LEHRVERANSTALTUNGSBESCHREIBUNG

1. Angaben zum Programm

1.1 Hochschuleinrichtung	**Babes-Bolyai Universität, Cluj-Napoca**
1.2 Fakultät	Mathematik und Informatik
1.3 Department	Informatik
1.4 Fachgebiet	Informatik
1.5 Studienform	Bachelor
1.6 Studiengang / Qualifikation	Informatik

2. Angaben zum Studienfach

2.1 LV-Bezeichnung		Formale Sprachen und Kompiliertechniken					
2.2 Lehrverantwortlicher – Vorlesung							
2.3 Lehrverantwortlicher – Seminar							
2.4 Studienjahr	3	2.5 Semester	5	2.6. Prüfungsform	P	2.7 Art der LV	Pflichtfach
Modulnummer		MLG5023					

3. Geschätzter Workload in Stunden

3.1 SWS	6	von denen: 3.2 Vorlesung	2	3.3 Labor	2+2
3.4 Gesamte Stundenanzahl im Lehrplan	84	von denen: 3.5 Vorlesung	28	3.6 Seminar/Übung	56
Verteilung der Studienzeit:					Std.
Studium nach Handbücher, Kursbuch, Bibliographie und Mitschriften					30
Zusätzliche Vorbereitung in der Bibliothek, auf elektronischen Fachplattformen und durch Feldforschung					30
Vorbereitung von Seminaren/Übungen, Präsentationen, Referate, Portfolios und Essays					30
Tutorien					6
Prüfungen					20
Andere Tätigkeiten:					-

3.7 Gesamtstundenanzahl Selbststudium	116
3.8 Gesamtstundenanzahl / Semester	200
3.9 Leistungspunkte	8

4. Voraussetzungen (falls zutreffend)

4.1 curricular	• Datenstrukturen und Algorithmen
4.2 kompetenzbezogen	• Programmingskills

5. Bedingungen (falls zutreffend)

5.1 zur Durchführung der Vorlesung	• Vorlesungsraum, Beamer, Laptop
5.2 zur Durchführung des Seminars / der Übung	• Computerraum

6. Spezifische erworbene Kompetenzen

Berufliche Kompetenzen	• Wissen, Verstehen und Anwenden der Grundbegriffe der Informatik • Fähigkeit sowohl selbständig als auch im Team zu arbeiten. • Programmierfähigkeiten in höheren Programmiersprachen.
Transversale Kompetenzen	• Fähigkeit Kompiliertechniken für das Lösen verschiedener Probleme anzuwenden. • Fähigkeit veschiedene Phänomene mit Hilfe formaler Sprachen zu modellieren.

7. Ziele (entsprechend der erworbenen Kompetenzen)

7.1 Allgemeine Ziele der Lehrveranstaltung	• Das Erlernen und Verstehen wie man Compiler aufbaut. • Verbesserung der Programmierfähigkeiten.
7.2 Spezifische Ziele der Lehrveranstaltung	• Kenntnisse über ein compiler back-end. • Aneignen der grundlegenden Begriffe der formalen Sprachen. • Aneignen der grundlegenden Begriffe über Compiler.

8. Inhalt

8.1 Vorlesung	Lehr- und Lernmethode	Anmerkungen
1. Einführung in die Compiler Implementierung. Grammatiken und Sprachen.	Vortrag, Erklärung, Debatte, praktische Beispiele	
2. Lektische Analysis.	Vortrag, Erklärung, Debatte, praktische Beispiele	
3. Reguläre Grammatiken, endliche Automata.	Vortrag, Erklärung, Debatte, praktische Beispiele	
4. Kontext unabhängige Grammatiken.	Vortrag, Erklärung, Debatte, praktische Beispiele	
5. Push-down Automata.	Vortrag, Erklärung, Debatte, praktische Beispiele	
6. Spezielle Grammatiken – LL(k) Grammatiken.	Vortrag, Erklärung, Debatte, praktische Beispiele	

7. Spezielle Grammatiken – LR(k) Grammatiken.	Vortrag, Erklärung, Debatte, praktische Beispiele	
8. Spezielle Grammatiken – LR(k) Grammatiken. SLR, LR(1) und LALR Analysis.	Vortrag, Erklärung, Debatte, praktische Beispiele	
9. Compiler Struktur.	Vortrag, Erklärung, Debatte, praktische Beispiele	
10. Lektischer Analysis Generator lex/flex.	Vortrag, Erklärung, Debatte, praktische Beispiele	
11. Beweise, Konstruktionen und Anwendungen. Äquivalenz der endlichen Automata mit den regulären Grammatiken.	Vortrag, Erklärung, Debatte, praktische Beispiele	
12. Beweise, Konstruktionen und Anwendungen. Eigenschaften regulärer Sprachen.	Vortrag, Erklärung, Debatte, praktische Beispiele	
13. Semantische Analysis.	Vortrag, Erklärung, Debatte, praktische Beispiele	
14. Anwendungen formaler Mechanismen in der semantischen Analysis.	Vortrag, Erklärung, Debatte, praktische Beispiele	

Literatur

1. C. WAGENKNECHT, HIELSCHER M., Formale Sprachen, abstrakte Automaten und Compiler, Vieweg Teubner, 2009.
2. ASTEROTH, A., BAIER, C., Theoretische Informatik, eine Einführung in Berechnbarkeit, Komplexität und formale Sprachen, Pearson Studium, 2002.
3. HROMKOVIC, J., Theoretische Informatik, Formale Sprachen, Berechenbarkeit, Komplexitätstheorie, Algorithmik, Kommunikation und Kryptographie, Vieweg Teubner, 2011.

