THEORY OF COMPUTATION AND COMPILER DESIGN TEST 3

Number of Questions: 35 **Section Marks: 30**

Directions for questions 1 to 35: Select the correct alternative from the given choices.

1. Which of the following is not a feature of 'Deterministic Finite Automata'?
(A) Finite set of states
(B) Finite set of input symbols
(C) Any number of start states
(D) A set of final states

2. Which language is accepted by an NFA?
(A) Regular languages
(B) Superset of Regular languages.
(C) Proper subset of Regular languages.
(D) Context free languages

3. Which of the following has the power to be in several states at once?
(i) DFA (ii) NFA
(A) (i) only (B) (ii) only
(C) Both (i) and (ii) (D) Neither (i) nor (ii)

4. In which of the following terms DFA and NFA will differ?
(i) Set of states (ii) Set of inputs
(iii) Start state (iv) Set of final states
(v) Transition function
(A) (iii), (iv) (B) (iii), (iv), (v)
(C) (i), (iii), (v) (D) (v) only

5. Which of the following is FALSE for a transition function of NFA?
(A) It takes a state from a finite set of states as an argument.
(B) It takes an input symbol from a finite set of inputs as an argument.
(C) It returns a subset of states.
(D) It returns a state.

6. Which of the following is FALSE?
(A) A language L is accepted by some DFA if L is accepted by some NFA.
(B) An NFA with n states will have an equivalent DFA with 2^n states.
(C) NFA and DFA have equal expressing capability.
(D) $\in$-NFA has more expressing capability than NFA.

7. What are the equivalent sets of ϕ^* and ϕ^0 respectively?
(A) $\{\in\}, \{\in\}$ (B) $\{\in\}, \phi$
(C) $\phi, \{\in\}$ (D) ϕ, ϕ

8. Let L be the language is defined over $L = \{0, 1\}$ then, $L^* =$
(A) $\{\in\}$
(B) $\{0, 1\}$
(C) $\{\in, 0, 1, 00, 11, 01, 10, ...\}$
(D) $\{0, 1, 00, 11, 01, 10, ...\}$

9. While scanning the input string, all the constants stored in the following Data structure?
(A) Symbol Table (B) Terminal Table
(C) Numeric Table (D) Literal Table

10. Eliminating left recursion results
(A) in converting a non LL(1) grammar to LL(1) grammar.
(B) in nonretainment of left associativity.
(C) in comparatively easy implementation
(D) may fall in to infinite loop.

11. Which one of the following parser does not require intelligence to parse string?
(A) RDP
(B) Brute Force Technique
(C) Table driven parser
(D) Operator precedence parser

12. In the Parsing of a string 'w' using LL(1) parsing algorithm, Top of stack contains a terminal and Look ahead symbol is same as Top of stack, then which of the following action is performed?
(A) successfull completion
(B) pop of stack
(C) increment input pointer
(D) pop of stack and increment input pointer

13. Which of the following statement is false about LL(1)?
(A) An ambiguous grammar is not LL(1)
(B) Left factored grammar is not LL(1).
(C) Left recursive grammar is not LL(1)
(D) In a grammar 'G', if every non-Terminal if produces only one production then G is not in LL(1).

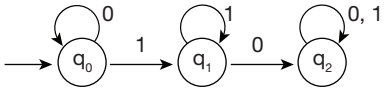
14. Which of the following statement is false?
(A) Bottom up parsing uses Reverse Right most derivation.
(B) Bottom up parsing uses canonical reduction sequence.
(C) Top down parsing uses Left most derivation.
(D) Top down parsing uses canonical Left sentential form.

15. Which of the following statement is false?
(A) CLR is most widely used parser
(B) Size of SLR(1), LALR(1) and CLR(1) parsers may or may not be equal.
(C) Go to and shift action of LR(0), SLR(1) and LALR(1) must be equal.
(D) LALR(1) parser is most widely used parser.

16. Consider the DFA which accepts $L = \{w/w$ is of the form $x10y$ for some strings x and y consisting of 0's and 1's only$\}$.
Which of the following strings are not accepted by given DFA?

