F I Ş A D I S C I P L I N E I [1)]

1. Date despre program

1.1. Instituţia de învăţământ superior	Universitatea Petrol-Gaze din Ploiești
1.2. Facultatea	Litere și Științe
1.3. Departamentul	Informatică, Tehnologia Informației, Matematică și Fizică
1.4. Domeniul de studii universitare	Informatică
1.5. Ciclul de studii universitare	Licență
1.6. Programul de studii universitare	Informatică

2. Date despre disciplină

2.1. Denumirea disciplinei	**Limbaje Formale și Compilatoare**
2.2. Titularul activităţilor de curs	**Lect. dr. Daniela Șchiopu**
2.3. Titularul activităţilor seminar/laborator	**Lect. dr. Daniela Șchiopu**
2.4. Titularul activităţii proiect	-
2.5. Anul de studiu	**II**
2.6. Semestrul*	**3**
2.7. Tipul de evaluare	**E**
2.8. Categoria formativă** / regimul*** disciplinei	**DF / O**

*numărul semestrului este conform planului de învăţământ;

**DF - Discipline fundamentale; DD - discipline de domeniu; DS - discipline de specialitate; DC - discipline complementare, DA - disciplina de aprofundare, DSI- disciplina de sinteza.

***obligatorie = O; opţională = A; facultativă = L

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

3.1. Număr de ore pe săptămână	4	din care: 3.2. curs	2	3.3. Seminar/laborator	2	3.4. Proiect	0
3.5. Total ore din planul de învăţământ	56	din care: 3.6. curs	28	3.7. Seminar/laborator	28	3.8. Proiect	0

3.9. Distribuţia fondului de timp	ore
Studiu după manual, suport de curs, bibliografie şi notiţe	20
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate şi pe teren	18
Pregătire seminarii/laboratoare, teme, referate, portofolii şi eseuri	23
Tutoriat	-
Examinări	8
Alte activităţi	

3.10 Total ore studiu individual	69
3.11. Total ore pe semestru	125
3.12. Numărul de credite	5

4. Precondiţii (acolo unde este cazul)

4.1. de curriculum	➢ Logică matematică și computațională ➢ Fundamentele programării

[1)] Adaptare după Ordinul Ministrului educaţiei, cercetării, tineretului şi sportului nr. 5 703/2011 privind implementarea Codului naţional al calificărilor din învăţământul superior, publicat în Monitorul Oficial al României, partea I, nr.880 bis / 13.XII.2011

	➢ Metode avansate de programare
4.2. de competenţe	➢ Cunoașterea programării procedurale (paradigmă, limbaj, fundamente şi elemente avansate); ➢ Cunoașterea unui limbaj de programare (de exemplu C, C++, Java etc.).și a unui mediu de dezvoltare de programe în acel limbaj (de exemplu CodeBlocks, Visual Studio etc.).

5. Condiţii (acolo unde este cazul)

5.1. de desfăşurare a cursului	➢ sală de curs multimedia cu videoproiector și conexiune la Internet
5.2. de desfăşurare a seminarului/laboratorului	➢ sală de laborator care să ofere software corespunzător pentru dezvoltarea de programe, precum și pentru generarea automată a componentelor unui compilator

6. Competenţe specifice acumulate

Competenţe profesionale	➢ 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. ➢ C2.3 Utilizarea metodologiilor, mecanismelor de specificare și a mediilor de dezvoltare pentru realizarea aplicațiilor informatice. ➢ C3.1 Descrierea de concepte, teorii si modele folosite in domeniul de aplicare. ➢ C3.2 Identificarea și explicarea modelelor informatice de bază adecvate domeniului de aplicare. ➢ C3.3 Utilizarea modelelor si instrumentelor informatice si matematice pentru rezolvarea problemelor specifice domeniului de aplicare. ➢ C3.4 Analiza datelor și a modelelor. ➢ C4.2 Interpretarea de modele matematice și informatice (formale). ➢ C4.3 Identificarea modelelor si metodelor adecvate pentru rezolvarea unor probleme reale. ➢ C4.5 Încorporarea de modele formale în aplicații specifice din diverse domenii.
Competenţe transversale	➢ CT1. Aplicarea regulilor de muncă organizată și eficientă, a unor atitudini responsabile față de domeniul didactic, științific și profesional, în vederea valorificării creative a propriului potențial, cu respectarea principiilor și normelor de etică profesională; ➢ CT2. Desfășurarea eficientă a activităților organizate în echipă și dezvoltarea capacităților empatice și de comunicare inter-personală, de relaționare și colaborare cu persoane și grupuri diverse implicate în dezvoltarea și utilizarea de sisteme software; ➢ CT3. Utilizarea de metode și tehnici eficiente de învățare, informare, cercetare și dezvoltare a capacităților de valorificare a cunoștințelor, dar și de adaptare la cerințele unei societăți dinamice și în continuă schimbare, precum și dezvoltarea capacității de a comunica eficient și profesionist atât în limba română, cât și într-o limbă de circulație internațională, prin însușirea și folosirea adecvată a limbajului de specialitate.

