A 1200CST302052301 Pages: 3

Reg No.:_______________ Name:__________________________

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Sixth Semester B.Tech Degree Supplementary Examination May 2023 (2019 Scheme)

Course Code: CST302

Course Name: COMPILER DESIGN

Max. Marks: 100 Duration: 3 Hours

PART A

Answer all questions, each carries 3 marks. Marks

1 Construct a regular expression for the language that consists of all strings ending with 00 over Σ = {0,1}. (3)

2 Define tokens, lexemes and patterns with suitable examples for each. (3)

3 Given a grammar : (3)

S → (L)| a

L → L,S | S

(i) Is the grammar ambiguous? Justify.

(ii)Build a parse tree for the string (a,((a,a), (a,a))).

4 Compute the FIRST and FOLLOW for the following Grammar. (3)

S → SS | AB

A → Aa | a

B → Bb | b

5 Define an operator grammar. Give an example. (3)

6 What are viable prefixes? (3)

7 Explain quadruples, triples and indirect triples with suitable examples. (3)

8 What are L-attributed definitions and S-attributed definitions in a syntax directed translation scheme? (3)

9 Construct the syntax tree and then draw the DAG for the statement: (3)

e := (a*b) + (c-d) *(a*b)

10 Explain any three issues in the design of a code generator. (3)

PART B

Answer one full question from each module, each carries 14 marks.

Module I

11 a) Explain in detail the various phases of the compiler with a neat diagram. Illustrate the output of each phase for the input, (9)

sum := a + b * 30

where a and b are float variables.

b) Apply bootstrapping to develop a compiler for a new high level language N on machine P. (5)

OR

12 a) Explain the role of transition diagrams in recognition of tokens. Draw the transition diagram for the regular definition: (8)

relop → < | <= | = | <> | >= | >

b) List and explain any three tools that help a programmer in building a compiler efficiently. (6)

Module II

13 a) Consider the following grammar: (9)

E→ E + T | T

T→ T * F |F

F→ ~ F | (E) | id

(i) Remove left recursion from the grammar.

(ii) Construct a predictive parsing table.

(iii) Justify the statement " The grammar is LL (1)".

b) Design a recursive descent parser for the grammar: S→cAd, A→ab/ b (5)

OR

14 a) Left factor the following grammar and then obtain LL(1) parsing table. (7)

S → TL;

T → int | float

L → L , id | id

Is the grammar LL(1)? Justify your answer.

b) Write all the moves by the LL(1) parser for parsing the input "int id,id;". [Use the parsing table created in question number 14.a] (7)

Module III

15 a) Construct canonical LR(0) collection of items for the grammar below. (10)

S → L = R

S → R

L → * R

L → id

R→ L

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

b) Write an algorithm for computing the closure of an LR(0) items. (4)

OR

16 a) Construct LALR parse table for the grammar: A→ BB, B→ bB | d (10)

b) What are the different parsing conflicts in the SLR parsing table? (4)

Module IV

17 a) Write the SDD for a desk calculator and draw the annotated parse tree for the expression: $4 * 5 + 6 - (3 * 2)$ (8)

b) Explain bottom- up evaluation of s-attributed definitions. (6)

OR

18 a) Write syntax directed definition to construct syntax tree and three address code for assignment statements. (7)

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

Module V

19 a) With suitable examples explain the following loop optimization techniques: (i) Code motion (ii) Induction variable elimination and (iii) strength reduction (7)

b) Explain the optimization of basic blocks. (7)

OR

20 a) For the following C statement, write the three-address code and quadruples. (8)

$$S := A - B + C * D - E + F$$

Also convert the three-address code into machine code.

b) Write the Code Generation Algorithm and explain the *getreg* function. (6)

****