



National Computing Education Accreditation Council NCEAC



NCEAC.DOC.001-E

COURSE DESCRIPTION

NAME OF INSTITUTION	Lahore Garrison University
PROGRAM (S) TO BE EVALUATED	Computer Science , Fall 2022

Course Description : he course is intended to teach the students the basic techniques that underlie the practice of Compiler Construction. The course will introduce the theory and tools that can be employed in order to perform syntax-directed translation of a high-level programming language into an executable code.

Course Code	CSC373																		
Course Title	Compiler Construction																		
Credit Hours	3+0																		
Prerequisites by Course(s) and Topics	Theory of Automata																		
Assessment Instruments with Weights (homework, quizzes, midterms, final, programming assignments, lab work, etc.)	SESSIONAL (Quizzes, Assignments, Presentations) =25 % Midterm Exam =25 % Final Exam = 50%																		
Course Coordinator	Aneela Mehmood																		
URL (if any)																			
Current Catalog Description																			
Textbook (or Laboratory Manual for Laboratory Courses)	Compilers: Principles, Techniques, and Tools By Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, Contributor Jeffrey D. Ullman, Addison-Wesley Pub. Co., 2nd edition, 2006																		
Reference Material	1. Modern Compiler Design, by Dick Grune, Henri E. Bal, Cerial J. H. Jacobs, Koen G. Langendoen, John Wiley, 2000. 2. Modern Compiler Implementation in C, by Andrew W. Appel, Maia Ginsburg, Contributor Maia Ginsburg, Cambridge University Press, 2004. 3. Modern Compiler Design by Dick Grune, Henri E. Bal, Cerial J. H. Jacobs, Koen G. Langendoen, 2003, John Wiley & Sons.																		
Course Goals	Understand and define grammars in BNF, syntax diagrams, regular expressions. Define tokens using the notation of regular expressions. Convert regular expressions into finite automata. Implement a lexical analyser.																		
Course Learning Outcomes (CLOs):	<table><tr><td>At the end of the course the students will be able to:</td><td>Domain</td><td>BT Level*</td></tr><tr><td>1. Describe the architecture of a compiler, function of its components to generate phases of compiler.</td><td>BT,C</td><td>1</td></tr><tr><td>2. follow compiler concepts to implement scanner and symbol table using programming language.</td><td>BT C,P</td><td>3,3</td></tr><tr><td>3. apply appropriate formal notations to analyze a programming language and to handle the error techniques.</td><td>BT,C</td><td>4</td></tr><tr><td>4. Compute syntax by using various algorithms including top down and bottom-up parser.</td><td>BT,C</td><td>3</td></tr><tr><td>* BT= Bloom's Taxonomy, C=Cognitive domain, P=Psychomotor domain, A= Affective domain</td><td></td><td></td></tr></table>	At the end of the course the students will be able to:	Domain	BT Level*	1. Describe the architecture of a compiler, function of its components to generate phases of compiler.	BT,C	1	2. follow compiler concepts to implement scanner and symbol table using programming language.	BT C,P	3,3	3. apply appropriate formal notations to analyze a programming language and to handle the error techniques.	BT,C	4	4. Compute syntax by using various algorithms including top down and bottom-up parser.	BT,C	3	* BT= Bloom's Taxonomy, C=Cognitive domain, P=Psychomotor domain, A= Affective domain		
At the end of the course the students will be able to:	Domain	BT Level*																	
1. Describe the architecture of a compiler, function of its components to generate phases of compiler.	BT,C	1																	
2. follow compiler concepts to implement scanner and symbol table using programming language.	BT C,P	3,3																	
3. apply appropriate formal notations to analyze a programming language and to handle the error techniques.	BT,C	4																	
4. Compute syntax by using various algorithms including top down and bottom-up parser.	BT,C	3																	
* BT= Bloom's Taxonomy, C=Cognitive domain, P=Psychomotor domain, A= Affective domain																			

Topics Covered in the Course, with Number of Lectures on Each Topic (assume 15-week instruction and one-hour lectures)	Week	Lecture	Topics Covered
	Week 1	1	General introduction
		2	Detailed introduction of Compiler, Preprocessor, Assembler, Linker/Loader, Interpreter
	Week 2	3	types of compiler, cousin of compiler
		4	phases of compiler
	Week 3	5	define Token, Pattern, Lexeme ,input buffer
		6	look a head, error in lexical
	Week 4	7	symbol table, Draw Symbol table
		8	interaction of symbol table with different phases of compiler
	Week 5	9	The Reason Why Lexical Analysis is a Separate Phase, role of lexical analyzer
		10	Regular expression and Grammer
	Week 6	11	Draw integer, reserved word and comparison operator machine
		12	Draw number system machine
	Week 7	13	Lex and Flex generator, Creating a Lexical Analyzer with Lex and Flex
		14	RE, NFA, DFA using subset construction
	Week 8	1 hours	Mid Term
	Week 9	15	e closure move and optimization of DFA
		16	Draw NFA DFA
	Week 10	17	Chomsky law, derivation techniques
		18	error handling techniques, left derivation, right derivation
	Week 11	19	
		20	
	Week 12	21	syntax tree,parsing techniques
		22	top down parsing, predictive parsing
	Week 13	23	LL(0) Problem
		24	difference between LL(0) and LL(1)
	Week 14	25	QUIZ
		26	left recursion and back tracking
	Week 15	27	first and follow set
		28	follow set
	Week 16	29	parse table ,ambiguous grammar
		30	stack table,parse tree
	Week 17	2 hours	Final Term
Laboratory Projects/Experiments Done in the Course			
Programming Assignments Done in the Course			

Instructor Name	Aneela Mehmood
Instructor Signature	
Date	