7. Obiectivele disciplinei (reieșind din grila competențelor specifice acumulate)

7.1. Obiectivul general al disciplinei	➢ dobândirea de către studenți a cunoștințelor necesare pentru a înțelege utilitatea și funcționalitatea compilatoarelor, dar și a modului de proiectare și dezvoltare a acestora (bazat pe teoria

	limbajelor formale și a automatelor), precum și aplicarea acestor cunoștințe în rezolvarea altor probleme înrudite.
7.2. Obiectivele specifice	***După parcurgerea disciplinei, studenții vor putea să:*** ➢ Identifice și să descrie corect principalele componente și funcții ale unui compilator; ➢ Definească în mod corespunzător noțiuni variate de la gramatici și limbaje generate de acestea, la expresii regulate și automate de acceptare (recunoaștere); ➢ Rezolve corect aplicații cu gramatici și limbaje generate de către acestea, mulțimi, expresii și limbaje regulate, precum și cu automate finite deterministe și nedeterministe; ➢ Rezume corect etapele realizării unui compilator și să explice concis tehnicile specifice folosite la dezvoltarea fiecărei componente a acestuia; ➢ Proiecteze și să implementeze analizoare lexicale și sintactice de complexitate elementară; ➢ Utilizeze generatoare automate pentru realizarea componentelor unul compilator; ➢ Interpreteze corect comportamentul compilatoarelor în interacțiune cu utilizatorii și cu programele lor.

8. Conţinuturi

8.1. Curs	Nr.ore	Metode de predare	Observaţii
1. Introducere în compilare. Procesoare de limbaje. Structura și fazele unui compilator. Analiza lexicală, analiza sintactică, analiza semantică, generarea codului intermediar, optimizare cod, generare cod, managementul tabelei de simboluri, medii de construcție pentru componentele unui compilator.	2	prelegeri active și angajante, supervizare și mentorat "deschise", învățarea prin descoperire, învățare pe grupuri, învățare bazată pe rezolvarea de probleme, învățare centrată pe student, learning by doing, învățare hibridă, folosirea resurselor educaționale open disponibile online, învățare reflectivă etc.	
2. Un translator simplu, orientat de sintaxă. Definirea sintaxei (gramatici, derivări, arbori de sintaxă etc.), analiza lexicală, tabela de simboluri, generarea de cod intermediar.	2		
3. Introducere în teoria compilării și elemente de teoria limbajelor formale. Noțiunea de limbaj. Gramatici și limbaje Chomsky. Mulțimi regulate, expresii regulate, limbaje regulate. Automate de acceptare – automate finite deterministe și nedeterministe. Gramatici regulate. Expresii regulate. Automate push-down. Gramatici	12		

independente de context.			
4. Proiectarea compilatoarelor. Analiza lexicală. Analiza sintactică - analiza descendentă (cu revenire, fără revenire) și analiza ascendentă (LR, SLR, canonică LR, LALR). Analiza semantică – definiții orientate de sintaxă, arbori de sintaxă, scheme de translatare orientată pe sintaxă.	7		
5. Lucrul cu generatoarele de analizoare lexicale și sintactice (FLEX și BISON).	2		
6. Generarea codului. Producerea codului intermediar. Medii de execuție. Generarea codului obiect. Optimizarea codului obiect.	1		
7. Recapitulare. Discutarea subiectelor de examen.	2		

Bibliografie

- Aho, Alfred V., **Compilers: Principles, techniques, and tools**, Pearson Education Limited, 2014 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Hopcroft, John, E., **Introduction to automata theory, languages, and computation**, Pearson, 2007 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Mogensen, Torben Aigidus, **Introduction to compiler design**, Springer, 2017 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Watson, Des, **A parctical approach to compiler construction**, Springer, 2017 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Stevanovic, Milan, **Advanced C and C++ compiling**, Apress, 2014 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Sunitha, K.V.N., Kalyani, N., **Formal Languages and Automata Theory**, Pearson, 2016.
- Lint Peter, **An introduction to formal languages and automata**, Jones and Bartlett Publishers, 2006.
- Saverio Perugini, **Programming Languages: Concepts and Implementation**, Publisher: Jones & Bartlett Learning, 2021.
- Terence Parr, **The Definitive ANTLR 4 Reference**, 2nd Edition, 2012.
- Athanasiu, I., **Limbaje formale şi automate**, Matrix Rom, Bucureşti, 2002.
- Resurse educaționale disponibile la:

