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 (de) (en) (ro)	Formale Sprachen und Compiler Formal languages and compilers Limbaje formale şi tehnici de compilare						
2.2 Lehrverantwortlicher – Vorlesung				Lect. Dr. Iulian Simion			
2.3 Lehrverantwortlicher – Seminar				Lect. Dr. Iulian Simion			
2.4 Studienjahr	3	2.5 Semester	5	2.6. Prüfungsform	P	2.7 Art der LV	Pflichtfach

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	K 4.1 Definieren der Grundkonzepte und Prinzipien der Informatik, sowie der mathematischen Theorien und Modelle K 4.2 Interpretation der formalen Modelle der Mathematik und Informatik K 4.3 Identifizierung der geeigneten Modelle und Methoden für die Lösung realer Probleme K 4.4 Anwendung der Simulierungen für die Untersuchung der Verhaltensweise der angewandten Modelle und Bewertung der Ergebnisse K4.5 Einbauen der formalen Modelle in geeignete Anwendungen der spezifischen Gebiete
Transversale Kompetenzen	TK1 Anwendung der Regeln für gut organisierte und effiziente Arbeit, für verantwortungsvolle Einstellungen gegenüber der Didaktik und der Wissenschaft, für kreative Förderung des eigenen Potentials, mit Rücksicht auf die Prinzipien und Normen der professionellen Ethik TK3 Anwendung von effizienten Methoden und Techniken für Lernen, Informieren und Recherchieren, für das Entwicklen der Kapazitäten der praktischen Umsetzung der Kenntnisse, der Anpassung an die Bedürfnisse einer dynamischen Gesellschaft, der Kommunikation in rumänischer Sprache und in einer internationalen Verkehrssprache

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 den Aufbau eines Compiler. ¬ Aneignen der grundlegenden Begriffe der formalen Sprachen. ¬ Aneignen der grundlegenden Begriffe über Compiler.

8. Inhalt

8.1 Vorlesung	Lehr- und Lernmethode	Anmerkungen
1. Überblick: • Aufbau eines Compiler • Frontend, Optimizer, Backend • Reguläre Ausdrücke in der Praxis	Vortrag, Erklärung, Debatte, praktische Beispiele	
2. Scanner: • Endliche Automaten • reguläre Ausdrücke und reguläre	Vortrag, Erklärung, Debatte, praktische Beispiele	

Sprachen		
3. Scanner: • Thompson-Konstruktion • Teilmengenkonstruktion	Vortrag, Erklärung, Debatte, praktische Beispiele	
4. Scanner: • Hopcroft-Algorithmus • Scannerumsezungen	Vortrag, Erklärung, Debatte, praktische Beispiele	
5. Scanner: • Scannerumsezungen • Kleene-Konstruktion	Vortrag, Erklärung, Debatte, praktische Beispiele	
6. Parser: • kontextfreie Grammatiken • Parse-Bäume	Vortrag, Erklärung, Debatte, praktische Beispiele	
7. Top-Down-Parser: • generischer Algorithmus • backtrackingfreie Grammatiken	Vortrag, Erklärung, Debatte, praktische Beispiele	
8. Top-Down-Parser: • Beseitigung von Linksrekursion • backtrackingfreie Grammatiken	Vortrag, Erklärung, Debatte, praktische Beispiele	
9. Top-Down-Parser: • Rekursiver Abstieg • LL(1)-Parser	Vortrag, Erklärung, Debatte, praktische Beispiele	
10. Bottom-Up-Parser: • LR(1)-Parser • Aufbau der LR(1)-Tabellen	Vortrag, Erklärung, Debatte, praktische Beispiele	
11. Bottom-Up-Parser: • Aufbau der LR(1)-Tabellen • Wiederanlauf im Fehlerfall	Vortrag, Erklärung, Debatte, praktische Beispiele	
12. Attributgrammatiken	Vortrag, Erklärung, Debatte, praktische Beispiele	
13. Backend:	Vortrag, Erklärung, Debatte, praktische Beispiele	Je nach Bedarf, werden die Themen in den letzten

