	THOMAS ADEWUMI UNIVERSITY, OKO COURSE OUTLINE	
Faculty	**Computing and Applied Sciences**	
Department	Mathematical and Computing Science	
Course Title	**COMPILER CONSTRUCTION I**	
Year of Study	3	
Course Code	CSC 310	
Credit Hours	3	
Contact Hours	45	
Mode of Delivery	Classroom Lectures, Recitations, Tutorial	
Mode of assessment		Weight%
Continuous Assessment		30%
Final Examination		70%
Total		100%
Course Lecturers and Instructors	**Prof Francisca Oladipo**	
Course Description	The course introduces the design and implementation of a compiler with emphasis on principles and techniques for program analysis and translation. It also gives an overview of the tools for compiler construction. Lexical analysis, token selection, transition diagrams, and finite automata. The use of context-free grammars to describe syntax, derivations of parse trees, and construction of parsers. Syntax-directed translation schemes; Intermediate code; Symbol table; Code generation; Detection, reporting, recovery, and correction of errors.	
Course Objectives	**Some of the following**: This course would enable the understanding of the following: 1. the major steps involved in compiling a high-levelprogramming language down to a low-level target machine language 2. the major components and functions of a modern compiler and well as the different phases of a compiler.	

	3. the key issues in the construction of production of compilers for real high-level Languages and real targetmachines • 4. how a compiler can generate code to make good use of some target machine characteristics 5. various classes of grammars, languages, and automata, and employ these to solve common software problems 6. work together effectively in teams on a substantial software implementation project 7.	
Learning Outcomes	Upon completion of this course, students will be able to do the following: 1. Write and use simple high-level programming languages to implement. sub-components/ sub-parts of a compiler 2. Use compiler construction tools to generate lexical and syntax analyzers. 3. Understand and be able to explain the functions of the different phases of a compiler. 4. Understand the key issues in the construction of production of compilers for real high-level Languages and real target machines 5.Understand how a compiler can generate code to make good use of some target machine characteristics □ _	
Teaching and Learning	1. **Lectures**: contents of the course will be presented and taught to students in the classroom. Classroom teachings will be supported with practical examples. 2. **Projects**: Group and individual projects will be given to students to solve practical problems in compiler construction. Students will be expected to come to the classroom individually and defend their respective individual projects. 3. **Assignments**: Students will; be asked to solve class assignments with respect to topics covered in the class to examine, test the understanding of and reveal the state of assimilation of the course contents by students.	

	4. **Term Papers**: Students will be asked to write comprehensive term papers on selected sub-topics within the compiler construction course contents. The term papers will help students develop their understanding and in-depth analysis of components of the course contents. In some situations, students will be given a typical compiler construction research paper and will be asked to study, analyze and summarize in their own understanding, gaps and findings from the contents of the paper. Such term papers however will be subjected to thorough plagiarism checks	
Detailed Course Content	Review of compilers assemblers and interpreters, structure and functional aspects of a typical compiler, syntax semantics and, functional relationship between lexical analysis, expression analysis and code generation. Internal form of course programme. Use of a standard compiler (FORTRAN<COBOL/PL) as a working vehicles. Error detection and recovery. Grammars and Languages: the parsing problem. The scanner.	
Weeks	**Detailed Course Outlines**	**Allocated Time**
Week 1, 2	1. Introduction to compiler construction, Reasons why it is essential to study compiler construction, Languages and Translators 2. Review of compilers assemblers and interpreters, structure and functional aspects of a typical compiler	6 hours
Week 3,4	3. Compiler structure and design issues, Phases of compiler, Internal Structure of a Compiler, Compiletime and run- time diagnostics **Recitation and Tutorials** **Continuous Assessment I** -	6 hours
Week 5,6-	4. syntax semantics and, functional relationship between lexical analysis, expression analysis and code generation 5. Symbol tables and their data structures, Lexical analysis, Token, Pattern and lexemes, Operations on languages, Regular expressions, Introduction to Syntax analysis and Applications of Syntax analysis.	6 hours

	Recitation and Tutorials	
Week 7,8	6. Types of Errors encountered during compiler usage and how to recover from such errors **Recitation and Tutorials**	6 hours
Week 9	Error detection and recovery. Grammars and Languages: the parsing problem. The scanner. **Recitation and Tutorials**	3 hours
Week 10-12	8. Use of a standard compiler (FORTRAN<COBOL/PL) as a working vehicle. 9. Practical Sessions **10. Revision and Continuous Assessment II**	9 hours
After week 12	29. Examination	

Recommended Reading Material

1. Aho, Alfred & Sethi, Ravi & Ullman, Jeffrey. *Compilers: Principles, Techniques, and Tools* ISBN 0201100886 The Classic Dragon book.
2. Appel, *A., Modern Compiler Implementation in Java,* 2nd ed., Cambridge University Press, 2002.
3. Appel, Andrew *Modern Compiler Implementation in C/Java/ML* (respectively ISBN 0-521-58390-X,ISBN 0-521-58388-8, ISBN 0-521-58274-1) is a set of cleanly written texts on compiler design, studied from various different methodological perspectives.
4. Brown, P.J. *Writing Interactive Compilers and Interpreters* ISBN 047127609X Useful practical advice, not much theory.
5. Fischer, Charles & LeBlanc, Richard. *Crafting A Compiler* ISBN 0805332014 Uses an ADA like pseudo-code.
6. Fischer, LeBlanc, Cytron*, Crafting a Compiler Implementation*, Addison-Wesley
7. Holub, Allen *Compiler Design in C* ISBN 0131550454 Extensive examples in "C".
8. Hunter, R. *The Design and Construction of Compilers* ISBN 0471280542 several chapters on theory of syntax analysis, plus discussion of parsing difficulties caused by features of various source languages.
9. Keith, D. Cooper & Linda Torczon, "Engineering a Compiler", Morgan Kaufmann Publishers, 2004
10. Pemberton, S. & Daniels, M.C. *Pascal Implementation. The P4 Compiler* ISBN 0853123586 (Discussion) and ISBN 085312437X (Compiler listing) Complete listing and readable commentary for a Pascal compiler written in Pascal.
11. Randy Allen and Ken Kennedy, "Optimising Compilers for Modern Architectures", Morgan Kaufmann Publishers, 2001.
12. Weinberg, G.M. *The Psychology of Computer Programming: Silver Anniversary Edition* ISBN 0932633420 Interesting insights and anecdotes.
13. Wirth, Niklaus *Compiler Construction* ISBN 0201403536 From the inventor of Pascal,

Modula-2 and Oberon-2, examples in Oberon.