MCA/5TH SEM/MCAP 3182/2019

COMPILER DESIGN
(MCAP 3182)

Time Allotted : 3 hrs **Full Marks : 70**

Figures out of the right margin indicate full marks.

Candidates are required to answer Group A and any 5 (five) from Group B to E, taking at least one from each group.

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

Group – A
(Multiple Choice Type Questions)

1. Choose the correct alternative for the following: **10 × 1 = 10**

(i) The optimization which avoids test at every iteration is
(a) Loop Unrolling (b) Loop Jamming
(c) Constant Folding (d) Loop Invariant Removal.

(ii) A dangling reference is a
(a) reference pointing to storage which is freed
(b) reference to an object that no longer exists
(c) reference pointing to storage which is still in use
(d) reference pointing to uninitialized storage.

(iii) S → CC, C → c C | d The grammar is
(a) LL(1) (b) SLR(1) but not LL(1)
(c) LALR(1) but not SLR(1) (d) LR(1) but not LALR(1).

(iv) Given an arbitrary non-deterministic finite automaton (NFA) with n states, the maximum number of states in an equivalent minimized DFA is at least
(a) n^2 (b) 2^n
(c) 2n (d) n!

(v) Context Sensitive grammar is accepted by
(a) Turing Machine (b) Linearly bounded automaton
(c) Pushdown automaton (d) Finite automaton.

(vi) The graph that shows basic blocks and their successor relationship is called
(a) DAG (b) Flow Graph
(c) Control Graph (d) Hamilton Graph.

(vii) Given a grammar G = <{ E }, { id, + }, P, E> where, P is given by E → E + E | id. Then FOLLOW (E) will contain
(a) {$} (b) { + }
(c) { $, + } (d) { $, id, + }.

(viii) 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.

(ix) If G is S → aS| bS | a | b, then L(G) is
(a) $\{a,b\}^*$ (b) $\{a,b\}^+$
(c) {a,b,S} (d) {a,b}.

(x) The regular expression for the language of all strings that have zero or more a's followed by zero or more b's is
(a) ab (b) (a + b)
(c) a*b* (d) $(ab)^+$.

Group – B

2. (a) What is the significance of the signal 'get next token'? Explain in terms of the working principle of a compiler.

(b) Write a LEX program that accepts a English Statement and coverts lower case words into their upper case counterparts.

(c) Explain the order of the phase 'Code Optimization' in terms of the steps of compilation. Give the standard views on it.

3 + 5 + 4 = 12

3. (a) Write a regular expression to recognize a signed real number in mantissa-exponent form.

(b) Write down the rules of Thompson Construction with suitable example.

(c) Convert the following regular expression into NFA and then to minimal state DFA. $0(0|1)^*1$.

3 + 4 + 5 = 12

Group – C

4. (a) In which situation there can be a S-R conflict in a LALR parser. Explain with proper example.

(b) Construct a LALR parsing table from the following grammar. S→AA, A→aA, A→b.

3 + 9 = 12

5. (a) Consider the following grammar:
S → aABb, A → c | ε, B → d | ε. Create the LL(1) parsing table for this grammar. Then parse a suitable string using the table.

(b) Check whether the following grammar is LL(1).
S → A | B, A → cA + b | a, B → cB + a | b

9 + 3 = 12

Group – D

6. (a) Explain Activation Record with proper example. What do you mean by activation of a procedure?

(b) Explain the following terms with suitable example: (i) call by value and (ii) copy restore.

(3 + 3) + (3 + 3) = 12

7. (a) Comment on the use of symbol table for the compiler. What information should be associated with a symbol name in the symbol table?

(b) Describe data structures for the symbol table and compare them.

(2 + 4) + 6 = 12

Group – E

8. (a) Construct DAG for the basic block whose code is given below:

```
t1 = b + c
t2 = d * e
t3 = b + c
t4 =  t2 * t3
t5 = t4 * f
x = t1 – t5.
```

(b) Consider the source code:

```
x = y + z
p = z + q + y
```

Explain how commutativity and associativity can be used to generate more efficient code from the DAG of this source code.

6 + 6 = 12

9. Explain the following terms with suitable examples: **(3 × 4) = 12**

(i) Constant Folding.
(ii) Common Sub Expression Elimination.
(iii) Loop Unrolling.