Pokhara University
Faculty of Science and Technology

Course Code.: CMP 422 (2 Credits)
Course title: **Compiler Design (2-2-2)**
Nature of the course: Theory & Practice
Program: BE
Level: Bachelor

Full marks: 100
Pass marks: 45
Time per period: 1 hour
Total periods: 30

1. Course Description

The primary objective of this course is to introduce the major concepts of language translation and compiler design and impart the knowledge of practical skills necessary for constructing a compiler. The course includes phases of compiler, parsing, syntax-directed translation, type-checking use of symbol tables, code optimization techniques, intermediate code generation, code generation, and data flow analysis.

2. General Objectives

- Introduce students to the need for a compiler and various phases of a compiler.
- To familiarize students with the concept of designing a compiler for given a set of language features.
- Enhance the knowledge of patterns, tokens & regular expressions for lexical analysis.
- To help the students acquire skills in using lex tool & yacc tool for developing a scanner and parser.
- To enable the students to design and implement LL and LR parsers.
- To make the students able to design algorithms to do code optimization to improve the performance of a program in terms of space and time complexity.
- To familiarize the students with the concept of designing algorithms to generate machine code.

3. Contents in Detail

Specific Objectives	Contents
The structure of a compiler, the science of building a compiler, programming language basics.	**Unit I: Overview of Compilation (4 Hrs)** 1.1 Compiler and Interpreter 1.2 Compiler Structure 1.3 Overview of Translation 1.3.1 The Front End 1.3.2 The Optimizer 1.3.3 The Backend
The Role of the Lexical Analyzer, Input Buffering, Recognition of Tokens, The Lexical-Analyzer Generator Lex, Finite Automata, From Regular Expressions to Automata, Design of a Lexical-Analyzer	**Unit II: Scanners (7 Hrs)** 2.1 The Role of the Scanner 2.2 Recognizing Words 2.2.1 A Formalism of Recognizer 2.2.2 Recognizing Complex Words 2.3 Regular Expression 2.3.1 Formalizing the Notation

Generator, Optimization of DFA-Based Pattern Matchers.	2.3.2 Closure properties of REs 2.4 From Regular Expression to Scanner 2.4.1 Nondeterministic Finite Automata 2.4.2 Regular Expression to NFA: Thomson's Construction 2.4.3 NFA to DFA 2.4.4 Using a DFA as a Recognizer
Introduction, Context-Free Grammars, Writing a Grammar. Learning Regular Expressions. Top-Down Parsing, Bottom-Up Parsing.	**Unit III: Parsers (6 Hrs)** 3.1 Introduction to Syntax Analyzer 3.2 Context Free Grammar 3.3 Top Down Parsing 3.3.1 Transforming a Grammar for Top Down Parsing 3.3.2 Recursive Descent Parsing 3.3.3 Table Driver LL(1) Parsers 3.4 Bottom Up Parsing 3.4.1 The LR(1) Parsing Algorithm 3.4.2 Building LR(1) Table
Intermediate-Code Generation: Variants of Syntax Trees	**Unit IV: Intermediate Representations (5 Hrs)** 4.1 A Taxanomy of Intermediate Representations 4.2 Graphical IRs 4.2.1 Syntax Related Trees 4.2.2 Graphs 4.3 Linear IRs 4.4 Mapping Values to Names 4.5 Symbol Tables 4.5.1 Hash Tables 4.5.2 Building a Symbol Table
Understanding Code Generation Techniques	**Unit V: Code Generation (4 Hrs)** 5.1 Issues in Design of Code Generation 5.2 Basic Blocks and Flow Graphs & Optimization of Basic Blocks 5.3 A Simple Code Generator & Peephole Optimization 5.4 Register Allocation & Assignment, Dynamic Programming Code Generation
Optimizing the generated code.	**Unit 6: Machine Independent Optimization (4 Hrs)** 6.1 The Principal Sources of Optimization 6.2 Loops in Flow Graphs, Constant Propagation, Partial Redundancy Elimination 6.3 Introduction to Data-Flow Analysis 6.4 Solving Data-Flow Equations

5. List of Tutorials

The following tutorial activities of 15 hours per group of maximum 24 students should be conducted to cover all the required contents of this course.

S.N.	Tutorials
1	Describing various phases and structures of a compiler.
2	Understand Regular Expressions, NFA, DFA, Thomson's Construction
3	Course works on Bottom Up and Top Down Parsing. Building LR(1) and LL(1) tables.
4	Creating graphical and linear intermediate code generations. Building Hash tables.
5	Basic exercise on designing code generator and Peephole optimization
6	Designing Flow Graphs, Data Flow Analysis. Solving Data-Flow Equation

6. Practical Works

S.N.	Practical works
1	Programming Language Basics.
2	Using Lex Tools
3	Optimization of DFA Based Pattern Matcher
4	Writing Grammar and Implementing the Grammar
5	Desiging a Basic Recursive Descent Parser, Computing FIRST & FOLLOW Values, Predictive Parsing Table Construction & LL(1) Grammar
6	Simple LR Parser, Creating Parsing Table, Parser Generator
7	Control Flow, Switch-Statements, Intermediate Code for Procedures

7. Evaluation system and Students' Responsibilities

Evaluation System

In addition to the formal exam(s) conducted by the Office of the Controller of Examination of Pokhara University, the internal evaluation of a student may consist of class attendance, class participation, quizzes, assignments, presentations, written exams, etc. The tabular presentation of the evaluation system is as follows.

External Evaluation	Marks	Internal Evaluation	Marks
Semester-End Examination	50	Class attendance and participation	5
		Practical	20
		Quizzes/assignments and presentations	5
		Internal Term Exam	20
Total External	50	Total Internal	50
Full Marks 50+50 = 100			

Students' Responsibilities:

Each student must secure at least 45% marks in the internal evaluation with 80% attendance in the class to appear in the Semester End Examination. Failing to obtain such score will be given NOT QUALIFIED (NQ) and the student will not be eligible to appear in the End-Term examinations. Students are advised to attend all the classes and complete all the assignments within the specified time period. If a student does not attend the class(es), it is his/her sole responsibility to cover the topic(s) taught during the period. If a student fails to attend a formal exam, quiz, test, etc. there won't be any provision for a re-exam.

8. Prescribed Books and References

Text Book

Keith D. Cooper and Linda Torczon, *Engineering a Compiler*, 2nd Edition, Morgan Kaufman

Alfred V. Aho, Monica S. Lam, Ravi Sethi, Jeffry D. Ullman, *Compilers: Principles, Techniques, and Tools*, 2nd Edition, Addison Wesley

Reference Books

Douglas Thain, *Introduction to Compilers and Language Design: Second Edition*, ISBN-13 : 979-8655180260