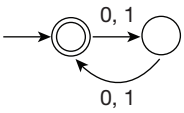
(i) 01 (ii) 11010
(iii) 100011 (iv) ∈
(A) (i), (ii) (B) (i), (ii), (iv)
(C) (i), (iv) (D) (i), (ii), (iii), (iv)

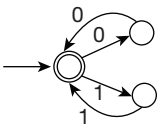
17. Consider below DFA:

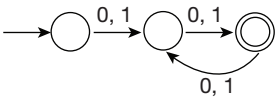


What is the language accepted by given DFA?
(A) All strings which ends with either '0' or '1' only.
(B) All strings which ends with '0' only.
(C) All strings which has the substring '10'.
(D) All strings which has the substring '110'.

18. Which of the following DFA accepts all the even length strings on {0, 1}?

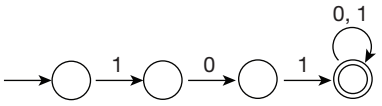
(A)


(B)


(C)


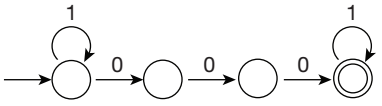
(D) All the above

19. Consider below DFA:



What is the language accepted by above DFA?
(A) All the strings which contain '101' as substring.
(B) All the strings which begin with '101'.
(C) All the strings which ends with either '0' or '1'.
(D) All the strings which do not contain the substring '00'.

20. Which of the following regular expression represents below DFA?



(A) 1*0001* (B) 1* + 000 + 1*
(C) 1* + 0 + 0 + 0 + 1* (D) (10001)*

21. What is the language accepted by the transition table of a DFA given below:

δ	A	B
Q_0	Q_1	Q_0
Q_1	Q_0	Q_1

(A) All the strings which do not have even length.
(B) All the strings which have at least one a.
(C) All the strings with even numbers of a's.
(D) All the strings with odd number of a's.

22. Consider below transition table of a DFA with some blanks:

δ	0	1
q_0	–	q_0
q_1	–	–
q_2	q_2	q_2

What are the missing transitions if the DFA accepts all the strings with '00' as substring?
(A) $q_0 \xrightarrow{0} q_1, q_1 \xrightarrow{0} q_2, q_1 \xrightarrow{0} q_0$
(B) $q_0 \xrightarrow{0} q_1, q_1 \xrightarrow{1} q_2$
(C) $q_0 \xrightarrow{0} q_1, q_1 \xrightarrow{0} q_2, q_0 \xrightarrow{1} q_0$
(D) $q_0 \xrightarrow{0} q_1, q_1 \xrightarrow{0} q_2$

23. Let R and S be two regular expressions then which of the following is not a regular expression?
(A) $R + S$ (B) RS
(C) R^* (D) None of the above

24. What is the regular expression for the language over {0, 1}, which accepts the set of all strings that begin with '110'?
(A) 110 (10)* (B) 110 (01)*
(C) 110 (1 + 0)* (D) $110^+ (1 + 0)^*$

25. Give a regular expression for the language over {0} which accepts the set of all strings of odd number of 0's?
(A) 0* (B) 0^+
(C) 0(00)* (D) 0 + (00)*

26. Lexical Analyzer uses which one of the following pattern (Lexer) to generate tokens?
(A) Regular expression + Priorities + Longest Matching Token rule.
(B) Regular Expression + priorities + Shortest Matching token rule.
(C) Regular Expression + longest Matching Token rule
(D) Regular Expression + Shortest Matching token rule.

27. Consider the following code segment. If C compiler compiles the code, what will be the response?

```
#include <stdio.h>
main( )
{
/* this is My first program*/
int a, b, c;
/* initializing variables
/* a = 10, b = 20 */ compute c value
   */
c = a + b;
printf("%d", c);
}
```

(A) No error, produces object code.
(B) Run time error but No compile time error.
(C) syntax error but No Lexical error.
(D) None of the above.

28. Find the Number of Tokens in following C code

```
main ( )
{
int a, b;
a = 1; b = 0;
if (!a! = b)
a << = 1;
else
b >> = 2;
}
```

(A) 36 (B) 38
(C) 39 (D) lexical error

29. Regular Expression for identifier is $L(L \cup N)^*$: Here L-Letter and N-number, and some keywords in a language are {int, float, main, double, ...}. Then a string "int" is given to scanner how many strings or sub strings satisfies the patterns of Lexical Analyzer?
(A) 3 (B) 4
(C) 1 (D) None of the above

