|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SCSA1604** | **COMPILER DESIGN** | **L** | **T** | **P** | **Credits** | **Total Marks** |
| **3** | **\*** | **0** | **3** | **100** |

**COURSE OBJECTIVES**

* + To understand, design and implement a lexical analyser.
  + To understand, design and implement a parser.
  + To understand, design code generation schemes.
  + To understand optimization of codes and runtime environment.
  + To design and develop a compiler.

**UNIT 1 INTRODUCTION TO COMPILERS 9 Hrs.**

Structure of compiler – Functions and Roles of lexical phase – Input buffering – Representation of tokens using regular expression –LEX- Properties of regular expression – Finite Automata – Regular Expression to Finite Automata – NFA to Minimized DFA.

**UNIT 2 PARSER 9 Hrs.**

Role of Parser-Context-free Grammar – Derivations and Parse Tree - Types of Parser –Bottom Up: Shift Reduce Parsing - Operator Precedence Parsing, SLR parser- Top Down: Recursive Decent Parser - Non-Recursive Decent Parser-Error handling and Recovery in Syntax Analyzer-YACC.

**UNIT 3 INTERMEDIATE CODE GENERATION 9 Hrs.**

Types of Intermediate Code – Representation of three address code - Syntax Directed Translation scheme- Intermediate code generation for: Assignment statements - Boolean statements - Switch-case statement –Procedure call - Symbol Table Generation.

**UNIT 4 CODE OPTIMIZATION 9 Hrs.**

Principle sources of Optimization - Basic Blocks and Flow Graphs - Loop Optimization & its types – DAG - Peephole optimization - Dominators - Global Data Flow Analysis.

**UNIT 5 CODE GENERATION 9 Hrs.**

Issues involved in Code generation – Register allocation – Conversion of three address code to assembly code using code generation algorithm – Examples – Procedure for converting assembly code to machine code – Case study.

**Max.45 Hrs.**

**COURSE OUTCOMES**

On completion of the course, student will be able to

1. Describe the various stages involved in the design of a compiler.
2. Construct grammars and lexical rules for a programming language.
3. Explain the syntactic and semantic structure in compiler design.
4. Evaluate various methods of optimizations on intermediate code
5. Generate target code for any source code
6. Design, develop, and implement a compiler for any programming language.

**TEXT / REFERENCE BOOKS**

1. Alfred V Aho, Monica S. Lam, Ravi Sethi and Jeffrey D Ullman, “Compilers – Principles, Techniques and Tools”, 2nd Edition, Pearson Education, 2007.
2. Steven S. Muchnick, “Advanced Compiler Design and Implementation,“Morgan Kaufmann Publishers – Elsevier Science, India, Indian Reprint 2003.
3. V. Raghavan, Principles of Compiler Design‖, Tata McGraw Hill Education Publishers, 2010.

**END SEMESTER EXAM QUESTION PAPER PATTERN**

**Max. Marks : 100 Exam Duration : 3 Hrs.**

**PART A :** 10 Questions of 2 marks each-No choice **20 Marks**

**PART B :** 2 Questions from each unit with internal choice, each carrying 16 marks **80 Marks**