurilor potrivite de instruire pentru a facilita procesul de asimilare a cunoștințelor.
Aptitudini	- Absolventul este capabil de a prezenta și a explica metodele, algoritmi, paradigmele și tehnicile folosite în diferite ramuri ale informaticii. - Absolventul este capabil să identifice probleme complexe și să examineze probleme conexe pentru a dezvolta opțiuni de rezolvare și implementa soluții. - Absolventul este capabil să combine informații diverse pentru a formula soluții și genera idei de dezvoltare pentru noi produse și aplicații.
Responsabilități și autonomie	- Absolventul are abilitatea de a aplica reguli generale unor probleme specifice și de a produce soluții relevante. - Absolventul are capacitatea de a alege și folosi paradigme de programare (procedural, orientat obiect, funcțional) pentru realizarea de aplicații software adecvate specificului domeniului aplicației dezvoltate.

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	- Abilitatea de a înțelege structura unui compilator și a conceptelor teoretice din teoria compilării - Îmbunătățirea abilităților de programare
7.2 Obiectivele specifice	- Abilitatea de a înțelege și lucra cu conceptele din teoria limbajelor formale: ierarhie Chomsky, gramatici regulate, autoamte finite, echivalența dintre ele; gramatici independente de context, automate push down și echivalența dintre ele; mașini Turing - Abilitatea de a înțelege și lucra cu concepte din teoria compilării: analiza lexicală, analiza sintactică

8. Conținuturi

8.1 Curs	Metode de predare	Observații
----------	-------------------	------------

1. Structura generala a unui compilator. Fazele compilarii	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
2. Analiza lexicala.	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
3. Notiuni introductive de limbaje formale. Gramatici si Automate Finite	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
4. Limbaje și expresii regulate, echivalența dintre AF, gramatici regulate și expresii regulate. Lema de pompare	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
5. Gramatici independente de context (GIC), arbore sintactic	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
6. Analiza sintactica: notiuni de baza, clasificare; arbore sintactic	Expunere: descriere, explicație, exemplificare, studii de caz	
7. Analiză sintactica descendenta cu reveniri	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
8. Analiză sintactică LL(1)	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
9. Analiză sintactică LR(k). Analiză sintactică LR(0)	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
10. Analiză sintactică SLR, LR(1), LALR	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
11. Generatoare de analiza lexicala (lex) si sintactica (yacc)	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	
12. Gramatici de atribute. Generare de cod intermediar	Expunere: descriere, explicație, exemplificare, demo live	
13. Optimizare de cod intermediar. Generare de cod obiect	Expunere: descriere, explicație, exemplificare, studii de caz	
14. Automate Push Down si masini Turing	Expunere: descriere, explicație, exemplificare, dezbateri, dialog	

Bibliografie

1. A.V. AHO, D.J. ULLMAN - Principles of computer design, Addison-Wesley, 1978.
2. A.V. AHO, D.J. ULLMAN - The theory of parsing, translation and compiling, Prentice-Hall, Engl. Cliffs., N.J., 1972, 1973.
3. D. GRIES - Compiler construction for digital computers,, John Wiley, New York, 1971.
4. MOTOGNA, S. – Metode de proiectare a compilatoarelor, Ed. Albastra, 2006
5. SIPSER, M., Introduction to the theory of computation, PWS Pub. Co., 1997
6. CSÖRNYEI ZOLTÁN, Bevezetés a fordítóprogramok elméletébe, I, II., ELTE, Budapest, 1996
7. L.D. SERBANATI - Limbaje de programare si compilatoare, Ed. Academiei RSR, 1987.
8. CSÖRNYEI ZOLTÁN, Fordítási algoritmusok, Erdélyi Tankönyvtanács, Kolozsvár, 2000.
10. GRUNE, DICK - BAL, H. - JACOBS, C. - LANGENDOEN, K.: Modern Compiler Design, John Wiley, 2000

8.2 Seminar / laborator	Metode de predare	Observații
1. Specificarea unui limbaj de programare; notația BNF	Explicație, dialog, studii de caz	
2. Gramatici; limbaj generat de gramatică; gramatică corespunzătoare unui limbaj	Explicație, dialog, studii de caz, demonstrație	
3. Automat finit: limbaj acceptat de AF; AF corespunzător unui limbaj	Explicație, dialog, studii de caz, demonstrație	
4. Transformări: AF – gramatici regulate – expresii regulate	Explicație, dialog, studii de caz, exemple	
5. Transformări: AF – gramatici regulate – expresii regulate	Explicație, dialog, studii de caz, exemple	
6. Transformări: AF – gramatici regulate – expresii regulate	Explicație, dialog, studii de caz, exemple	
7. Gramatici independente de context; analiza sintactica descendenta cu reveniri	Explicație, dialog, studii de caz, exemple	
8. Analiză sintactică LL(1)	Explicație, dialog, studii de caz, exemple	
9. Analiză sintactică LR(0)	Explicație, dialog, studii de caz, exemple	

