AUTOMATA THEORY AND COMPILER DESIGN
(MCA2141)

Time Allotted : 2½ hrs **Full Marks : 60**

Figures out of the right margin indicate full marks.

Candidates are required to answer Group A and any 4 (four) from Group B to E, taking one from each group.

Candidates are required to give answer in their own words as far as practicable.

Group – A

1. Answer any twelve: **12 × 1 = 12**

Choose the correct alternative for the following

(i) Conventionally, lower case Greek alphabets are members of
(a) V_N (b) $\sum$
(c) Strings of terminals (d) $(V_N \cup \sum)^*$

(ii) S –>aSa| bSb| a| b;the language generated by the grammar is the set of
(a) All palindromes
(b) All odd length palindromes
(c) Strings beginning and ending with the same symbol
(d) All even length palindromes

(iii) The number of states required to minimize a DFA with redundant states is:
(a) Always less than original (b) Always equal to original
(c) Equal or less than original (d) More than original

(iv) The phase of a compiler where an unmatched bracket can be detected is
(a) Lexical Analysis (b) Syntax Analysis
(c) Semantic Analysis (d) Code Optimization

(v) Which compiler is written in the same source programming language that it intends to compile?
(a) Language Rewriter (b) Cross compiler
(c) Bootstrap compiler (d) One pass compiler

(vi) The Pumping Lemma is used to
(a) Minimize DFAs (b) Prove non-regularity of languages
(c) Convert CFG to CNF (d) Generate tokens

(vii) Which optimization technique is used to reduce the multiple jumps?
(a) Latter optimization technique (b) Peephole optimization technique
(c) Local optimization technique (d) Loop optimization technique

(viii) Which of the following bottom-up parsing methods is also known as a shift-reduce parser?
(a) LL parsing (b) Predictive parsing
(c) Recursive descent parsing (d) LR parsing

(ix) Which of the following is not an intermediate code form?
(a) Postfix notation (b) Syntax tree
(c) Three address code (d) Triple

(x) Which of the following is NOT an intermediate representation?
(a) Three Address Code (b) Quadruples
(c) Abstract Syntax Tree (d) High-Level Source Code

Fill in the blanks with the correct word

(xi) The _____ phase involves translating the intermediate code into machine code or assembly language.

(xii) The _____ of an NFA is the set of states that can be reached from a given state with a sequence of zero or more input symbols.

(xiii) The language accepted by a finite automaton is called a _____ language.

(xiv) In loop optimization, the process of _____ involves restructuring a loop to improve its performance by reducing the number of iterations or minimizing overhead within the loop.

(xv) In compiler design, the lexical analysis phase produces a sequence of _____.

Group - B

2. (a) Explain all types of Non-determinism in NFA with suitable examples. *[(CO1)(Understand/LOCQ)]*
(b) Form a regular expression to express a C language identifier. *[(CO1)(Apply/IOCQ)]*
(c) Given the regular expression $(a|b)^*$ describe the language it generates. *[(CO1)(Apply/IOCQ)]*

6 + 3 + 3 = 12

3. Explain the following terms in context of DFA with suitable examples:
i. Unreachable State,
ii. Dead State,
iii. Non-distinguishable State. *[(CO1)(Understand/LOCQ)]*

(4 × 3) = 12

Group - C

4. (a) Explain the Chomsky hierarchy of grammars with examples. *[(CO3)(Understand/LOCQ)]*
(b) Convert the grammar $S \rightarrow AB$, $A \rightarrow aA|\varepsilon$, $B \rightarrow bB|\varepsilon$ into Chomsky Normal Form. *[(CO3)(Apply/IOCQ)]*

6 + 6 = 12

5. (a) Explain The role of the LEX specifications section and how it is used to define regular expressions and actions. *[(CO2)(Understand/LOCQ)]*

(b) How LEX handles overlapping regular expressions and the concept of "longest match." *[(CO2)(Apply/IOCQ)]*

(c) Describe the role of Lexical Analyzer in the design of a compiler. *[(CO2)(Understand/LOCQ)]*

4 + 4 + 4 = 12

Group - D

6. (a) Create a LL(1) Parsing Table from the following CFG.
E -> T + E | T, T -> id | id * T | (E) *[(CO3)(Analyse/HOCQ)]*

(b) Parse a suitable string using Table created from the CFG given above. *[(CO4)(Remember/LOCQ)]*

6 + 6 = 12

7. (a) Create the SLR parsing table for the following grammar.
$E \rightarrow E + T \mid T, T \rightarrow T * F \mid F, F \rightarrow id$ *[(CO4)(Apply/HOCQ)]*

(b) Parse a string that can be derived from the given grammar using the SLR table you have created. *[(CO4)(Evaluate/HOCQ)]*

6 + 6 = 12

Group - E

8. (a) Given the following C-like code snippet, generate the corresponding three-address code (TAC) for it:

```
if (x < y)
{
z = x + y * 2;
}
else
{
z = x - y / 2;
}
```

Show the steps for generating the three-address code, including any necessary conditional jumps. *[(CO5)(Apply/IOCQ)]*

(b) Optimize the generated three-address code by performing common sub-expression elimination and dead code elimination (if applicable). Clearly show the changes made during the optimization. *[(CO6)(Apply/IOCQ)]*

6 + 6 = 12

9. (a) Generate the three-address code (TAC) for the following expression:
A = (B + C) ∗ (D – E) + (B + C)
Show each step clearly and explain how temporary variables are introduced. *[(CO5)(Apply/IOCQ)]*

(b) Construct a Directed Acyclic Graph (DAG) for the same expression. Using the DAG, identify any common sub-expressions and demonstrate how the expression can be optimized. Compare the optimized intermediate code to the un-optimized version from the previous code. *[(CO6)(Evaluate/HOCQ)]*

6 + 6 = 12

Cognition Level	LOCQ	IOCQ	HOCQ
Percentage distribution	40	35	25