  Cursul **Compilers** - Stanford: https://web.stanford.edu/class/cs143/

8.2. Laborator	Nr. ore	Metode de predare	Observații
Aplicații cu gramatici si limbaje generate de către acestea, cu mulțimi, expresii si limbaje regulate, precum si cu automate finite deterministe și nedeterministe.	14	învățarea prin descoperire, învățare pe grupuri, învățare bazată pe rezolvarea de probleme, învățare centrată pe student, learning by doing, învățare hibridă, folosirea resurselor educaționale open disponibile online, învățare reflectivă	
Organizarea și proiectarea unui compilator.	8		
Generarea automată a componentelor unui compilator.	6		

Bibliografie

- Aho, Alfred V., **Compilers: Principles, techniques, and tools**, Pearson Education Limited, 2014 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Hopcroft, John, E., **Introduction to automata theory, languages, and computation**, Pearson, 2007 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Mogensen, Torben Aigidus, **Introduction to compiler design**, Springer, 2017 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Watson, Des, **A parctical approach to compiler construction**, Springer, 2017 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Stevanovic, Milan, **Advanced C and C++ compiling**, Apress, 2014 – *disponibilă în Biblioteca Departamentului ITIMF*.
- Sunitha, K.V.N., Kalyani, N., **Formal Languages and Automata Theory**, Pearson, 2016.
- Lint Peter, **An introduction to formal languages and automata**, Jones and Bartlett Publishers, 2006.
- Saverio Perugini, **Programming Languages: Concepts and Implementation**, Publisher: Jones & Bartlett Learning, 2021.
- Terence Parr, **The Definitive ANTLR 4 Reference**, 2nd Edition, 2012.
- Athanasiu, I., **Limbaje formale şi automate**, Matrix Rom, Bucureşti, 2002.
- Resurse educaționale disponibile la:
  Cursul **Compilers** - Stanford: https://web.stanford.edu/class/cs143/

9. Coroborarea conţinuturilor disciplinei cu aşteptările reprezentanţilor comunităţii epistemice, asociaţilor profesionale şi angajatori reprezentativi din domeniul aferent programului

- Conţinuturile disciplinei corespund cu aşteptările reprezentanţilor comunităţii epistemice, asociaţiilor profesionale şi angajatorilor reprezentativi din domeniul aferent programului, aşa după cum rezultă din prezenta fişă, dar şi din fişa specializării, acestea fiind în concordanţă deplină cu CNCIS şi COR.

10. Evaluare

Tip activitate	10.1. Criterii de evaluare	10.2. Metode de evaluare	10.3. Pondere din nota finală
10.4. Curs	Calitatea răspunsurilor la examen, coerența argumentării, calitatea corelațiilor etc.	*Examen scris și oral*	40% N1= nota la această probă este de la 1 la 10 (1 punct din oficiu)
10.5. Laborator	Participarea la activitățile de laborator prin realizarea de proiecte, teme de control	*Evaluare laborator:* teme de laborator, testare scrisă, referate, dezvoltarea de programe pentru implementarea algoritmilor specifici, activitatea la laborator	60% N2= nota la această probă este de la 1 la 10 (1 punct din oficiu)
Nota finală este: 40%*N1+60%*N2			
10.6. Standard minim de performanță			
➢ RNCIS: Implementarea și documentarea de unități de program în limbaje de programare de nivel înalt și			

folosirea eficientă a mediilor de programare; Modelarea si rezolvarea unor probleme cu grad redus de complexitate, folosind cunoștințe de matematică si informatică; Realizarea și întreținerea unor aplicații informatice pentru rezolvarea unor probleme reale de complexitate redusă; Realizarea componentelor informatice pentru o aplicație dedicata de complexitate medie;

- *Lucrul cu gramatici Chomsky și a limbajelor generate de acestea, aplicații ale automatelor; utilizarea generatoarelor automate pentru componentele unui compilator.*

Data completării	Semnătura titularului de curs	Semnătura titularului de laborator
23.09.2024	Lector dr. Daniela Șchiopu	Lector dr. Daniela Șchiopu

Data avizării în departament	Director de departament	Decan
24.09.2024	Lector dr. Anca Baciu	Prof. univ. dr. Mihaela Suditu