CLASS TEST

S.No.:02SKCS_DFGHI_03122023

MADE EASY

India's Best Institute for IES, GATE & PSUs

Delhi | **Bhopal** | **Hyderabad** | **Jaipur** | **Pune** | **Bhubaneswar** | **Kolkata**

Web: www.madeeasy.in | **E-mail:** info@madeeasy.in | **Ph:** 011-45124612

THEORY OF COMPUTATION

COMPUTER SCIENCE & IT

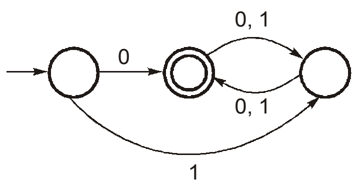
Date of Test : 03/12/2023

ANSWER KEY

1.	(c)	7.	(d)	13.	(a)	19.	(a)	25.	(d)
2.	(d)	8.	(a)	14.	(c)	20.	(d)	26.	(b)
3.	(c)	9.	(c)	15.	(d)	21.	(a)	27.	(b)
4.	(a)	10.	(c)	16.	(c)	22.	(a)	28.	(c)
5.	(c)	11.	(d)	17.	(a)	23.	(d)	29.	(a)
6.	(a)	12.	(b)	18.	(d)	24.	(d)	30.	(b)

DETAILED EXPLANATIONS

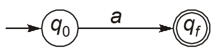
1. (c)



$$= 0((0+1)(0+1))^* + 1(0+1)((0+1)(0+1))^*$$
$$= (0+1(0+1))((0+1)(0+1))^*$$

2. (d)

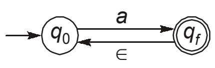
Consider the following automata *A* :



Above automata satisfies the condition as there is no transition into q_0 and no transition out of q_f.

$$L = \{a\}$$

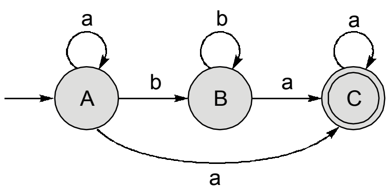
Modified Automata:



By adding this null transition in the automata. Now in this string '*a*' can occur in a string multiple times and the automata will accept it.

Hence, it accepts L^+.

3. (c)



$$\begin{aligned} \text{R.E.} &= a^*(bb^*a + a)a^* \\ &= a^*((bb^* + \varepsilon)a)a^* \\ &= a^*b^*aa^* \\ &= a^*b^*a^*a \end{aligned}$$

4. (a)

L_2 is even palindrome on $\{a, b\}^*$

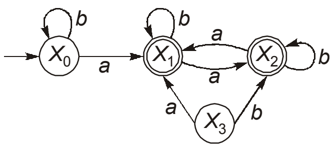
L_3 is odd palindrome on $\{a, b\}^*$

L_1 is any palindrome on $\{a, b\}^*$

Clearly, $L_2 \subset L_1$ and $L_3 \subset L_1$ and $L_1 = L_2 \cup L_3$

5. (c)

Converting the given transition to a state diagram, we get

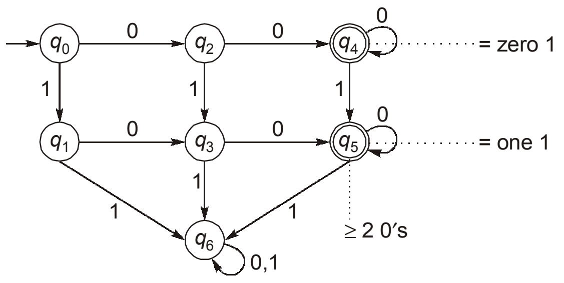


From the diagram, the expression can be written as: $X_0 = b^*a(a+b)^*$.

6. (a)

- Let $L_1 = \varepsilon$, $L_2 = \{ab\}$, $L_3 = \{a^2b^2\}$, $L_4 = \{a^4b^4\}$ ... so on.
  $L = L_1 \cup L_2 \cup L_3 \dots L_n$
  $= \{\varepsilon\} \cup \{ab\} \cup \{a^2b^2\} \cup \dots$
  $= a^n\, b^n$ that is CFL
- Let L be ϕ which is regular language.
  $L^* = \phi^* = \varepsilon$, that is regular but finite.

7. (d)



8. (a)
When grammar is in CNF i.e., when the production are of the form S → AB, S → a.
Then, length of every derivation is (2n – 1).
When grammar is in GNF i.e., when the production are of the form S → VT*. Then length of every derivation is n.
Hence, option (a) is correct.

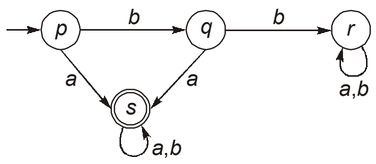
9. (c)
Traversing the states of the turing machine, it can be seen that for every 'a' as the input, it is accepting 3 b's. For every 'a' machine writes 'X' on the tape, then take right moves till it reaches 'b'. For every 3 b's it writes symbol Y.
Hence accepting the language $L = \{a^m b^n \mid 3m = n;\ m, n \geq 0\}$.

