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

SAINTGITS COLLEGE OF ENGINEERING (AUTONOMOUS)

(AFFILIATED TO APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY, THIRUVANANTHAPURAM)

SIXTH SEMESTER B.TECH DEGREE EXAMINATION (R), MAY 2023
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. Differentiate analysis and synthesis phase of a compiler.
2. Discuss the relevance of symbol table in compilation process.
3. Write the steps to remove left recursion.
   Consider the following grammar and eliminate left recursion.
   A → ABd / Aa / a
   B → Be / b
4. Check whether the given grammar G is ambiguous or not.
   A → AA
   A → (A)
   A → a
5. Demonstrate the identification of handles in operator precedence parsing.
6. Calculate the FIRST and FOLLOW functions for the given grammar-
   S → (L) / a
   L → SL'
   L' → ,SL' / ∈
7. Define S-attributed and L- attributed definition. Give an example for each.
8. What is the role of activation record in compiler design?
9. Write the algorithm for partitioning a sequence of three address instruction into basic blocks.
10. List any three issues in the design of code optimization.

PART B

(Answer one full question from each module, each question carries 14 marks)

MODULE I

11. a) Explain the role of transition diagrams in recognition of tokens. (8)
    b) Discuss different compiler construction tools. (6)

OR

12. a) What are the different phases of compiler? Explain the phases in detail. Write down the output of each phase for the expression a:=b+c*50. (10)

b) Explain bootstrapping with an example (4)

MODULE II

13. a) Write the algorithm for recursive descent parser to implement the following Grammar. (7)

$E \rightarrow TE'$

$E' \rightarrow +TE'$

$T \rightarrow FT'$

$T' \rightarrow * FT' \mid \varepsilon$

$F \rightarrow (E) \mid id$

b) Given a grammar

$E \rightarrow EE+$

$E \rightarrow E(E)$

$E \rightarrow id$

Given the parse tree for the string id(id)id +. Differentiate leftmost derivation and rightmost derivation. (7)

OR

14. a) Write Non-recursive predictive parsing algorithm (6)

b) Prove that the following grammar is not LL(1) (8)

$S \rightarrow iEtSS' \mid a$

$S' \rightarrow eS \mid \varepsilon$

$E \rightarrow b$

MODULE III

15. a) Construct the SLR Parsing table for the following grammar. (10)

$E \rightarrow E + T \mid T$

$T \rightarrow TF \mid F$

$F \rightarrow F^* \mid a \mid b$

b) Write all moves by the LR parser for parsing the input a * b + a [use the parsing table created in question number 15.a] (4)

OR

16. a) Consider the grammar (7)

$S \rightarrow (L) \mid a$

$L \rightarrow L , S \mid S$

For the string (a , (a , a)) show the actions of a shift reduce parser. Clearly indicate the stack and input configurations at each step.

b) Construct a CLR parsing table for the given context-free grammar (7)

$S \rightarrow AA$

$A \rightarrow aA \mid b$

MODULE IV

17. a) What is SDD? Write the SDD for a type declaration and draw the annotated parse tree for the declaration float id1,id2,id3 (7)

b) Explain static allocation and heap allocation strategies. (7)

OR

18. a) Write the SDD for a desk calculator, write the steps involved in the bottom up evaluation for the expression (3+4)*(5+6)n (6)

b) Construct Quadruples, Triples, and Indirect Triples for the expression -(a + b) * (c + d) - (a + b + c) (8)

MODULE V

19. a) Write the code generation algorithm. (6)

b) Explain Optimization of basic blocks. (8)

OR

20. a) Translate the expression d : = (a-b) + (a-c) + (a-c) into the address code sequence and then generate the machine code for the three address code. (10)

b) With an example explain the following loop optimization (4)
(i)Code motion
(ii)Strength reduction

****