30. Consider the following statements about function and Token, when it is recognized during scanning.
(i) it produces Token value.
(ii) put identifier in symbol table.
(iii) increment line number
(iv) get next line and input to scan
Which of the following is true?
(A) (i), (ii), (iv) (B) (ii) and (iv)
(C) (ii), (iii) and (iv) (D) (i), (ii), (iii), (iv)

31. Time complexity of a parser that works for any unambiguous grammar where 'n' is the length of the input is:
(A) $O(n)$ (B) $O(n^2)$
(C) $O(n^3)$ (D) $O(n \log n)$

32. While parsing a string $w = abcd$ using Bottom-up parsing, what are the possible strings or sub strings that can be considered?
(A) {a, bc, cd, d, bcd} (B) {a, ab, $abcd$, bd}
(C) {b, c, ac, bd, dc} (D) {d, dc, dcb, $dcba$}

33. Consider the following grammar:
$S \rightarrow aABe$
$A \rightarrow \frac{Abc}{b}$
$B \rightarrow d$
Which of the following is the correct sequence of handles to parse a string $w = abbcde$?
(A) {$aABc$, d, Abc, b} (B) {b, d, Abc, $aABc$}
(C) {d, b, Abc, $aABc$} (D) {b, Abc, d, $aABc$}

34. Which of the following statement is false?
(A) LALR(1) $\subseteq$ LR(1) (B) LL(1) $\subseteq$ LALR(1)
(C) LL(0) $\subseteq$ LR(0) (D) LL(K) $\subseteq$ LR(K)

35. Consider the following grammar:
$S \rightarrow Aa/b$
$A \rightarrow Bc/d/aA$
$B \rightarrow Sb/c/Bd$
Which one of the following statement is false?
(A) S has indirect Left recursion.
(B) All non terminals have indirect Left recursion.
(C) S, A have immediate Left Recursion.
(D) B has immediate and indirect left Recursion.

Answer Keys

1. C	**2.** A	**3.** B	**4.** D	**5.** D	**6.** D	**7.** A	**8.** C	**9.** D	**10.** C
11. B	**12.** D	**13.** D	**14.** D	**15.** A	**16.** C	**17.** C	**18.** A	**19.** B	**20.** A
21. D	**22.** C	**23.** D	**24.** C	**25.** C	**26.** A	**27.** D	**28.** A	**29.** B	**30.** D
31. C	**32.** A	**33.** D	**34.** B	**35.** B					

Hints and Explanations

1. A DFA will have finite states, inputs, transition function, a start state & a set of final states. Choice (C)

2. NFA's accept exactly the regular languages, same as DFA's. Choice (A)

3. NFA's have the power to be in several states at once (Because of the non determinism). Choice (B)

4. DFA & NFA will differ only in their transition function. Choice (D)

5. The transition function of NFA will take a state, an input symbol and returns a set of states. Choice (D)

6. DFA = NFA = $\in$ – NFA. Choice (D)

7. Choice (A)

8. L^* (kleen closure) of a language L represents the set of those strings that can be formed by taking any number of strings from L, possibly with repetitions and concatenating all of them. Choice (C)

9. Symbol Table mainly maintains record for identifiers, all constants and all type of literals are stored in Literal Table. Choice (D)

10. Left recursive grammars are easy to implement (compare to a grammar after eliminating Left Recursion). Choice (C)

11. Brute Force Technique 'blindly' substitutes first production; no intelligence is required. Choice (B)

12. In LL(1) parsing Algorithm '*a*' is look ahead symbol, *X* is the top of the stack, then, if $a = X \neq \$$ pop of stack and increment input pointer. Choice (D)

13. A grammar is said to be LL(1) if each and every cell contains at most one production. If every non-terminal is deriving only one production there is no chance of occurrence of two productions in one cell.

