ii. List some compiler construction tools. (4 Marks)

(OR)

b. Construct minimum state DFA for the regular expression $(a|b)^*a(a|b)$.

29. a. Show that the grammar is LR(1) but not LALR

$S \rightarrow Aa \mid bAc \mid Bc \mid bBa$

$A \rightarrow d$

$B \rightarrow d$

(OR)

b. Construct LALR parsing table for the grammar

$S \rightarrow AA$

$A \rightarrow aA \mid b$

30. a.i. Write down the translation scheme to generate code for assignment statement. Use the scheme for generating three address code for the assignment statement $g = a + b - c * d$. (8 Marks)

ii. Define quadruple. Give an example. Why are quadruples preferred over triples in an optimizing compiler? (4 Marks)

(OR)

b.i. Generate intermediate code for the following code segment along with the required syntax-directed translation scheme

```
while(i < 10)
if ((i%2) == 0)
    even = even + i;
else
    odd = odd + i;
```

(8 Marks)

ii. What are the types of three address statements? (4 Marks)

31. a.i. Write in detail about the issues in the design of a code generator. (8 Marks)

ii. What is meant by address descriptors and register descriptors? (4 Marks)

(OR)

b. Generate the code for the following expression using the code generator algorithm $s = (a-b)+(a-c)+(a-c)$.

32. a.i. Describe in detail about optimization of basic block with example.

ii. Explain in detail about various methods of passing parameters.

(OR)

b. Explain in detail about principal sources of optimization.

* * * * *

Reg. No.															

B.Tech. DEGREE EXAMINATION, MAY 2019
3rd to 8th Semester

15CS314J – COMPILER DESIGN
(For the candidates admitted during the academic year 2015 – 2016 to 2017 – 2018)

Note:
(i) **Part - A** should be answered in OMR sheet within first 45 minutes and OMR sheet should be handed over to hall invigilator at the end of 45th minute.
(ii) **Part - B** and **Part - C** should be answered in answer booklet.

Time: Three Hours Max. Marks: 100

PART – A (20 × 1 = 20 Marks)
Answer ALL Questions

1. A compiler for a high level language that runs on one machine and produce code for different machines is called
(A) Optimizing compiler (B) One pass compiler
(C) Cross compiler (D) Multipass compiler

2. The outcome of the *lex* utility is
(A) *lex.er* (B) *lex.yy.c*
(C) *lex.ll.c* (D) *lex.yy.cc*

3. Which of the following regular expression operators has the least precedence
(A) Concatenation (B) Kleene closure
(C) Alternation (D) Positive closure

4. If L_1 is represented by $(a|b|c)^*de$ and L_2 is represented by $(0|1|2)^*34$ then choose the right string that is generated by L_1L_2
(A) *ae*234 (B) *abcccde*0011234
(C) *abcccde*1232 (D) *de*223

5. The grammar $A \rightarrow AA \mid (A) \mid \in$ is not suitable for predictive passing because the grammar is
(A) Ambiguous (B) Left-recursive
(C) Right-recursive (D) An operator-grammar

6. Consider the grammar shown below $S \rightarrow iEtSS' \mid a$, $S' \rightarrow eS \mid \in$, $E \rightarrow b$ in the predictive parse table M of this grammar, the entries $M[S',e]$ and $M[S' \rightarrow S]$ respectively are
(A) $\{S' \rightarrow eS\}$ *and* $\{S' \rightarrow e\}$ (B) $\{S' \rightarrow eS\}$ *and* $\{\ \}$
(C) $\{S' \rightarrow \in\}$ *and* $\{S' \rightarrow \in\}$ (D) $\{S' \rightarrow eS, S' \rightarrow \in\}$ *and* $\{S' \rightarrow \in\}$

7. Consider the grammar S→(S)|a. Let the number of states in SLR(1), LR(1) and LALR(1) parsers for the grammar be n_1, n_2 and n_3 respectively. The following relationship holds good
(A) $n_1 < n_2 < n_3$ (B) $n_1 = n_3 < n_2$
(C) $n_1 = n_2 = n_3$ (D) $n_1 \geq n_3 \geq n_2$

