

FIȘA DISCIPLINEI

1. Date despre program

1.1 Instituția de învățământ superior	Universitatea Babeș-Bolyai Cluj-Napoca
1.2 Facultatea	Facultatea de Matematica și Informatică
1.3 Departamentul	Departamentul de informatică
1.4 Domeniul de studii	Calculatoare și tehnologia informației
1.5 Ciclul de studii	Licență
1.6 Programul de studiu / Calificarea	Ingineria informației

2. Date despre disciplină

2.1 Denumirea disciplinei	Limbașe formale și tehnici de compilare						
2.2 Titularul activităților de curs	lect. dr. Lupsa Dana						
2.3 Titularul activităților de seminar	lect. dr. Lupsa Dana						
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	5	Din care: 3.2 curs	2	3.3 seminar/laborator	1 sem + 2 lab
3.4 Total ore din planul de învățământ	70	Din care: 3.5 curs	28	3.6 seminar/laborator	42
Distribuția fondului de timp:					ore
Studiul după manual, suport de curs, bibliografie și notițe					20
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					10
Pregătire seminarii/laboratoare, teme, referate, portofolii și eseuri					23
Tutoriat					7
Examinări					20
Alte activități:					
3.7 Total ore studiu individual					80
3.8 Total ore pe semestru					150
3.9 Numărul de credite					6

4. Precondiții (acolo unde este cazul)

4.1 de curriculum	<ul style="list-style-type: none"> structuri de date și algoritmi
4.2 de competențe	<ul style="list-style-type: none"> abilitați de programare (nivel mediu)

5. Condiții (acolo unde este cazul)

5.1 De desfășurare a cursului	<ul style="list-style-type: none">• sala dotata cu videoproiector
5.2 De desfășurare a seminarului/laboratorului	<ul style="list-style-type: none">• laborator dotat cu calculatoare; mediu de dezvoltare pt. limbaje de programare de nivel inalt

6. Competențele specifice acumulate

Competențe profesionale	<ul style="list-style-type: none">• cunoasterea, intelegerea si folosirea conceptelor informatice teoretice de baza• abilitatea de a munci independent si/sau in echipa pentru a rezolva probleme intr-un context profesional bine definit• o buna indemanare in scrierea de programe in limbaje de nivel inalt
Competențe transversale	<ul style="list-style-type: none">• abilitatea de a aplica tehnici de compilare pentru diferite probleme din viata reala• abilitatea de a modela diverse fenomene folosind limbaje formale• imbunatatirea abilitatilor de programare: depanarea programelor, o mai buna intelegere a modului de raportare a erorilor de compilare oferite de compilator (depanarea si corectarea erorilor compilatoarelor)

7. Obiectivele disciplinei (reieșind din grila competențelor acumulate)

7.1 Obiectivul general al disciplinei	<ul style="list-style-type: none">• sa fie capabil sa intelega metodele de proiectare a compilatoarelor• imbunatatirea abilitatilor de programare
7.2 Obiectivele specifice	<ul style="list-style-type: none">• cunostinte despre back-end-ul unui compilator• intelegerea conceptelor limbajelor formale si dezvoltarea abilitatilor de a lucra cu ele• intelegerea si capacitatea de a lucra cu concepte specifice compilatoarelor (analiza lexicala, analiza sintactica)

