Universitatea „Lucian Blaga” din Sibiu
Facultatea de Ştiinţe
Catedra de Informatică
Domeniul de studii de licenţă: Informatică
Specializarea: Informatică

PROGRAMA ANALITICĂ

Denumirea disciplinei: Proiectarea compilatoarelor
Codul disciplinei: 3906F05O044
Anul de studiu şi semestrul în care se studiază disciplina: III/5
Regimul disciplinei (obligatorie O, opţională A sau facultativă L): A
Categoria formativă (fundamentală Fd, de specialitate Sp, generală Gen): Fd
Discipline anterioare cerute *: Limbaje formale şi teoria automatelor, OOP
Forma de evaluare (examen E, verificare V, colocviu C): E
Catedra care coordonează disciplina: Catedra de Informatică
Titularul / titularii disciplinei: lector drd. Ralf Fabian

* disciplinele studiate anterior a căror cunoaştere este necesară pentru însuşirea disciplinei

Extinderea disciplinei în planul de învăţământ *:				
Curs	Seminar	Laborator	Proiect	Total ($NOAD_{sem}$)
28		28		56

* numărul semestrial de ore de activităţi didactice directe

Bugetul de timp şi creditele alocate disciplinei			
$NOAD_{sem}$	$NOSI_{sem}$	$NOT_{sem} = NOAD_{sem} + NOSI_{sem}$	**Numărul de credite**
56	112	168	6

Obiectivele disciplinei
Obiectivele cursului e) Iniţierea studenţilor în teoria compilatoarelor f) Noţiuni de construcţia compilatoarelor: fazele compilării, analiza lexicala, analiza sintactica, analiza semantica. g) Prezentarea sistematica a noţiunilor şi metodelor de baza utilizate la proiectarea şi implementarea limbajelor de programare. h) Familiarizarea studenţilor cu gândirea algoritmică;
Obiectivele activităţilor aplicative **(seminar, laborator, proiect)** e) forme de specificare a limbajelor (regulare, independente de context); f) înţelegerea modurilor de specificărilor unui limbaj nou; g) structura si funcţionarea unui translator;

h) deprinderilor de realizare în grup a unui produs program corect (cu limbajele C++, C#, Java), prin parcurgerea tuturor etapelor necesare şi reflectarea lor într-o documentație completă.
i) Aprofundarea cunoştinţelor de programare ale studenților.

Conținutul disciplinei (capitolele cursului / tematica seminarului / lucrărilor practice / etapele proiectului)			
CURS			
Nr. crt.	**Tema**	**Nr.ore**	**Săptămâna**
1.	Noțiuni introductive şi notații. Tipuri de limbaje de programare. Procesoare de limbaje. Structura generala a unui compilator.	2	1
2.	Expresii regulate. Operații cu expresii regulate. Automatul finit echivalent cu o expresie regulată. Tehnici de construire a automatelor finite din expresii regulate. Automat finit nedeterminist echivalent cu un automat finit determinist. Minimizarea automatelor finite.	4	2-3
	Analiza lexicală	2	4
3.	Analiza sintactică. Algoritmi generali de analiză sintactică top-down şi bottom-up. Algoritmul de analiză Cocke-Younger-Kasami	4	5-6
4.	Algoritmi de analiză cu complexitate liniară. Algoritmul de analiza sintactică LL(k)	2	7
5.	Algoritmul de analiza sintactică LR(k)	2	8
6.	Gramatici de precedență	2	9
7.	Analiza semantică. Gramatici de atribute. Verificarea tipurilor	2	10
8.	Generarea şi optimizarea de cod	4	11, 12
9.	Gestiunea tabelei de simboluri. Tipuri de tabele Detectarea erorilor. Sursele şi corectarea erorilor. Observații şi concluzii	4	13, 14
LABORATOR			
Nr. crt.	**Tema**	**Nr.ore**	**Săptămâna**
1.	Recapitulare noțiunilor de baza pentru mecanismele care vor fi studiate. Prezentarea principalelor componente ale unui compilator. Familiarizarea cu mediul de dezvoltare.	2	1
2.	Modelul analizorului lexical. Generare analizoarelor lexicale. Construcția unui semiautomat finit pornind de la o gramatica de tipul 3. Transformarea expresiilor regulate în automate finite. Stabilirea echipelor pentru temele de implementare. Împărțirea proiectelor.	2	2
3.	Transformarea unui automat finite nedeterminist în automat finit determinist. Minimizarea automatelor finite.	2	3

