

Reg No.: _____

Name: _____

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY
SIXTH SEMESTER B.TECH DEGREE EXAMINATION(S), DECEMBER 2019

Course Code: CS304

Course Name: COMPILER DESIGN

Max. Marks: 100

Duration: 3 Hours

PART A

Answer all questions, each carries 3 marks.

Marks

- | | | |
|---|---|-----|
| 1 | Scanning of source code in compilers can be speeded up using input buffering. Explain. | (3) |
| 2 | Draw the DFA for the regular expression $(a b)^* (abb a+ b)$. | (3) |
| 3 | Differentiate leftmost derivation and rightmost derivation. Show an example for each. | (3) |
| 4 | Find out context free language for the grammar given below:
$S \rightarrow abB$
$A \rightarrow aaBb \epsilon$
$B \rightarrow bbAa$ | (3) |

PART B

Answer any two full questions, each carries 9 marks.

- | | | |
|---|--|-----|
| 5 | a) Explain compiler writing tools. | (5) |
| | b) Given a grammar :
$S \rightarrow (L)a$
$L \rightarrow L,S S$
(i) Is the grammar ambiguous? Justify
(ii) Give the parse tree for the string $(a,((a,a), (a,a)))$ | (4) |
| 6 | a) Construct the predictive parsing table for the following grammar:
$S \rightarrow (L) a$
$L \rightarrow L,S S$ | (5) |
| | b) Explain how the regular expressions and finite state automata are used for the specification and recognition of tokens? | (4) |
| 7 | a) Explain the working of different phases of a compiler. Illustrate with a source language statement. | (5) |
| | b) Can recursive descent parsers used for left recursive grammars? Justify your answer. Give the steps in elimination of left recursion in a grammar. | (4) |

PART C*Answer all questions, each carries 3 marks.*

- 8 Compute FIRST and FOLLOW for the grammar: (3)
 $S \rightarrow SS+ \mid SS^* \mid a$
- 9 Write the algorithm to construct LR(1) collection for a grammar. (3)
- 10 What is an SDD? Show an example. (3)
- 11 Distinguish between synthesized and inherited attributes. (3)

PART D*Answer any two full questions, each carries 9 marks.*

- 12 a) Write algorithm for SLR parsing table construction. (5)
 b) Construct syntax directed translation scheme for infix to postfix translation. (4)
- 13 a) Construct the SLR table for the grammar: (5)
 $S \rightarrow aSbS \mid a$
 b) Give the annotated parse tree for the expression: $1*2*3*(4+5)$ n (4)
- 14 a) Differentiate CLR and LALR parsers. (4)
 b) Explain the specification of a simple type checker. (5)

PART E*Answer any four full questions, each carries 10 marks.*

- 15 a) Explain how DAGs help in intermediate code generation? (4)
 b) Explain the code generation algorithm. Illustrate with an example. (6)
- 16 a) Define the following and show an example for each. (6)
 i). Three-address code iii). Triples
 ii). Quadruples iv). Indirect triples
 b) State the issues in design of a code generator. (4)
- 17 a) Explain different stack allocation strategies with suitable examples. (10)
- 18 a) Explain different code optimization techniques available in local and global optimizations? (10)
- 19 a) How is storage organization and management done during runtime? (4)
 b) How the optimization of basic blocks is done by a compiler? (6)
- 20 a) Write the algorithm for partitioning a sequence of three-address instructions into basic blocks. (4)
 b) Construct the DAG and three address code for the expression $a+a*(b-c)+(b-c)*d$ (6)