Register No.: ..... Name: .....

SAINTGITS COLLEGE OF ENGINEERING (AUTONOMOUS)

(AFFILIATED TO APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY, THIRUVANANTHAPURAM)

SIXTH SEMESTER B.TECH DEGREE EXAMINATION (R,S), MAY 2024
COMPUTER SCIENCE AND ENGINEERING
(2020 SCHEME)

Course Code : 20CST302

Course Name: Compiler Design

Max. Marks : 100 **Duration: 3 Hours**

PART A

(Answer all questions. Each question carries 3 marks)

1. Define tokens, lexemes, and patterns. Give example.
2. Apply Bootstrapping to develop a compiler for a new high-level language P on machine N.
3. Is the grammar **S→S(S)S/ ε** ambiguous. Justify your answer.
4. Eliminate left recursion from the following grammar:
   E → E + T / T
   T → T * F / F
   F → id
5. Determine the FIRST and FOLLOW sets for the non-terminals in the following grammar:
   S → AaAb / BbBa
   A → ε
   B→ ε
6. List out the conflicts that could arise during shift-reduce parsing?
7. What are annotated parse trees? Give examples.
8. Distinguish between S-attributed definitions and L-attributed definitions.
9. Interpret Loop Optimization techniques with suitable examples.
10. Generate a code sequence for the assignment statement
    d=(a-b)+(a-c)+(a-c)

PART B

(Answer one full question from each module, each question carries 14 mark

MODULE I

11. a) Explain the various phases of a compiler with a neat diagram. Illustrate the output of each phase for the input **x = 2 * a + b**, where **a** and **b** are float variables. (7)
    b) Write a short note on compiler construction tools. (7)

OR

12. a) Describe the input buffering scheme in a lexical analyser. (7)

b) Draw a Transition Diagram for the token relation operators and identifiers. (7)

relop➔< | <= | = | <> | >= | >

MODULE II

13. a) Explain Left recursive grammar with an example. What are the steps in removing left recursion? Differentiate leftmost derivation and rightmost derivation with examples for each. (7)

b) Design a recursive descent parser for the grammar (7)

S→ cAd

A→ ab/ b

OR

14. a) Construct the predictive parsing table for the following grammar: (7)

S→ (L)|a

L→ L,S|S

b) Check if the following grammar is LL(1) by constructing a parse table: (7)

S → i E t S S' | a

S' → eS | ε

E → b

MODULE III

15. a) Explain operator grammar and operator precedence parsing (7)

b) Create an LALR (1) parsing table for the following grammar (7)

S → AA

A → aA

A → b

OR

16. a) (7)

What are the 4 possible actions of shift reduce parsing?

Consider the following Grammar

S → CC

C → cC

C → d

Check whether input string "ccdd" is accepted or not accepted using Shift-Reduce parsing.

b) Construct a canonical LR(0) collection of items for the grammar below. (7)

S➔ L=R

S➔R

L ➔ *R

L ➔ id

R ➔ L

Also, identify a shift-reduce conflict in the LR(0) collection constructed above.

MODULE IV

17. a) Explain quadruples, triples and indirect triples with examples. (7)
    b) Describe storage organization and storage allocation strategies. (7)

OR

18. a) Construct the DAG and three address code for the following expression (7)
    a+a*(b-c)+(b-c)*d
    b) With an SDD for a desk calculator, give the appropriate code to be executed at each reduction in the LR parser designed for the calculator. (7)
    Also, give the annotated parse tree for the expression **(3 * 5) - 2**

MODULE V

19. a) Summarize the principal sources of optimization. (7)
    b) Outline the issues in the design of a code generator. (7)

OR

20. a) Write briefly about the optimization of basic blocks. (7)
    b) Consider a simple code generation algorithm. Discuss its briefly. (7)

****