10. (c)

- $L_1 \cdot L_2 \cdot L_3 = \{0^p\, 1^q\, 2^r \mid p, q, r \geq 0\} = 0^*\ 1^*\ 2^*$ is regular.
  Since regular, hence context-sensitive too.
- $L_1 \cdot L_2 \cdot L_3 \neq \{0^n\, 1^n\, 2^n \mid n \geq 0\}$
- Since it is a regular language and regular languages are closed under complement, hence complement of $L_1 \cdot L_2 \cdot L_3$ is regular language.

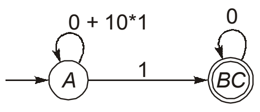
11. (d)

- A is a DFA. So, $L(A)$ is regular. Complement of $L(A)$ is also regular since regular languages are closed under complement.
- $L(A) = ((a+ba)\,(a+b)^*)$ or can be written as $(a\,(a+b)^* + ba\,(a+b)^*)$.
- The minimal DFA will be.



- '*bba*' ends with '*a*' but is not accepted by the above automata.

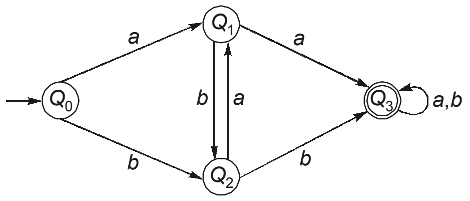
12. (b)



$= (0 + 10^*1)^*\ 10^*$

13. (a)

Set of strings that contain either 'aa' or 'bb' as the substring.



So, the regular expression for the above DFA is:

$$R = (a+b)^*\ (aa+bb)\ (a+b)^*$$

14. (c)

$$\begin{aligned} L_1/L_2 &= bba^*baa^* \,/\, ab^* \\ &= bba^*baa^* \,/\, a \\ &= bba^*baa^* \end{aligned}$$

15. (d)

$Q_0 \to Q_1$: Pushes '0' to the stack when stack initially contain \$.
$Q_1 \to Q_1$: Pushes all '0' of the string to the stack.
$Q_1 \to Q_2$: Pushes '1' to the stack, when stack initially contains 0.
$Q_2 \to Q_3$: Pop '1' from the stack, when string has 0.
$Q_3 \to Q_3$: Pop all 1's from the stack, till the string symbol is 0.
$Q_3 \to Q_4$: Pop 0's from the stack, when string actually has 1.
$Q_4 \to Q_4$: Pop all the 0's from the stack, till the string symbol is 1.
$Q_4 \to Q_5$: Stack contains \$ and string symbol is $\in$, means the string is accepted.

Hence, $L = \{0^m 1^n 0^n 1^m \mid n, m > 0\}$.

16. (c)

$$L = \{a^m b^n b^k d^l \mid (n+k = \text{odd}) \text{ only if } m = l\}$$

If, we check the condition carefully, the condition is actually logical implication.

$$L = \{a^m b^n b^k d^l \mid (n+k = \text{odd}) \to m = l\}$$

Either $n+k$ will be odd or it will be even, if $(n+k)$ is odd, then it's necessary that m should be equal to l, if $(n+k)$ is even then l can be any number.

$$\begin{aligned} L &= \{a^m\, b^{2n+1}\, d^l \text{ and } l = m\} \text{ or } \{a^m\, b^{2n}\, d^l\} \\ &= \{a^m\, b^{2n+1}\, d^m\} \cup \{a^m\, b^{2n}\, d^l\} \end{aligned}$$

$\Rightarrow$ $\qquad$ DCFL $\cup$ regular $=$ DCFL.

17. (a)

- L_1 is regular , since we can create DFA for given language.
- L_2 is CFL, since their is a comparison between number of a and number of b in strings i.e., difference is less than equal to 10.
- L_3 is regular, since by making $w = \in$ and $c = (a+b)^*$ we get language $(a+b)^*$ which contain every string belongs to $wcww^r$.

18. (d)

By pumping lemma, we can never say that a language is regular or CFL. It can only be used to prove that a certain language is not regular or not CFL.

Since pumping lemma isn't satisfied for regular, hence we can say it is not regular, but since the lemma is satisfied for context-free, we can't say that the language is CFL.

19. (a)

The language $\{0^p 1^{2p}\}$ is context-free language, hence it is recursive also. Since $L(M) \leq_p$ REC, so $L(M)$ also recursive, now given input (i.e. recursive language) to turing machine and finding it is accept or not is non-trival property so it is undecidable by Rice's theorem.

