

A

1200CST302122401

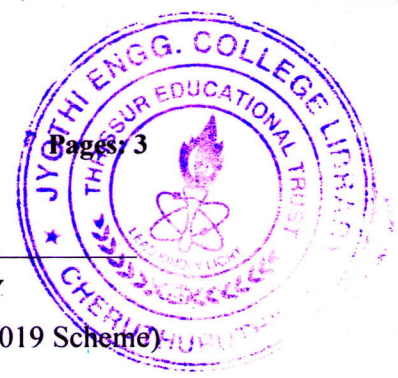
Pages: 3

Reg No.: _____

Name: _____

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

B.Tech Degree S6 (S, FE) / S6 (PT) (S,FE) Examination December 2024 (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 | Distinguish between front end and back end of a compiler. | (3) |
| 2 | What are the different types of errors detected by compiler? | (3) |
| 3 | Differentiate between syntax tree and parse tree with an example. | (3) |
| 4 | Show that the grammar $S \rightarrow 0S1 \mid SS \mid \epsilon$ is ambiguous or not. | (3) |
| 5 | Define an operator grammar. Give an example. | (3) |
| 6 | What are the possible actions of a shift-reduce parser? | (3) |
| 7 | Draw syntax tree and DAG representation for the given statement.
$s = (a * b) * (b + c) + (a * b)$ | (3) |
| 8 | What are L-attributed definitions and S-attributed definitions in a syntax directed translation scheme? | (3) |
| 9 | Differentiate local and global optimization in compiler design. | (3) |
| 10 | Write the algorithm for partitioning a sequence of three-address instructions into basic blocks. | (3) |

PART B*Answer one full question from each module, each carries 14 marks.***Module I**

- 11 a) Explain the functions of different phases of compiler with a neat diagram. (9)
Illustrate the output of each phase for the input: $a = b + c - 10$.
- b) Define lexeme, token and pattern. Identify the lexemes that make up the tokens in the following program segment. Indicate corresponding token. (5)

```
void swap (int i, int j)
```

```
{
```

```
    int t;
```

```
    t=i;
```

```
    i=j;
```

```
    j=t;
```

```
}
```

OR

- 12 a) Explain any four different compiler construction tools. (8)
 b) What is bootstrapping in compiler development? Explain with an example. (6)

Module II

- 13 a) Write the algorithm and eliminate left recursion from the following grammar (8)

$$S \rightarrow a^i | (T)$$

$$T \rightarrow T, S | S$$
 b) Explain recursive descent parser with an example. (6)

OR

- 14 a) Find the FIRST and FOLLOW of the grammar (7)

$$S \rightarrow ACB / CbB / Ba$$

$$A \rightarrow da / BC$$

$$B \rightarrow g / \epsilon$$

$$C \rightarrow h / \epsilon$$
 b) Construct predictive parsing table for the grammar (7)

$$S \rightarrow (L) | a$$

$$L \rightarrow SL'$$

$$L' \rightarrow) SL' | \epsilon$$
 Check whether the given grammar is LL (1) or not. Justify your answer.

Module III

- 15 a) Construct Canonical LR(1) parsing table for the grammar (8)

$$S \rightarrow CC$$

$$C \rightarrow aC$$

$$C \rightarrow d$$
 b) Illustrate the actions of the operator precedence parser for the input (6)

$$id1 + id2 * id3$$
 based on the given grammar

$$E \rightarrow E + E / E * E / (E) / id$$

OR

- 16 a) Explain the term handle and handle pruning with an example. (6)
 b) What is a shift-reduce parser? Explain the conflicts that may occur during shift-reduce parsing. (8)

Module IV

- 17 a) Write the syntax directed translation scheme for a simple desk calculator. Give an annotated parse tree for the input string $23 * 5 + 4$. (8)
 b) Explain static allocation and heap allocation strategies. (6)

OR

- 18 a) Write the quadruple, triple, indirect triple for the expression (8)

$$(a * b) + (c + d) - (a + b + c + d)$$

- b) Generate the intermediate code for the following code segment. (6)
- ```
if (a>b)
x= a + b
else
x= a - b
```

**Module V**

- 19 a) With suitable examples, explain the following loop optimization techniques: (6)  
(i) Code motion (ii) Induction variable elimination and (iii) Strength reduction
- b) What is Peephole optimization? Explain any three transformations done in peephole optimization. (8)

**OR**

- 20 a) Explain any three issues in the design of a code generator (6)
- b) Translate the expression  $W := (A-B) + (A-C) + (A-C)$  into three address code and generate the machine code for the three address code. (8)

\*\*\*\*