DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY, TIRUCHIRAPPALLI

COURSE PLAN – PART I			
Name of the programme and specialization	B. Tech Computer Science and Engineering		
Course Title	Compiler Design		
Course Code	CSPC62	**No. of Credits**	4
Course Code of Pre-requisite subject(s)	CSPC62		
Session	Jan 2022	**Section (if, applicable)**	B
Name of Faculty	Sitara K.	**Department**	CSE
Email	sitara@nitt.edu	**Telephone No.**	
Name of Course coordinator(s) (if, applicable)	----		
E-mail	--	**Telephone No.**	--
Course Type	[✓ **Core course**]	☐ **Elective course**	

Syllabus (approved in BoS)

UNIT I Introduction to Compilation

Compilers - Analysis of the source program - Phases of a compiler - Cousins of the Compiler - Grouping of Phases - Compiler construction tools - Lexical Analysis - Role of Lexical Analyzer - Input Buffering - Specification of Tokens.

Lab Component: Tutorial on LEX / FLEX tool, Tokenization exercises using LEX.

UNIT II Syntax Analysis

Role of the parser - Writing Grammars - Context-Free Grammars - Top Down parsing - Recursive Descent Parsing - Predictive Parsing - Bottom-up parsing - Shift Reduce Parsing - Operator Precedent Parsing - LR Parsers - SLR Parser - Canonical LR Parser - LALR Parser.

Lab Component: Tutorial on YACC tool, Parsing exercises using YACC tool.

UNIT III Intermediate Code Generation

Intermediate languages - Declarations - Assignment Statements - Boolean Expressions - Case Statements - Back patching - Procedure calls.

Lab Component: A sample language like C-lite is to be chosen. Intermediate code generation exercises for assignment statements, loops, conditional statements using LEX/YACC.

UNIT IV Code Optimization and Run Time Environments

Introduction - Principal Sources of Optimization - Optimization of basic Blocks - DAG representation of Basic Blocks - Introduction to Global Data Flow Analysis - Runtime Environments - Source Language issues - Storage Organization - Storage Allocation strategies - Access to non-local names - Parameter Passing - Error detection and recovery.

Lab Component: Local optimization to be implemented using LEX/YACC for the sample language.

UNIT V Code Generation

Issues in the design of code generator - The target machine - Runtime Storage management - Basic Blocks

and Flow Graphs - Next-use Information - A simple Code generator - DAG based code generation - Peephole Optimization.
Lab Component: DAG construction, Simple Code Generator implementation, DAG based code generation using LEX/YACC for the sample language.

COURSE OBJECTIVES

- To introduce the major concept areas in compiler design and know the various phases of the compiler
- To understand the various parsing algorithms and comparison of the same
- To provide practical programming skills necessary for designing a compiler
- To gain knowledge about the various code generation principles
- To understand the necessity for code optimization

COURSE OUTCOMES (CO)

Course Outcomes	**Aligned Programme Outcomes (PO)**
1. Apply the knowledge of LEX & YACC tool to develop a scanner and parser	1, 3, 6, 10, 12
2. Design and develop software system for backend of the compiler	1, 3, 4, 10
3. Suggest the necessity for appropriate code optimization techniques	1, 3, 8, 10, 11
4. Conclude the appropriate code generator algorithm for a given source language	2, 4, 6, 8, 11, 12
5. Design a compiler for any programming language	1, 2, 6, 7, 10, 11

COURSE PLAN – PART II

COURSE OVERVIEW

A compiler is a system software that translates the code written in one language to some other language without altering the meaning of the program. A compiler should also produce the target code which is efficient and optimized.

Compiler design principles provide an in-depth view of translation from a source (high-level) language to target (low-level typically assembly) language followed by optimization. The compiler involves six phases namely, lexical, syntax, semantic analysis as front end, and intermediate code generation, code generation and optimization as back-end. This course discusses these six phases of the compiler in detail by providing appropriate algorithms, methodologies and its implementation at all phases.

