Compiler Design (CA3) Total Marks 25

Write the correct choice (Answer any 5) **1 X 5 = 5**

1. Parse tree is generated in the phase of
    a. Syntax Analysis
    b. Semantic Analysis
    c. Code Optimization
    d. Intermediate Code Generation
2. FIRST(αβ) is
    a. FIRST(α)
    b. FIRST(α)U FIRST(β)
    c. FIRST(α)U FIRST(β) if FIRST(α) contains ε else FIRST(α)
    d. None of these
3. Left factoring guarantees
    a. Not occurring of backtracking
    b. Cycle free parse tree
    c. Error free target code
    d. Correct LL(1) parsing table
4. YACC builds up
    a. SLR parsing table
    b. LALR parsing table
    c. Canonical LR parsing table
    d. None of these
5. The regular expression $(a|b)^*abb$ denotes
    a. All possible combinations of a's and b's
    b. Set of all strings ending with abb
    c. Set of all strings starting with a and ending with abb
    d. None of these
6. An annotated parse tree is a parse tree
    a. With values of only some attributes shown at parse tree nodes
    b. With attribute values shown at the parse tree node
    c. Without attribute values shown at the parse tree nodes
    d. With grammar symbols shown at the parse tree nodes
7. An intermediate code form is _____
    a. Postfix Notation
    b. Syntax Trees
    c. Three address code
    d. All of the mentioned

Answer the following (Answer any 4) **4 X 5 = 20**

8. What is a 'handle'? Consider the grammar E → E + n | E x n |n. For a sentence n + n x n, write the handles in the right-sentential forms of the reduction. What is predictive parsing? [5]

9. Draw the Syntax Tree and generate the postfix notation of the expression a := b * - c + b * - c [5]

10. Draw the annotated parse tree of real id1, id2, id3 for the following grammar [5]

    D ® T L
    T ® int
    T ® real
    L ® L_1 , id
    L ® id

11. Design the dependency graph for the following grammar: [5]

    S → T List
    T → int
    T → float
    T → char
    T → double
    List → $List_1$, id
    List → id

12. Write the quadruple, triple, indirect triple for the statement [5]

    a: = b * - c + b * - c

13. Draw a DAG for expression a + a * (b – c) + (b – c) * d [5]