8. An LALR(1) parser for a grammar G can have shift-reduce (S-R) conflict if and only if
   (A) The SLR(1) parser for G has S-R conflict
   (B) The LR(1) parser for G has S-R conflict
   (C) The LR(0) parser for G has S-R conflict
   (D) The LALR(1) parser for G has reduce-reduce conflict

9. Semantic routines are used to
   (A) Check the token formation
   (B) Create intermediate code
   (C) Check the syntax of the sentence
   (D) Create machine code

10. Synthesized attributes of a node in the parse are computed
   (A) From the attributes of the left sibling
   (B) From the attributes of the right sibling
   (C) From the attributes of the root node
   (D) From the attributes of its children

11. Consider the translation scheme shown below
   S→TR
   R→T {print '+':} R/ϵ
   T→num {print (num.val)}
   Here num is a token that represents an integer and num.val represents the corresponding integer value. For an I/P string 9+5+2, this translation scheme will print
   (A) 9+5+2
   (B) 95+2+
   (C) 952++
   (D) ++952

12. Consider the grammar with the following translation rules and E as the start symbol. Compute E-value for the root of the parse tree for the expression 2≠3&5≠6&4
   E→E1≠T {E.value = E1.value T.value}
   |T {E.value = T.value}
   T→T1 LF {T.value = T1.value+F.value}
   |F {T.value = F.value}
   F→num {F.value = num.value}
   (A) 200
   (B) 180
   (C) 160
   (D) 40

13. Which code is faster?
   (A) MOV R_0, a
   (B) MOV R_0, R_1
   (C) MOV a, R_0
   (D) MOV R_1, a

14. Back patching
   (A) is applicable only for quadruples
   (B) is applicable only for machine code
   (C) is applicable for both quadruples and machine
   (D) is not applicable to quadruples and machine code

15. The graph that shoes basic blocks and their successor relationship is called
   (A) DAG
   (B) Flow graph
   (C) Control graph
   (D) Hamiltonion graph

16. The identification of common subexpressions and replacement of run-time computations by compile time computation is
   (A) Local optimization
   (B) Loop optimization
   (C) Constant folding
   (D) Data flow analysis

17. Local and loop optimization in turn provide motivation for
   (A) Data flow analysis
   (B) Constant folding
   (C) Peephole optimization
   (D) DFA and constant folding

18. Reduction in strength means
   (A) Replacing runtime computation by compile time computation
   (B) Removing loop invariant computation
   (C) Removing common sub expression
   (D) Replacing a costly operation by a relatively cheaper one

19. Identifying the common sub expressions will enable
   (A) The increase in code space
   (B) The code to run slowly
   (C) The reduction of code space
   (D) Change of meaning of the common expression

20. Why is the code optimizations are carried out on the intermediate code?
   (A) Because for optimization information from the front end cannot be used
   (B) Because program is more accurately analyzed on intermediate code than on machine code
   (C) Because for optimization information from data flow analysis cannot be used
   (D) Because they enhance the portability of the compiler to other target processor

PART – B (5 × 4 = 20 Marks)
Answer **ANY FIVE** Questions

21. List out the various error recovery strategies for a lexical analysis.

22. Consider the grammar $S \rightarrow A \mid B \quad A \rightarrow 0A \mid \in \quad B \rightarrow 0B \mid 1B \mid \in$. Find the left most derivation, right most derivation and parse tree for the following strings (i) 00101 (ii) 1001.

23. Find the shift reduce parsing algorithm for input string (a, a)
   $S \rightarrow (L) \mid a$
   $L \rightarrow L,S \mid S$

24. For the given grammar compute first and follow set
   $S \rightarrow A\varepsilon B \mid bA \mid \in$
   $A \rightarrow aAb \mid \in$
   $B \rightarrow bB \mid \in$

25. Write the three address code for the statements $a = b * -c + b * -c$ and draw the syntax tree.

26. Construct DAG for $a + a * (b - c) + (b - c) * d$.

27. Define common sub-expression elimination with example.

PART – C (5 × 12 = 60 Marks)
Answer **ALL** Questions

28. a.i. Explain in detail about the phases of compiler and translate the stamen pos= init + rate *60. (8 Marks)