CSC 320: Compiler Design: **Mr. Barasa Peter Wawire**

Learning outcomes:

At the end of this course unit, the learner should be able to:

Describe Compiler Organization and Implementation.

Know how to use compiler construction tools, such as generators of scanners and parsers
Explain the basic concepts, definitions and in-depth view of translation and optimization process.

Explain Syntax directed translation and code optimization.
Explain Run-time optimization of programming languages.

CORE REFERENCES

1. Compilers by A.L and Vilma
2. The design and construction of compilers by Robin Hunter.
3. Alfred V. Aho, Monica S. Lam, Ravi Sethi, and Jeffrey D. Ullman. Compilers; Principles, Techniques and Tools. Addison-Wesley, 2007.

OTHER REFERENCES

4. John R. Allen and Ken Kennedy. Optimizing compilers for modern architectures: a dependence-based approach. Morgan Kaufmann, 2001.
5. Andrew W. Appel. Modern Compiler Implementation in ML. Cambridge University Press, 1998.
6. John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. Introduction to
7. Automata Theory, Languages and Computation, 2nd ed. Addison-Wesley, 2001.
8. Scott Owens, John Reppy, and Aaron Turon. Regular-expression derivatives re-examined. J. Funct. Program., 19(2):173–190, 2009.

Contents: Compiler organization and implementations, code optimization. Run-time organization of programming languages. Programming language constructs, their syntax and semantic. Pattern matching in text-strings. Lexical analysis, token selection, transition diagrams and finite automata. Regular expression. Design of token recognizes using transition diagrams. Syntax directed translation. Use of context-free grammars to describe syntax; derivations of parse trees; construction of parsers. Front end and back end processing.

MODE OF DELIVERY

Lectures, tutorials, and laboratory exercises

MODE OF EXAMINATION

END OF SEMESTER EXAMINATION	ASSIGNMENTS & PRACTICALS	CATS	Total
70	10	20	100

COURSE OUTLINE

WEEK	TOPIC	Sub Topics	Learning Resources	Learning Activities
1	-	Reporting & Unit Registration		
2	COMPILER DESIGN – OVERVIEW	Compilation Process: Language Processing System Preprocessor Interpreter Assembler Linker Loader Cross-compiler Source-to-source Compiler	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer.
3	COMPILER DESIGN - ARCHITECTURE	Analysis Phase Synthesis Phase Lexical Analysis Syntax Analysis Semantic Analysis Intermediate Code Generation Code Optimization Code Generation Symbol Table Tokens Specifications of Tokens Alphabets Strings Special Symbols Longest Match Rule Language	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer.

4	COMPILER DESIGN - LEXICAL ANALYSIS	Introduction Regular expressions Nondeterministic finite automata. Converting a regular expression to an NFA. Deterministic finite automata. Converting an NFA to a DFA Minimization of DFAs. Lexers and lexer generators Properties of regular languages	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer.Discussion, Assignments
5	COMPILER DESIGN - SYNTAX ANALYSIS	Context-Free Grammar. Syntax Analyzers. Derivation. Left-most Derivation Right-most Derivation Parse Tree Ambiguity Associativity Precedence Left Recursion Removal of Left Recursion Left Factoring First and Follow Sets First Set Algorithm for calculating First set Follow Set Algorithm for calculating Follow set: Limitations of Syntax Analyzers	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer.

6	COMPILER DESIGN - TYPES OF PARSING	Bottom-up Parsing Top-down Parsing	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer.
7	COMPILER DESIGN - TOP-DOWN PARSER	Recursive Descent Parsing Back-tracking Predictive Parser LL Parser LL Parsing Algorithm	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer. Discussion, Assignments
8	COMPILER DESIGN - BOTTOM-UP PARSER	LR Parser LR Parsing Algorithm LL vs. LR	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question /answer. Discussion, Assignments
9	COMPILER DESIGN - ERROR RECOVERY	Panic mode Statement mode Error productions Global correction Abstract Syntax Trees	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question /answer. Discussion, Assignments

10	COMPILER DESIGN - SEMANTIC ANALYSIS	Semantics Semantic Errors Attribute Grammar Synthesized attributes L-attributed SDT S-attributed SDT Inherited attributes	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer. Discussion, Assignments
11	COMPILER DESIGN - RUN-TIME ENVIRONMENT	Activation Trees Storage Allocation Static Allocation Stack Allocation Heap Allocation Parameter Passing r-value l-value Formal Parameters Actual Parameters Pass by Value Pass by Reference Pass by Copy-restore Pass by Name	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer. Discussion, Assignments
12	COMPILER DESIGN - SYMBOL TABLE	Implementation Operations Insert() lookup () Scope Management	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip Charts LCD projector Internet Fully equipped library	Note taking, question/answer. Discussion, Assignments

13	COMPILER - INTERMEDIATE CODE GENERATION	Intermediate Representation Three-Address Code Quadruples Triples Declarations Indirect Triples Directed Acyclic Graph Peephole Optimization Redundant instruction elimination Unreachable code Flow of control optimization Algebraic expression simplification Strength reduction Accessing machine instructions Code Generator Descriptors Code Generation Machine-independent Optimization Machine-dependent Optimization Basic Blocks statements and loops Basic block identification Control Flow Graph Loop Optimization Dead-code Elimination Partially dead code Partial Redundancy	Computer Pens Photocopy papers Files Foolscaps Marker pens Flip projector Internet LCD	Note taking, question/answer. Discussion, Assignments
14	REVISION			
15	END OF SEMESTER EXAMINATIONS			