4.	Analiza sintactica. Algoritmul general de analiza top-down (cu şi fără revenire).	2	4
5.	Algoritmi generat de analiza bottom-up. Algoritmul de analiza Cocke-Younger-Kasami	2	5
6.	Algoritmi de analiza sintactica de complexitate liniara. Gramatici si limbaje LL(k). Implementarea operatorului $\oplus_k$.	2	6
7.	Gramatici şi limbaje LR(k). Construcția tabelelor de analiza LR(k)	2	7
8.	Gramatici şi limbaje de precedenta. Construcția matricei de precedență	2	8
9.	Tabele de simboluri. Analiza tipurilor	2	9
10.	Construcția unui translator simplu. Interpretor de cod virtual.	2	10
11.	Construcția unui generator simplu de cod maşină.	4	11, 12
12.	Predarea şi prezentarea temelor de implementare.	4	13, 14

Descrierea metodelor de predare
- Pentru predare se va folosi prelegerea, dezbaterea, învățarea prin cooperare, explicația, tematizarea precum şi prin expunere, construcție de exemple, discuții şi activități interactive, munca independenta; - asigurarea calității recepționării cunoştințelor se face prin urmărirea pe parcursul semestrului a rezolvărilor de probleme/exerciții cerute; - lucrări de laborator disponibile studenților, la cerere pe suport magnetic. - Sunt valabile regulamentele oficiale ale universității în legătură cu prezența studenților la activitățile didactice şi cu cazurile de copiat şi plagiat. - Promovarea examenului este condiționată de predarea completă a lucrărilor de proiect. - Prezenta la orele de laborator este obligatorie.

Descrierea formelor şi metodelor de evaluare a cunoştințelor
Nota finală se va stabili după cum urmează: xxx) Proiect de semestru 60 % xxxi) Examen final 40 % Evaluarea proiectului de semestru constă în: - predarea şi susținerea în ultima săptămâna din semestru a programului şi documentația realizata, pe care se va acorda o notă. Nu se accepta întârzieri; - se va pune accent deosebit pe scrierea cât mai clară a documentației complete şi la timp.

Bibliografie obligatorie
8. Vasile Crăciunean, *Proiectarea Translatoarelor*, Sibiu, Editura Alma Mater, 2002. 9. Ralf Fabian, *Limbaje formale Teorie. Exemple. Probleme*, Editura Universității „Lucian Blaga”, Sibiu, 2006, pag. 123, ISBN 973-739-225-6 10. Hopcroft, J.E., Motwani, R. and Ullman, J.D. - *Introduction to Automata Theory, Languages, and Computation*, Addison Wesley 2007.

Bibliografie opțională
1. Emil M. Popa, *Limbaje formale, Fundamentele limbajelor de programare*, Editura „Alma Mater”, Sibiu, 2003. 2. Emil M. Popa, *Formal Syntax and Semantics of Programming Language*, Editura „Alma Mater”, Sibiu, 2004.

3. Aho,A.V., Ullman, J.D., *The Theory of Parsing Translations and Compiling*, vol.1: Parsing, vol2: Compiling, New Jersey, Prentice-Hall, inc., 1972.
4. Teodor Rus, *Mecanisme formale pentru specificarea limbajelor*, Ed. Academie Române, Bucureşti, 1983.
5. Luca Dan Şerbănaţi, *Limbaje de programare şi compilatoare*, Ed. Academiei Române, Bucureşti, 1987.
6. Ernest G. Manes, Michael A. Arbib. *Algebraic Approaches to program semantics* – Springer – Verlag New York Berlin Heidelberg London Paris Tokyo – 1986
7. Salomaa A., *Formal Languages*, New York, Academic Press, 1973.
8. Creanga I., Reischer C., Simovici D. *Introducerea algebrică in informatică*, vol. 1 şi vol. 2, Editura Tehnică 1974
9. Paun Gh., *Probleme actuale in teoria limbajelor formale*, Editura Academiei, 1983.
10. Păun Gh., *Gramatici contextuale*, Bucureşti, 1982
11. Păun Gh., *Mecanisme generative ale proceselor economice*, Ed Tehnică, Bucureşti, 1988

Data elaborării:

Titularul / titularii disciplinei
lector drd. Ralf Fabian