Choice (D)

14. Top-down parsers uses LMD. Bottom up parser uses Reverse RMD. Reverse Right most derivation is also called canonical reduction sequence. Choice (D)

15. CLR(1) is most powerful parser.
LALR(1) is most widely used parser as it requires Less space compared to CLR(1) Choice (A)

16. '*L*' must contain the substring '10'.
(i) is not accepted
(ii) accepted
(iii) accepted
(iv) not accepted

Choice (C)

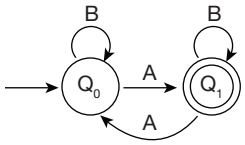
17. Given DFA accepts all the strings which have the substring '10'. The strings accepted are 10, 010, 110, ...
Choice (C)

18. The even length strings on {0, 1} are $\in$, 00, 11, 01, 10, 0011, 0101, 1100, ... only choice (A) accepts all these strings. Choice (A)

19. Given DFA accepts all the strings which begin with '101' only. It won't accept 0101; it accepts 10100.
Choice (B)

20. Given DFA accepts all the strings with exactly three consecutive zeros. Choice (A)

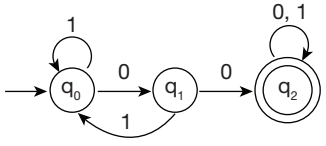
21. The DFA for given transition table is given below:



It can have any number of *b*'s but the number of *a*'s must be odd {*a*, *aaa*, *aaaaa*, ...}.

Choice (D)

22. The DFA for given partial transition table is:



If it accepts all the strings which have '00' as substring, $q_0 \to q_1$, $q_1 \to q_2$ must have '0' transition. After one '0' there is a possibility of 1's so put a '1' transition from q_1 to q_0. Choice (C)

23. $R + S$ contains union of $L(R)$ and $L(S)$. RS contains concatenation of $L(R)$ and $L(S)$. R^* is closure of R.
Choice (D)

24. The strings of the language *L* must start with '110' after that there may be any number of 0's and 1's, so the regular expression is 110 (1 + 0)*.

Choice (C)

25. 0* accepts zero or more number of 0's.
0^+ accepts one or more number of 0's.
0(00)* accepts 0, 000, 00000, ... i.e., odd number of zeros.
0 + (00)* accepts either 0 or 00 or 000, ...
Choice (C)

26. Ex : identifier L(LUT)*
int *ab*;
'*a*' is satisfying Regular Expression but it won't be treated as Lexeme. Longest matching string will be treated as Lexeme, '*ab*' is Lexeme.

Choice (A)

27. There is only Lexical Error because of Nested comments. Choice (D)

28.
```
main/ (/) /
{/int/a/,/b/;/
a/=/1/;/b/=/0/;/
if/(/!/a/! =/b/)/
a/<</=/1/;/
else/
b/>>/=/2/;/
}/
```
∴ Total 36 Tokens. Choice (A)

29. Lexical Analysis scans the input string character by char acter and checks each and every substring matching with the pattern but using lexer definition it takes longest matching string as Lexeme.
i → L(LUN)*
in → L(LUN)*
int → L(LUN)*
keywords (priority high) Choice (B)

30. Whenever a Token is generated it produces a value. If it is an identifier a record is created in symbol table. A hidden token is created for every read.

Choice (D)

31. If string length is '*n*' then any parser for unambiguous grammar takes $O(n^3)$ time. Choice (C)

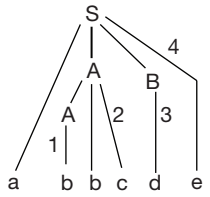
32. In bottom-up Parsing, when scanning a string for identifying handles, it considers all prefixes and suffixes of strings.

for *abcd*,

Prefixes	Suffixes
a	*d*
ab	*cd*
abc	*bcd*
abcd	*abcd*

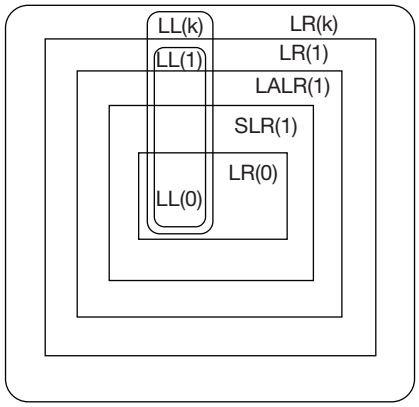
will be considered for handles Choice (A)

33.



$A \rightarrow b$
$A \rightarrow Abc$
$B \rightarrow d$
$S \rightarrow aABc$ Choice (D)

34. $LL(1) \subseteq LR(1)$



Choice (B)

35. $S \Rightarrow Aa \Rightarrow Bca \Rightarrow Sbca$
$A \Rightarrow Bc \Rightarrow Sbc \Rightarrow Aabc$
$B \Rightarrow Bd$ (immediate)
$B \Rightarrow Sb \Rightarrow Aab \Rightarrow Bcab$ (indirect) Choice (B)