

**Computer Science Paper 6**  
**Compiler Construction**  
**[DESC-III]**

<b>Semester: VI</b>	<b>Credits: 2</b>	<b>Subject Code: BS62206</b>	<b>Lectures: 36</b>
---------------------	-------------------	------------------------------	---------------------

**Course Outcomes:**

**At the end of this course, the learner will be able to:**

- Explain the phases of the compiler.
- Apply the process of scanning and parsing of source code.
- Use techniques in code generation and code optimization.
- Implement tools like LEX and YACC.
- Design the conversion code written in source language to machine language.

<b>Unit 1: Introduction</b>	<b>5</b>
<ul style="list-style-type: none"><li>● Definition of Compiler, Aspects of compilation.</li><li>● The structure of Compiler.</li><li>● Phases of Compiler – Lexical Analysis, Syntax Analysis, Semantic Analysis, Intermediate Code generation, code optimization, code generation.</li><li>● Error Handling.</li><li>● Introduction to one pass &amp; Multipass compilers, cross compiler, Bootstrapping.</li></ul>	
<b>Unit 2: Lexical Analysis (Scanner)</b>	<b>4</b>
<ul style="list-style-type: none"><li>● Review of Finite automata as a lexical analyzer,</li><li>● Applications of Regular Expressions and Finite Automata (lexical analyzer, searching using RE), Input buffering, Recognition of tokens.</li><li>● LEX: A Lexical analyzer generator (Simple Lex Program)</li></ul>	
<b>Unit 3: Syntax Analysis (Parser)</b>	<b>14</b>
<ul style="list-style-type: none"><li>● Definition, Types of Parsers</li><li>● Top-Down Parser<ul style="list-style-type: none"><li>○ Top-Down Parsing with Backtracking: Method &amp; Problems</li><li>○ Drawbacks of Top-Down parsing with backtracking</li><li>○ Elimination of Left Recursion (direct &amp; indirect)</li><li>○ Need for Left Factoring &amp; examples</li></ul></li><li>● Recursive Descent Parsing: Definition, Implementation of Recursive Descent Parser Using Recursive Procedures</li><li>● Predictive [LL (1)] Parser (Definition, Model)</li></ul>	

<b>Board of Studies</b>	<b>Name</b>	<b>Signature</b>
Chairperson (HoD)	Mrs. Ashwini Kulkarni	

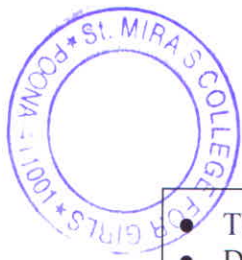


<ul style="list-style-type: none"> <li>○ Implementation of Predictive Parser [LL (1)]</li> <li>○ FIRST &amp; FOLLOW</li> <li>○ Construction of LL (1) Parsing Table</li> <li>○ Parsing of a String using LL (1) Table</li> <li>● Bottom-Up Parsers             <ul style="list-style-type: none"> <li>○ Shift Reduce Parser</li> <li>○ Reduction, Handle, Handle Pruning</li> <li>○ Stack Implementation of Shift Reduce Parser (with examples)</li> </ul> </li> <li>● Operator Precedence Parser             <ul style="list-style-type: none"> <li>○ Basic Concepts,</li> <li>○ Operator Precedence Relations form Associativity &amp; Precedence</li> <li>○ Operator Precedence Grammar</li> <li>○ Algorithm for LEADING &amp; TRAILING (with ex.)</li> <li>○ Algorithm for Operator Precedence Parsing (with ex.) Precedence Functions</li> </ul> </li> <li>● LR Parser:             <ul style="list-style-type: none"> <li>○ Model, Types</li> <li>○ SLR (1)- Method &amp; examples.</li> <li>○ Canonical LR-Method &amp; examples.</li> <li>○ LALR-Method &amp; examples.</li> </ul> </li> <li>● YACC –program sections, simple YACC program for expression evaluation</li> </ul>	
---	--

<p><b>Unit 4: Syntax Directed Definition</b></p>	<p><b>5</b></p>
<ul style="list-style-type: none"> <li>● Syntax Directed Definitions (SDD)</li> <li>● Inherited &amp; Synthesized Attributes</li> <li>● Annotated Parse Tree, Evaluating an SDD at the nodes of a Parse Tree, Example</li> <li>● Evaluation Orders for SDD's</li> <li>● Dependency Graph</li> <li>● S-Attributed Definition</li> <li>● L-Attributed Definition</li> <li>● Applications of SDT</li> <li>● Translation Schemes-Definition</li> </ul>	

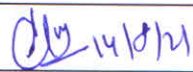
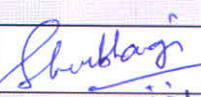
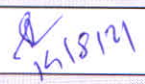
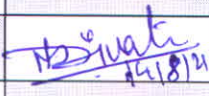
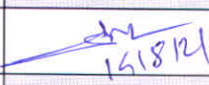
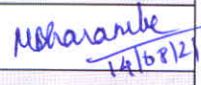
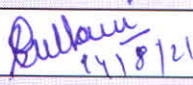
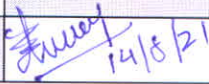
<p><b>Unit 5: Code Generation and Optimization</b></p>	<p><b>8</b></p>
<ul style="list-style-type: none"> <li>● Compilation of expression – Concepts of operand descriptors and register descriptors with example. Intermediate code for expressions – postfix notations,</li> <li>● Triples, Quadruples and Expression trees.</li> <li>● Code Optimization – Optimizing transformations – compile time evaluation, elimination of common sub expressions, dead code elimination, frequency reduction, strength reduction.</li> </ul>	

Board of Studies	Name	Signature
Chairperson (HoD)	Mrs. Ashwini Kulkarni	



<ul style="list-style-type: none"> <li>• Three address code</li> <li>• DAG for Three address code-The Value-number method for constructing DAG's.</li> <li>• Definition of basic block, Basic blocks, and flow graphs</li> <li>• Directed acyclic graph (DAG) representation of basic block.</li> <li>• Issues in design of code generator.</li> </ul>	
--	--

<b>Recommended Reference Books:</b>
<ul style="list-style-type: none"> <li>• Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman <i>Compilers: Principles, Techniques, and Tools</i>; 2004.</li> <li>• Alfred V. Aho, Jeffrey D. Ullman, Narosa Publication House <i>Principles of Compiler Design</i>; 2002. O'reilly Publication <i>LEX &amp; YACC</i>, 2<sup>nd</sup> edition; 2012</li> </ul>

Board of Studies	Name	Signature(in white cell)
Chairperson (HoD)	Mrs. Ashwini Kulkarni	 14/8/21
Faculty	Mrs. Shubhangi Jagtap	
Faculty	Mrs. Smita Borkar	 14/8/21
Subject Expert (Outside SPPU)	Dr. Manisha Divate	 14/8/21
Subject Expert (Outside SPPU)	Mr. Aniket Nagane	 14/8/21
VC Nominee	Dr. Manisha Bharambe	 14/8/21
Industry Expert	Mrs. Snehal Biyala	 14/8/21
Alumni	Ms. Mamta Choudhary	 14/8/21

Board of Studies	Name	Signature
Chairperson (HoD)	Mrs. Ashwini Kulkarni	