Weitere Literatur:

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 Pulb. Co., 1997.

6. L.D. SERBANATI - Limbaje de programare si compilatoare, Ed. Academiei RSR, 1987.

8.2 **Übung**	Lehr- und Lernmethode	Anmerkungen
1-2. Spezifizierung einer Programmiersprache.	Beispiele, Diskussionen, Teamarbeit	

Die BNF Notation.		
3-4. Grammatiken, die von Grammatiken erzeute Sprache, die Grammatik einer Sprache.	Beispiele, Diskussionen	
5-6. Endliche Automata	Beispiele, Diskussionen	
7-8. Kontext unabhängige Grammatiken. Syntaktische Analysis LL(1).	Beispiele, Diskussionen, Teamarbeit	
9-10. Syntaktische Analysis LR(k).	Beispiele, Diskussionen, Teamarbeit	
11-12. Spracheigenschaften. Beweise und Anwendungen.	Beispiele, Diskussionen, Teamarbeit	
13-14. APD. Grammatiken vom Typ 1, 2 und 3 (Chomsky Hierarchie).	Diskussionen	

Literatur

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

Labor		
1. Spezifizieren einer mini Programmiersprache und Implementieren eines lexikalen Analysators. 1.1. Spezifikation der mini Programmiersprache	Beispiele, Diskussionen, Teamarbeit	
2. Spezifizieren einer mini Programmiersprache und Implementieren eines lexikalen Analysators. 1.2. Implementation der Hauptfunktionen des Analysators.	Beispiele, Diskussionen, Teamarbeit	
3. Spezifizieren einer mini Programmiersprache und Implementieren eines lexikalen Analysators.	Beispiele, Diskussionen, Teamarbeit	

1.3. Organisation der Symbolentafel.		
4. Spezifizieren einer mini Programmiersprache und Implementieren eines lexikalen Analysators. 1.4. Hauptprogramm. Testen und Abgabe.	Beispiele, Diskussionen, Teamarbeit	
5. Endliche Automata: Verifikation der Akzeptierung einer Datensequenz. Das Wählen der Datenstrukturen und die Architektur der Anwendung.	Beispiele, Diskussionen, Teamarbeit	
6. Endliche Automata: Implementierung, Testen und Abgabe.	Beispiele, Diskussionen, Teamarbeit	
7. Endliche Automata: Anpassen des Programms für die lektische Analysis um die Benutzung endlicher Automata zu erlauben. Bestimmen der Sequenzen lektischer Atome.	Beispiele, Diskussionen, Teamarbeit	
8. Implementierung eines syntaktischen Analysators: Datenstruktur Auswahl und Programmarchitektur.	Beispiele, Diskussionen, Teamarbeit	
9. Implementierung eines syntaktischen Analysators: Hauptfunktionen.	Beispiele, Diskussionen, Teamarbeit	
10. Implementierung eines syntaktischen Analysators: Hauptprogramm und Modulintegration.	Beispiele, Diskussionen, Teamarbeit	
11. Implementierung eines syntaktischen Analysators: Testen und Fehlerbehebung.	Beispiele, Diskussionen, Teamarbeit	
12. Implementierung eines syntaktischen Analysators: Abgabe.	Beispiele, Diskussionen, Teamarbeit	
13. Anwendung von lex/flex + yac/bison: Implementierung.	Beispiele, Diskussionen, Teamarbeit	
14. Anwendung von lex/flex + yac/bison: Testen und Abgabe.	Beispiele, Diskussionen, Teamarbeit	

Literatur:

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. L.D. SERBANATI - Limbaje de programare si compilatoare, Ed. Academiei RSR, 1987.

9. Verbindung der Inhalte mit den Erwartungen der Wissensgemeinschaft, der Berufsverbände und der für den Fachbereich repräsentativen Arbeitgeber

Diese Vorlesung wird an international bekannten Universitäten im Fachgebiet Informatik.

Der Inhalt des Kurses gilt als wichtiger Teil der Programmierkenntnisse der Informatiker in Software-Unternehmen.

10. Prüfungsform

Veranstaltungsart	10.1 Evaluationskriterien	10.2 Evaluationsmethoden	10.3 Anteil an der Gesamtnote
10.4 Vorlesung	Grundkenntnisse.	Schriftliche Arbeit	75%
10.5 Seminar / Übung	Algorithmenanwendung	Labor Arbeiten	25%
10.6 Minimale Leistungsstandards			
Für das Bestehen der Prüfung muss die Mindestnote 5 erzielt werden (bei der schriftlichen Arbeit, bzw. beim Lösen der Laboraufgaben).			

Ausgefüllt am: Vorlesungsverantwortlicher Seminarverantwortlicher

Genehmigt im Department am: Departmentdirektor