COURSE TEACHING AND LEARNING ACTIVITIES

S.No.	**Week/Contact Hours**	**Topic**	**Mode of Delivery – Online (MS Teams)**
1	3 hours	**Unit 1 Introduction to compilation:** Compilers - Analysis of the source program, Phases of the compiler – Cousins of the compiler - Grouping of phases, Compiler construction tools	*PPT*
2	3 hours	Lexical Analysis - Role of Lexical Analyzer - Specification and Recognition of tokens, Review of NFA, DFA, Regular expression (Assignment problems)	*PPT*
3	1 hour	Input Buffering, Sentinels – algorithm	*PPT*

4	2 hours	**UNIT II Syntax Analysis:** Role of the parser, Writing Grammars - Context free grammar, examples, parsing – top down and bottom up	*PPT*
5	3 hours	Top Down parsing - Recursive Descent Parsing - Predictive Parsing - Bottom-up parsing - Shift Reduce Parsing - Example Problems	*PPT*
6	2 hours	Operator Precedent Parsing - Example Problems	*PPT*
7	4 hours	LR Parsers - SLR Parser - Canonical LR Parser - LALR Parser - Example Problems	*PPT*
8	3 hours	**UNIT III Intermediate Code Generation:** Intermediate languages - Declarations - Assignment Statements - SDT for Arrays	*PPT*
9	1 hour	Boolean Expressions	*PPT*
10	2 hours	Loops - Case Statements - Back patching - Procedure calls	*PPT*
11	3 hours	**UNIT IV Code Optimization and Run Time Environments:** Introduction - Principal Sources of Optimization - Optimization of basic Blocks - DAG representation of Basic Blocks - Runtime Environments	*PPT*
12	3 hours	Source Language issues - Storage Organization - Storage Allocation strategies - Access to non-local names - Parameter Passing - Error detection and recovery - Introduction to Global Data Flow Analysis	*PPT*
13	3 hours	**UNIT V Code Generation:** Issues in the design of code generator - The target machine - Runtime Storage management - Basic Blocks and Flow Graphs	*PPT*
14	3 hours	Next-use Information - A simple Code generator - DAG based code generation - Peephole Optimization	*PPT*
Lab Activities			
1	Week 1, 2 hours	Lexical Analyser generator – Introduction to LEX tool and sample programs	*PPT/ Virtual lab*
2	Week 2, 2 hours	Tokenization exercises using LEX	*PPT/ Virtual lab*
3	Week 3, 2 hours	Introduction to YACC tool and sample programs	*PPT/ Virtual lab*
4	Week 4, 2 hours	Parsing exercises using YACC tool	*PPT/ Virtual lab*
5	Week 5, 2 hours	Intermediate code generation exercises for assignment statements and expressions using LEX/YACC	*PPT/ Virtual lab*
6	Week 6, 2 hours	Intermediate code generation exercises for loops and conditional statements using LEX/YACC	*PPT/ Virtual lab*

7	Week 7, 2 hours	Local optimization to be implemented using LEX/YACC for the sample language	*PPT/ Virtual lab*
8	Week 8, 2 hours	Local optimization to be implemented using LEX/YACC for the sample language	*PPT/ Virtual lab*
9	Week 9, 2 hours	DAG construction, Simple Code Generator implementation using LEX/YACC for the sample language.	*PPT/ Virtual lab*
10	Week 10, 2 hours	DAG based code generation using LEX/YACC for the sample language.	*PPT/ Virtual lab*

Text Books

1. Alfred V. Aho, Jeffrey D Ullman, "Compilers: Principles, Techniques and Tools", Pearson Education Asia, 2012.
2. Jean Paul Tremblay, Paul G Serenson, "The Theory and Practice of Compiler Writing", BS Publications, 2005.
3. Dhamdhere, D. M., "Compiler Construction Principles and Practice", Second Edition, Macmillan India Ltd., New Delhi, 2008.