20. (d)

The given grammar G:

$$E \to 0XE2$$
$$E \to 0X2$$
$$X0 \to 0X$$
$$X2 \to 12$$
$$X1 \to 11$$

$L(G) = \{012, 001122, \dots\}$

$E \to 0X2 \Rightarrow 012$

$E \to 0XE2 \Rightarrow 0X0X22 \Rightarrow 0X0122 \Rightarrow 00X122 \Rightarrow 001122$

$\therefore \; L = \{0^n 1^n 2^n \mid n > 0\}$

21. (a)

The relationship between the number of leading a's and trailing d's in the language indicates that recursive rule is needed to generate them. The same is true for b's and c's. Derivations in the grammar

$$S \to aSdd \mid A$$
$$A \to bAc \mid bc$$

generate strings in an outside-to-inside manner.

Option (b) does generate $\in$ which is not present in $L(G)$.

Option (c) does generate "*add*" which is not present in $L(G)$.

Option (d) does not generate "*abbccdd*" which actually present in $L(G)$.

Option (a) contain all strings present in $L(G)$.

22. (a)

For an input whose length is less than 100 on which turing machine halt or not can be decidable by running turing machine for 100 steps. So, it is decidable language.

Whether a turing machine accept a given language is non-trival question so by Rice's theorem it is undecidable problem.

23. (d)

Considering each option once,

(a)
$$\begin{array}{ccccc} 0 & 0 & 0 & 1^* & (0+1) \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 0 & 0 & 0 & 1 & 0, \end{array}$$ hence its a member.

(b) $0\,(0+1)^*\,(0+1)$
$$\begin{array}{ccccc} 0 & (0+1) & (0+1) & (0+1) & (0+1) \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 0 & 0 & 0 & 1 & 0, \end{array}$$ hence its a member.

(c) $(11 + 00)^* (0 + 1)^* 0$

$(11+00)$ $(0+1)$ $(0+1)$ 0
$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$
00 0 1 0, hence its a member.

(d) $(11 + (00)^*1)^*$

In this regular expression, number of 0's will always be divisible by 2, but the given string contains three zeros, which illustrates that, the string is not a member of the given regular expression.

24. **(d)**

Considering each of the options:

- '*w*' = 1 is present in the language but can't be accepted by M.
- '*w*' = 00 is present in the language but can't be accepted by M.
- '*w*' = 111 is present in the language but can't be accepted by M.

Option (d) is correct.

25. **(d)**

Considering the languages:

L_1 : This language is non-regular, since '*b*' can be any in number more than 3, so a's should in such a way that they are less than or equal to '*b*' which needs a comparison.

Hence language is DCFL.

L_2 : $\{a^n b^m \mid n \geq 1, m \geq 1\} \cap \{a^n b^n \mid n \geq 1\}$
$= \{a^n b^n \mid n \geq 1\}$

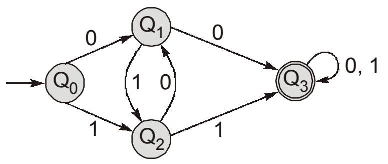
Which is a non regular DCFL.

Hence, both L_1 and L_2 are non-regular.

26. **(b)**

$(0 + 1)^* (00 + 11) (0 + 1)^*$ = (Containing 00 or 11) (atleast 2 consecutive 0's or 2 consecutive 1's)

DFA :



The number of states in minimized DFA for regular expression $(0 + 1)^* (00 + 11) (0 + 1)^*$ is 4.

27. **(b)**

The PDA is designed, initially on every '*a*' in the string '*a*' is pushed in the stack, then on every '*b*' in the string, a skip operation is performed. For, every '*c*' in the string '*a*' is popped from the stack, but string is accepted only if when input is finished and the stack still contains '*a*' which shows that number of '*a*' should be more than '*c*'.

Hence,

$$L = \{a^n b^m c^p \mid n > p;\ n, m > 0;\ p \geq 0\}.$$

28. **(c)**

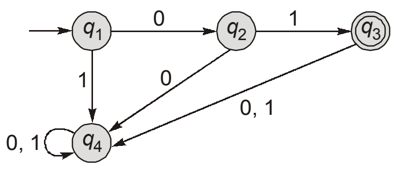
- 'aaa' can't be generated
- 'bbbb' can't be generated
- $S \rightarrow a^{m+1} b^m + b^{n+1} a^n + a^m b^{m+n} a^n + \in$

Hence, 'abba' is generated

- 'abab' can't be generated

29. (a)

$$\begin{aligned} L &= \{0^n 1 \mid n \geq 0\} \cap \{01^n \mid n \geq 0\} \\ &= \{1, 01, 001, 0001, ...\} \cap \{0, 01, 011, 0111, ...\} \\ &= \{01\} \end{aligned}$$



30. (b)

$$\begin{aligned} L &= \left\{\overline{ww \mid w \in \{0,1\}^*}\right\} \cap \{xy \mid |x| = |y|\} \\ &= \text{CFL} \cap \text{Regular} \\ &= \text{CFL} \end{aligned}$$

■■■■