8. Conținuturi

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

1. Prezentarea generala a problemelor proiectarii si implementarii unui compilator. Structura unui compilator (prezentare intuitiva). Analiza lexicala	Expunere, descriere, explicatie, exemplificare	
2. Gramatici si limbaje. <ul style="list-style-type: none"> - Alfabet, secventa, limbaj - Gramatici si analiza sintactica. <ul style="list-style-type: none"> Analiza sintactica ascendenta-descendenta Derivari de stanga, derivari de dreapta - Gramatici echivalente. - Ierarhia Chomsky <ul style="list-style-type: none"> o Exemple de gramatici (!!); derivari; arbori de derivare o BNF si gramatica indep. de context echivalenta 	Expunere, descriere, explicatie, exemplificare	
3. Limbaje regulate <ul style="list-style-type: none"> • Gramatici regulate. Automate finite. Expresii regulate. (definitii, exemple) • teoreme de echivalenta (doar teorema; fara constructii; partea de constructii in cursul 12) • AF: determinism si nedeterminism 	Expunere, descriere, explicatie, exemplificare	
4. Gramatici independente de context (GIC). Introducere <ul style="list-style-type: none"> • definitii, exemple 	Expunere, descriere, explicatie, exemplificare	
5. Gramatici de atribute. Introducere. <ul style="list-style-type: none"> • definitii, exemple 	Expunere, descriere, explicatie, exemplificare	
6,7. Generatoare de analizoare <ul style="list-style-type: none"> • Generator de analizor lexical lex/flex. • Generator de analizor sintactic yacc/bison • Exemple de atributare a gramaticilor <ul style="list-style-type: none"> o generare de cod intermediar (functional ; limbaj de asamblare ca si cod intermediar) 	Expunere, descriere, explicatie, exemplificare	
8. Gramatici speciale - gramatici LR(k) Analiza LR(0)	Expunere, explicatie, exemplificare, studiu de caz	
9. Gramatici speciale - gramatici LR(k) Analiza SLR, LR(1), LALR	Expunere, explicatie, exemplificare, studiu de caz	
10. Gramatici speciale - gramatici LL(k) Analiza LL(1) Recursivitate la stanga, eliminarea rec. la stg.	Expunere, explicatie, exemplificare, studiu de caz	
11, 12. Aspecte formale: demonstratii, constructii si aplicatii De ce si l.regulare si l.i.c? Proprietati ale limbajelor regulate, limbajelor independente de context (leme de pompare). Exemple: structuri sintactice in limb. de programare ce nu pot fi descrise de limb. regulate De ce AFD pentru procesarea limb. regulate? Echivalenta dintre automatele finite si gramaticile regulate. Echivalenta dintre AF si expresiile regulate De ce l.i.c. si LL(1) si LR(*)? Automate push-down (APD). APD echivalent cu GIC. Ne-echivalenta dintre APDNedet. si APDDet (exemplu) Unicitatea alegerii in cazul LL(1) si LR(*)	Expunere, descriere, explicatie, exemplificare, demonstrare	
13, 14. Aplicativitatea mecanismelor formale studiate in proiectarea compilatoarelor Analiza semantica, forme intermediare, generarea codului intermediar, optimizarea codului intermediar. Introducere. Studiul consecintelor practice (in proiectarea compilatoarelor) ale aspectelor teoretice studiate.	Expunere: descrierea, exemplificarea	
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.		

<p>4. MOTOGNA, S. – Metode de proiectare a compilatoarelor, Ed. Albastra, 2006</p> <p>5. SIPSER, M., Introduction to the theory of computation, PWS Publ. Co., 1997.</p> <p>6. L.D. SERBANATI - Limbaje de programare si compilatoare, Ed. Academiei RSR, 1987.</p>		
8.2 Seminar	Metode de predare	Observatii
1. Specificarea unui limbaj de programare. Notatia BNF	Explicatia, dialogul, studiu de caz	Orele de seminar sint grupate cate 2, la fiecare 2 saptamani
2. Automate finite: <ul style="list-style-type: none"> limbaj generat de un automat finit, automat finit corespunzator unui limbaj structuri de date pt. automate finite 	Dialogul, dezbaterea, studiul de caz, exemplificarea	
3. Gramatici, limbaje generate de o gramatica, gramatica corespunzatoare unui limbaj	Dialogul, studiul de caz, exemplificarea, demonstratia	
4. Analiza sintactica LR(k)	Dialogul, dezbaterea, studiul de caz, exemplificarea,	
5. Gramatici independente de context. Analiza sintactica LL(1)	Dialogul, dezbaterea, studiul de caz, exemplificarea	
6. Proprietati ale limbajelor. Demonstratii si aplicatii.	Dialogul, studiul de caz, demonstratia	
7. APD. Exercitii recapitulative: gramatici de tip 1, 2 si 3 (ierahia Chomsky)	Dialogul, dezbaterea, studiul de caz, exemplificarea	
<p>Bibliografie</p> <p>1. A.V. AHO, D.J. ULLMAN - Principles of computer design, Addison-Wesley, 1978.</p> <p>2. A.V. AHO, D.J. ULLMAN - The theory of parsing, translation and compiling, Prentice-Hall, Engl. Cliffs., N.J., 1972, 1973.</p> <p>3. MOTOGNA, S. – Metode de proiectare a compilatoarelor, Ed. Albastra, 2006</p> <p>4. G. MOLDOVAN, V. CIOBAN, M. LUPEA - Limbaje formale si automate. Culegere de probleme, Univ. Babes-Bolyai, Cluj-Napoca, 1996</p>		
8.3 Laborator	Metode de predare	Observatii
1. Tema 1: Specificarea uni mini-limbaj de programare si implementarea unui analizor lexical 1.1 specificarea mini-limbajului de programare	Explicatia, dialogul, studiu de caz	
2. Tema 1: Specificarea uni mini-limbaj de programare si implementarea unui analizor lexical 1.2 implemetarea functiilor principale ale analizorului	Explicatia, dialogul, studiu de caz	
3. Tema 1: Specificarea uni mini-limbaj de programare si implementarea unui analizor lexical 1.3 organizarea tabelii de simboluri	Explicatia, dialogul, studiu de caz	
4. Tema 1: Specificarea uni mini-limbaj de programare si implementarea unui analizor lexical 1.4 programul principal, testare + predare	Discutarea datelor de test, evaluare	
5. Tema 2: Automate finite 2.1 verificarea acceptarii unei secvente <ul style="list-style-type: none"> numai pentru AF deterministe verificati ca AF este determinist 	Explicatia, dialogul, studiu de caz	
6. Tema 2: Automate finite: 2.2 Rescrieti/adaptati programul de analiza lexicala (tema 1) astfel incat sa foloseasca automate finite pentru a determina secventele corespunzatoare atomilor lexicali	Discutarea datelor de test, evaluare	
7. Tema 3: Implementarea unui analizor lexical folosind un generator de analizor lexical <ul style="list-style-type: none"> utilizare lex/flex 	Explicatia, dialogul, studiu de caz, discutarea datelor de test, evaluare	
8. Tema 4: Implementarea unui analizor sintactic folosind un generator de analizor sintactic <ul style="list-style-type: none"> utilizare yacc/bison 	Explicatia, dialogul, studiu de caz, discutarea datelor de test, evaluare	
9. Tema 5: Implementarea unui mini-compilator functional 5.1 specificare si implementare	Explicatia, dialogul, studiu de caz, discutarea datelor de	