Reference books

1. Allen I. Holub, "Compiler Design in C", Prentice Hall of India, 2003.
2. C. N. Fischer, R. J. LeBlanc, "Crafting a compiler with C", Benjamin Cummings, 2003.
3. Henk Alblas, Albert Nymeyer, "Practice and Principles of Compiler Building with C", PHI, 2001.
4. Kenneth C. Louden, "Compiler Construction: Principles and Practice", Thompson Learning, 2003.

COURSE ASSESSMENT METHODS - Theory

Sl. No.	**Mode of Assessment**	**Week/Date**	**Duration**	**% Weightage**
1	Cycle Test 1	14/02/2022 to 16/02/2022	1 hour	20
2	Cycle Test 2	18/04/2022 to 20/04/2022	1 hour	20
CP A	**Compensation Assessment***	**After completion of Cycle Test 2**	**1 hour**	**20**
3	Final Assessment *	As per academic schedule	As per institute norms	30
Total Theory Marks				70%

COURSE ASSESSMENT METHODS - Practical

4	Continuous Assessment	Weekly lab	--	10
5	Cycle Test	17/03/2022	1 hour	10
CP A	**Compensation Assessment Lab***	**After completion of Cycle Test**		**10**
6	Final Assessment* Practical	As per academic schedule	As per institute norms	10

Total Practical Marks	30%
Total Marks	100%

***mandatory; refer to guidelines on page 6**

COURSE EXIT SURVEY (mention the ways in which the feedback about the course shall be assessed)

1. Students' feedback through class committee meetings
2. Feedbacks are collected before final examination through MIS or any other standard format followed by the institute
3. Students, through their Class Representatives, may give their feedback at any time to the course faculty which will be duly addressed.

COURSE POLICY (preferred mode of correspondence with students, compensation assessment policy to be specified)

MODE OF CORRESPONDENCE (email/ phone etc)

Email

COMPENSATION ASSESSMENT POLICY

1. One compensation assessment will be given after completion of Cycle Test 1 and 2 for the students those who are absent for any assessment due to genuine reason.
2. Compensatory assessments would cover the syllabus of Cycle tests 1 & 2
3. Prior permission and required document must be submitted for absence.

ATTENDANCE POLICY (A uniform attendance policy as specified below shall be followed)

- **At least 75% attendance in each course is mandatory.**
- **A maximum of 10% shall be allowed under On Duty (OD) category.**
- Students with **less than 65% of attendance** shall be prevented from writing the final assessment and **shall be awarded 'V' grade.**

ACADEMIC DISHONESTY & PLAGIARISM

- Possessing a mobile phone, carrying bits of paper, talking to other students, copying from others during an assessment will be treated as punishable dishonesty.
- Zero mark to be awarded for the offenders. For copying from another student, both students get the same penalty of zero mark.
- The departmental disciplinary committee including the course faculty member, PAC chairperson and the HoD, as members shall verify the facts of the malpractice and award the punishment if the student is found guilty. The report shall be submitted to the Academic office.

  The above policy against academic dishonesty shall be applicable for all the programmes.

ADDITIONAL INFORMATION

1. Faculty and lab assistants are available for consultation during the time intimated to the students then and there.
2. Relative grading adhering to the instructions from the office of the Dean (Academic) will be adopted for the course.

FOR APPROVAL

Course Faculty [illegible] **CC-Chairperson** R. Leela **HOD** [illegible]

Guidelines

a) The number of assessments for any theory course shall range from 4 to 6.

b) Every theory course shall have a final assessment on the entire syllabus with at least 30%weightage.

c) One compensation assessment for absentees in assessments (other than final assessment) is mandatory. Only genuine cases of absence shall be considered.

d) The passing minimum shall be as per the regulations.

e) Attendance policy and the policy on academic dishonesty & plagiarism by students are uniform for all the courses.

f) Absolute grading policy shall be incorporated if the number of students per course is less than 10.

g) Necessary care shall be taken to ensure that the course plan is reasonable and is objective.