10. Analiză sintactică SLR	Explicație, dialog, studii de caz, exemple	
11. Analiză sintactică LR(1)	Explicație, dialog, studii de caz, exemple	
12. Gramatici de atribute	Explicație, dialog, studii de caz, exemple	
13. Cod intermediar	Explicație, dialog, studii de caz, exemple	
14. Automate push-down	Explicație, dialog, studii de caz, exemple	
Bibliografie 1. A.V. AHO, D.J. ULLMAN - Principles of computer design, Addison-Wesley, 1978. 2. A.V. AHO, D.J. ULLMAN - The theory of parsing, translation and compiling, Prentice-Hall, Engl. Cliffs., N.J., 1972, 1973. 3. MOTOGNA, S. – Metode de proiectare a compilatoarelor, Ed. Albastra, 2006 4. G. MOLDOVAN, V. CIOBAN, M. LUPEA - Limbaje formale si automate. Culegere de probleme, Univ. Babes-Bolyai, Cluj-Napoca, 1996		
8.3 Laborator	Metode de predare	Observații
1. Task 1: Specificarea unui mini-limbaj și implementare analiză lexicală 1.1: Specificare mini limbaj (notație BNF)	Explicație, dialog, studii de caz	
2. Task 1: Specificarea unui mini-limbaj și implementare analiză lexicală 1.2: Implementare functii principale	Explicație, dialog, studii de caz	
3. Task 1: Specificarea unui mini-limbaj și implementare analiză lexicală 1.3: Implementare tabela de simboluri	Explicație, dialog, studii de caz	
4. Task 1: Specificarea unui mini-limbaj și implementare analiză lexicală 1.4: Program principal, testare, predare	Explicație, dialog, studii de caz	
5. Task 2: Gramatici regulate + AF + transformări 2.1: Definire structuri de date pentru GR și AF; implementare transformări	Explicație, dialog, studii de caz	
6. Task 2: Gramatici regulate + AF + transformări 2.2: Program principal, testare, predare	Discuție date de testare, evaluare	
7. Task 3: GIC + transformări echivalente 3.1: Extindere task 2 pentru GIC; implementare transformări	Explicație, dialog, studii de caz	
8. Task 3: GIC + transformări echivalente 3.2: Program principal, testare, predare	Discuție date de testare, evaluare	
9. Task 4: Implementare analiză sintactică 4.1: Definire structuri de date și arhitectura	Explicație, dialog, studii de caz	Unul dintre: descendent cu reveniri, LL(1), LR(0), SLR
10. Task 4: Implementare analiză sintactică 4.2: implementare funcții analiză sintactică	Explicație, dialog, studii de caz	Task 4 în echipă de 2 studenți
11. Task 4: Implementare analiză sintactică 4.3: Program principal și integrare module	Explicație, dialog, studii de caz	
12. Task 4: Implementare analiză sintactică 4.4: testare pe gramatică	Discuție date de testare, evaluare	
13. Task 4: Implementare analiză sintactică 4.5: testare pe mini limbaj și secvență; predare	Discuție date de testare, evaluare	
14. Task 5: Instrumente lex și yacc: 5.1: implementare și predare	Explicație, dialog, studii de caz	
Bibliography 1. A.V. AHO, D.J. ULLMAN - Principles of computer design, Addison-Wesley, 1978. 2. A.V. AHO, D.J. ULLMAN - The theory of parsing, translation and compiling, Prentice-Hall, Engl. Cliffs., N.J., 1972, 1973. 3. MOTOGNA, S. – Metode de proiectare a compilatoarelor, Ed. Albastra, 2006 4. G. MOLDOVAN, V. CIOBAN, M. LUPEA - Limbaje formale si automate. Culegere de probleme, Univ. Babes-Bolyai, Cluj-Napoca, 1996.		

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

- Cursul respectă recomandările de curricula ACM și IEEE pentru studii de Informatica;
- Cursul există la majoritatea programelor de studii similare la universități de prestigiu din România și străinătate;
- Conținutul cursului este considerat important de către companii pentru acumularea de cunoștințe 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	- cunoașterea principiilor de bază din domeniu - Aplicarea conceptelor de la curs în rezolvarea de probleme	Examen scris	60%
10.5 Seminar/laborator	- abilitatea de a aplica algoritmi și a înțelege exemple – rezolvare de probleme	Probleme rezolvate; teme; observare continuă pe parcursul semestrului	10%
	-Abilitatea de implementa conceptele și algoritmii cursului - aplicarea tehnicilor pentru diferite clase de limbaje de programare	Examinare practică continuă pe parcursul semestrului – documentație, portofoliu Github	30%
10.6 Standard minim de performanță			
• Participarea la minim 75% din activitatea de seminar și minim 90% din activitatea de laborator pe timpul semestrului • Minim nota 5 (pe o scară de la 1 la 10) la examenul scris și activitatea de laborator • Înțelegerea conceptelor de bază din limbaje formale: gramatică, automat finit, automat push down, expresii regulate; înțelegerea principiilor compilării, analiză lexicală și sintactică			

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

Nu se aplică.

Data completării:

12.04.2025

Semnătura titularului de curs

Prof.dr. Simona Motogna

Semnătura titularului de seminar

Prof.dr. Simona Motogna

Data avizării în departament:

...

Semnătura directorului de departament

Conf.dr. Adrian STERCA

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".