<ul style="list-style-type: none"> • utilizare lex/flex + yacc/bison • re folosire rezolvarea problemelor de la temele 3 si 4 • generarea unui program asm echivalent semantic si testarea acestuia 	test	
10. Tema 5: Implementarea unui mini-compiler functional 5.2 compilarea si testarea programului asm generat testare si corectarea posibilelor erori	Discutarea datelor de test, evaluare	
11. Tema 6: Implementarea unui analizor sintactic 6.1 alegerea structurilor de date si arhitectura aplicatiei 6.2 implementarea functiilor principale ale analizorului	Explicatia, dialogul, studiu de caz	
12. Tema 6: Implementarea unui analizor sintactic 6.3 programul principal si integrarea modulelor	Explicatia, dialogul, studiu de caz	
13. Tema 6: Implementarea unui analizor sintactic 6.4 testare si corectarea posibilelor erori	Explicatia, dialogul, studiu de caz	
14. Tema 6: Implementarea unui analizor sintactic 6.5 predarea	Discutarea datelor de test, evaluare	
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. Alabastra, 2006 5. L.D. SERBANATI - Limbaje de programare si compilatoare, Ed. Academiei RSR, 1987.		

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

<ul style="list-style-type: none"> • Tematica cursului respecta recomandările de conținut IEEE si ACM pentru studiile din domeniul informatica • Cursul exista in programul de studiu al universitatilor importante din Romania si din alte tari • Tematica cursului este considerata de companiile soft ca fiind importanta cel putin pentru programatorii de nivel mediu

10. Evaluare

Tip activitate	10.1 Criterii de evaluare	10.2 metode de evaluare	10.3 Pondere din nota finală
10.4 Curs	- cunoasterea principiilor de baza ale domeniului - aplicarea conceptelor studiate la curs - rezolvarea de probleme	Examen scris	70%
10.5 Seminar			
10.6 Laborator	- sa fie capabili sa implementeze conceptele si algoritmi cursului - implementarea unui translator (vazut ca o parte a unui mini-compiler)	Verificare continua a activitatii in timpul orelor de laborator - verificarea documentatiei - verificarea programelor	30%
10.7 Standard minim de performanță			

- Cel puțin nota 5 la examenul scris, și cel puțin nota 5 pentru media calculată

Data completării

.....

Semnătura titularului de curs

Lect. Dr. LUPSA Dana

Semnătura titularului de seminar

Lect. Dr. LUPSA Dana

Data avizării în departament

.....

Semnătura directorului de departament

.....