• Instruction Scheduling (Befehlseinplanung)		zwei Vorlesungen erweitert oder ausgelassen
14. Chomsky-Hierarchie	Vortrag, Erklärung, Debatte, praktische Beispiele	Je nach Bedarf, werden die Themen in den letzten zwei Vorlesungen erweitert oder ausgelassen
Literatur [1] K.D. COOPER, L. TORCZON - Engineering a Compiler, Elsevier Science & Technology, 2011. [2] A.V. AHO, D.J. ULLMAN - Principles of compiler design, Addison-Wesley, 1978. [3] C. WAGENKNECHT, HIELSCHER M., Formale Sprachen, abstrakte Automaten und Compiler, Vieweg Teubner, 2009. [4] ASTEROTH, A., BAIER, C., Theoretische Informatik, eine Einführung in Berechnbarkeit, Komplexität und formale Sprachen, Pearson Studium, 2002. [5] HROMKOVIC, J., Theoretische Informatik, Formale Sprachen, Berechenbarkeit, Komplexitätstheorie, Algorithmik, Kommunikation und Kryptographie, Vieweg Teubner, 2011.		
8.2 **Übung**	Lehr- und Lernmethode	Anmerkungen
1. RA in der Praxis	Beispiele, Diskussionen, Teamarbeit	
2-5. Übungen zu: • NEA und DEA • reguläre Ausdrücke • reguläre Sprachen • entsprechende Algorithmen	Beispiele, Diskussionen	
6-12. Übungen zu: • kontextfreie Grammatiken • reguläre Grammatiken • LL(k)-Grammatiken • LR(k)-Grammatiken	Beispiele, Diskussionen	
13-14. Instruction Scheduling und/oder Chomsky-Hierarchie	Beispiele, Diskussionen	Je nach Bedarf, werden diese Themen erweitert oder ausgelassen
Literatur [1] K.D. COOPER, L. TORCZON - Engineering a Compiler, Elsevier Science & Technology, 2011. [2] A.V. AHO, D.J. ULLMAN - Principles of compiler design, Addison-Wesley, 1978. [3] C. WAGENKNECHT, HIELSCHER M., Formale Sprachen, abstrakte Automaten und Compiler, Vieweg Teubner, 2009. [4] ASTEROTH, A., BAIER, C., Theoretische Informatik, eine Einführung in Berechnbarkeit, Komplexität und formale Sprachen, Pearson Studium, 2002. [5] HROMKOVIC, J., Theoretische Informatik, Formale Sprachen, Berechenbarkeit, Komplexitätstheorie, Algorithmik, Kommunikation und Kryptographie, Vieweg Teubner, 2011.		
Labor		

1. RA in der Praxis	Beispiele, Diskussionen, Teamarbeit	
2. EA als Datenstruktur	Beispiele, Diskussionen, Teamarbeit	
3. DEA aus einem NEA	Beispiele, Diskussionen, Teamarbeit	
4. Hopcroft-Algorithmus	Beispiele, Diskussionen, Teamarbeit	
5. Tabellengesteuerter Scanner	Beispiele, Diskussionen, Teamarbeit	
6. Flex Scanner	Beispiele, Diskussionen, Teamarbeit	
7. KFG als Datenstrukturen	Beispiele, Diskussionen, Teamarbeit	
8. TD-Parsing mit Backtracking	Beispiele, Diskussionen, Teamarbeit	
9. Beseitigung von Linksrekursion	Beispiele, Diskussionen, Teamarbeit	
10. Tabellengesteuerter LL(1)-Parser	Beispiele, Diskussionen, Teamarbeit	
11. LR(1)-Parser Umsetzung	Beispiele, Diskussionen, Teamarbeit	
12. Erzeugen von LR(1)-Tabllen	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] K.D. COOPER, L. TORCZON - Engineering a Compiler,Elsevier Science & Technology, 2011.
[2] C. WAGENKNECHT, HIELSCHER M., Formale Sprachen, abstrakte Automaten und Compiler, Vieweg Teubner, 2009.

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

Die besprochene Theorie wird aus der Perspektive des Aufbau eines Compilers besprochen. Der Inhalt dient also als theoretische Grundlage für die Kentnisse der Informatiker in Software-Unternehmen.

Formale Sprachen und Automaten dienen als Basis für Berechenbarkeitstheorie und Komplexitätstheorie.

10. Prüfungsform

Veranstaltungsart	10.1 Evaluationskriterien	10.2 Evaluationsmethoden	10.3 Anteil an der Gesamtnote
10.4 Vorlesung	Grundkenntnisse.	Schriftliche Prüfung	60%
10.5 Seminar / Übung	Algorithmenanwendung	Labor Arbeiten	40%
10.6 Minimale Leistungsstandards			
• Fähigkeit aus regulären Ausdrücken einen minimalen deterministischen Automaten zu erhalten. • Fähigkeit eine LL(1)-Grammatik zu erkennen und backtrackingfreies Parsen durchzuführen. • Fähigkeit eine LR(1)-Tabellen zu erzeugen und backtrackingfreies Parsen durchzuführen. • Fähigkeit Grammatiken und Automaten anhand der Chomsky-Hierarchie zu erklären. • Für die Laboraufgaben wird man mindestens 5 erhalten müssen. • Für die schriftliche Prüfung wird man mindestens 5 erhalten müssen.			

Ausgefüllt am:	Vorlesungsverantwortlicher	Seminarverantwortlicher
1. Februar 2020	Lect. Dr. Iulian Simion	Lect. Dr. Iulian Simion

Genehmigt im Department am: Departmentdirektor

.....