MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)
Rasipuram - 637 408, Namakkal Dist., Tamil Nadu.

Estd. 2000

Department of Computer Science and Engineering
Question Bank - Academic Year (2021-2022)

Course Code & Course Name :19CSC20 & Compiler Design

Year/Sem/Sec : III / VI / A&B

Unit-I: INTRODUCTION TO COMPILERS

Part-A (2 Marks)

1. What is a Compiler?
2. Write the cousins of compiler.
3. Define translator.
4. List out the phases of Compiler.
5. Differentiate Compiler, Assembler and Interpreter.
6. State some compiler construction tools?
7. Define Error.
8. List out errors encountered in different phases.
9. Describe working principle of Compiler and Interpreter.
10. List out the sources of Errors.

Part-B (16 Marks)

1. Explain the phases of a compiler in detail with example. (16)
2. Briefly explain Compiler construction tools. (16)
3. Explain Briefly grouping of phases. (16)
4. Explain errors Encountered in Different Phases. (16)
5. Explain the working principle of Translators, Compilation and Interpretation. (16)

Unit-II : LEXICAL ANALYSIS

Part-A (2 Marks)

1. What is the output of syntax analysis phase? What are the three general types of parsers for grammars?
2. List some different strategies that a parser can employ to recover from a syntactic error?
3. Differentiate parse tree and Syntax tree ?
4. Differentiate Token, Lexeme, and pattern.
5. List out error recovery for syntactic phase recover.
6. Define Regular expression.

7. List the operations on languages.
8. Define Finite Automata.
9. Differentiate NFA and DFA.
10. Write a regular definition to represent date in the following format : JAN-5th2014.

Part-B (16 Marks)

1. Give a neat sketch of role of parser and explain the error recovery strategies on parsing. (16)
2. Explain the role Lexical Analyzer and Issues of Lexical Analyzer. (16)
3. Explain Briefly about Input buffering techniques. (16)
4. Give the minimized DFA for the following expression (a / b)*a (a / b). (16)
5. Write an algorithm for constructing a DFA from a regular expression. Describe with the RE= (a/b)*abb by using subset construction method. (16)

Unit-III : SYNTAX ANALYSIS

Part-A (2 Marks)

1. Differentiate Top down and Bottom up parsing.
2. What is Shift-Reduce parsing?
3. Define handle. What do you mean by handle pruning?
4. Define LR (0) items.
5. What do you mean by viable prefixes?
6. What is meant by an operator grammar? Give an example
7. List the disadvantages of operator precedence parsing?
8. State error recovery in operator-Precedence Parsing.
9. Why LR parsing is attractive one grammar
10. What are kernel and non kernel items?

Part-B (16 Marks)

1. Evaluate the parse tree for the input string w=cad using topdown parser S → cAd
   A→ab | a
2. Show SLR parsing table for the following grammar (16)
   S → Aa | bAc | Bc | bBa
   A → d
   B → d
   And parse the sentence "bdc" and "dd".
3. Find the LALR for the given grammar and parse the sentence (a + b) * c (16)
   E → E + T | T , T → T * F | F , F → (E) / id.
4. Give an LALR parser for the following grammar and parse the input id = id (16)
   S → L = R | R
   L → * R | id
   R → L

5. Show a predictive parser for the following grammar (16)
$S \rightarrow (L) | a$
$L \rightarrow L,S | S$

6. Give SLR parsing table for the following grammar (16)
$E \rightarrow E + T | T$
$T \rightarrow T F | F$
$F \rightarrow F * | a | b$

7. Evaluate predictive parsing table for the grammar and find moves made by predictive parser on input id+id*id and find FIRST and FOLLOW $E \rightarrow E+T | T$;; $T \rightarrow T*F | F$;; $F \rightarrow (E) | id$ (16)

8. Write down the algorithm to eliminate left-recursion and left-factoring and apply both to the following grammar $E \rightarrow E + T | E - T | T$; $T \rightarrow a | b | (E)$ Construct predictive parsing table for the above grammar and parse(a+b)-a (16)

9. Check whether the following grammar is a LL (1) grammar S->iEtS / iEtSeS /a, E->b (16)

Unit-IV : SYNTAX DIRECTED TRANSLATION & INTERMEDIATE CODE GENERATION

Part-A (2 Marks)

1. What are the benefits of using machine-independent intermediate form?
2. List the three kinds of intermediate representation.
3. How can you generate three-address code?
4. What is a syntax tree? Draw the syntax tree for the assignment statement
5. Define three-address code.
6. What is called an abstract or syntax tree?
7. Describe Boolean expressions.
8. Define back patching.
9. Pass by Value, Pass by Reference, Pass by Copy-restore.
10. Define procedure call.

Part-B (16 Marks)

1. How the compiler will allocate memory. Explain. (16)
2. Briefly explain about Boolean expression. (16)
3. Describe in detail the syntax directed translation of case statements (16)
4. How Back patching can be used the generate code for Boolean expressions and flow of control statements. (16)
5. Explain in detail the Run time environments. (16)
6. Describe briefly Intermediate Code Generation. (16)

Unit-V : CODE OPTIMIZATION AND CODE GENERATION

Part-A (2 Marks)

1. What are basic blocks?
2. What is a flow graph?
3. Mention the applications of DAGs.
4. What are the advantages and disadvantages of register allocation and assignments?
5. List the types of addressing modes.
6. What is input to code generator?
7. What are the primary structure preserving transformations on basic blocks?
8. Define DAG.
9. What are the issues in the design of code generators?
10. What is meant by Dead Code?
11. Draw the DAG for a := b * -c + b * -c

Part-B (16 Marks)

1.	Explain Principal sources of optimization.	(16)
2.	Describe in detail about optimization of basic blocks with example.	(16)
3.	Explain DAG.	(16)
4.	Describe issues in design of code generator.	(16)
5.	Write a simple code generator algorithm	(16)
6.	Explain in detail about single pass and two pass compilers.	(16)

Course Faculty **HoD**