EPT-TEST- 54(CD, Lexical, Syntax)
Total Questions: 15
Time: 60 Minutes

Q1.[MCQ]
Consider the following given grammar:
S → Aa
A → BD
B → b|ε
D → d|ε

Let a, b, d and $ be indexed as

a	b	d	$
3	2	1	0

Compute the FOLLOW set of the non-terminal B and write the index values for the symbols in the FOLLOW set in the descending order. (For example, if the FOLLOW set is {a, b, d, $}, then the answer should be 3210).

(A) 31
(B) 310
(C) 230
(D) 23

Q2. [MCQ]

Consider the following grammar G.
S → F | H
F → p | c
H → d | c

Where S, F and H are non-terminal symbols, p, d and c are terminal symbols. Which of the following statement(s) is/are correct

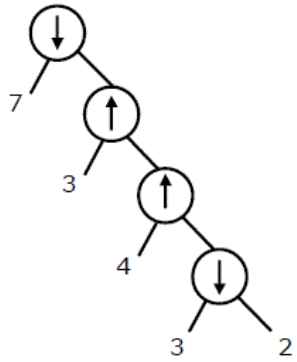
S1: LL(1) can parse all strings that are generated using grammar G.
S2: LR(1) can parse all strings that are generated using grammar G.

(A) Only S1
(B) Only S2
(C) Both S1 and S2
(D) Neither S1 and S2

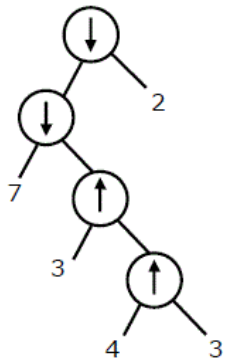
Q3.[MCQ]
Consider two binary operators '↑' and '↓' with the precedence of operator ↓ being lower than that of the operator ↑. Operator ↑ is right associative while operator ↓ is left associative.

Which one of the following represents the parse tree for expression (7↓3↑4↑3↓2)

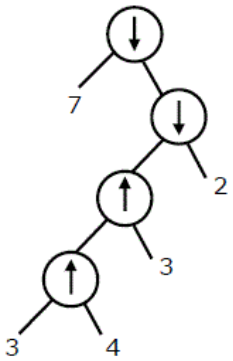
(A)



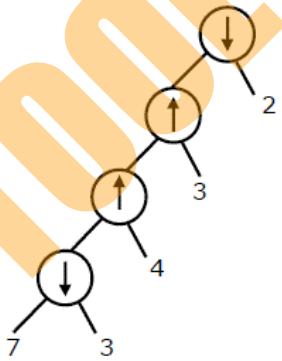
(B)



(C)



(D)



Q4.[MCQ]

Which of the following suffices to convert an arbitrary CFG to an LL(1) grammar?

(A) Removing left recursion alone
(B) Factoring the grammar alone
(C) Removing left recursion and factoring the grammar
(D) None of these

Q5. [MCQ]
The grammar A → AA | (A) | ε is not suitable for predictive-parsing because the grammar is

(A) ambiguous
(B) left-recursive
(C) right-recursive
(D) an operator-grammar

Q6. [MSQ]
For context-free grammar, FOLLOW(A) is the set of terminals that can appear immediately to the right of non-terminal A in some "sentential" form. We define two sets LFOLLOW(A) and RFOLLOW(A) by replacing the word "sentential" by "left sentential" and "right most sentential" respectively in the definition of FOLLOW (A).

A. **FOLLOW(A) and LFOLLOW(A) may be different.**
B. **FOLLOW(A) and RFOLLOW(A) are always the same.**
C. **All the three sets are identical.**
D. **All the three sets are different.**

Q7. [MCQ]
Match the following according to input(from the left column) to the compiler phase(in the right column) that process it:

(P) Syntax tree	(i) Code generator
(Q) Character stream	(ii) Syntax analyzer
(R) Intermediate representation	(iii) Semantic analyzer
(S) Token stream	(iv) Lexical analyzer

(A) P -> (ii), Q -> (iii), R -> (iv), S -> (i)
(B) P -> (ii), Q -> (i), R -> (iii), S -> (iv)
(C) P -> (iii), Q -> (iv), R -> (i), S -> (ii)
(D) P -> (i), Q -> (iv), R -> (ii), S -> (iii)

Q8.[MCQ]
The number of tokens in the following C statement is
printf("i = %d, &i = %x", i, &i);

(A) 3
(B) 26
(C) 10
(D) 21

Q9. [MCQ]
Consider the following two statements:

- **P: Every regular grammar is LL(1)**
- **Q: Every regular set has a LR(1) grammar**

Which of the following is TRUE?

A. Both P and Q are true
B. P is true and Q is false

C. P is false and Q is true

D. Both P and Q are false

Q10.[MCQ]

Consider the CFG with {S,A,B} as the non-terminal alphabet, {a,b} as the terminal alphabet, S as the start symbol and the following set of production rules:

S→aB S→bA

B→b A→a

B→bS A→aS

B→aBB S→bAA

For the string aabbab, how many derivation trees are there?

A. 1

B. 2

C. 3

D. 4

Q11.[MCQ]

A lexical analyzer uses the following patterns to recognize three tokens T_1, T_2, and T_3 over the alphabet {a,b,c}.

T_1: a?(b|c)*a

T_2: b?(a|c)*b

T_3: c?(b|a)*c

Note that 'x?' means 0 or 1 occurrence of the symbol x. Note also that the analyzer outputs the token that

matches the longest possible prefix.

If the string *bbaacabc* is processed by the analyzer, which one of the following is the sequence of tokens it outputs?

(A) $T_1T_2T_3$
(B) $T_1T_1T_3$
(C) $T_2T_1T_3$
(D) T_3T_3

Q12. [MCQ]

Consider the grammar shown below

S → i E t S S' | a
S' → e S | ε
E → b

In the predictive parse table M, of this grammar, the entries M[S', e] and M[S', $] respectively are

A. {S' → e S} and {S' → e}
B. {S' → e S} and {}
C. {S' → ε} and {S' → ε}
D. {S' → e S, S'→ ε} and {S' → ε}

Q13.[MCQ]

Given the following expression grammar:

E -> E * F | F + E | F

F -> F - F | id

Which of the following is true?

A.* has higher precedence than +
B.– has higher precedence than *
C.+ and — have same precedence
D.+ has higher precedence than *

Q14. [MCQ]

In the following grammar

X :: = X ⊕ Y / Y
Y :: = Z * Y / Z
Z :: = id

Which of the following is true?

A. ⊕ is left associative while * is right associative

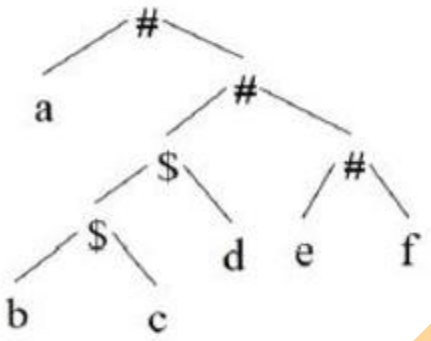
B. Both ⊕ and * are left associative

C. ⊕ is right associative while * is left associative

D. None of the above

Q15. [MCQ]

Consider the following parse tree for the expression a#bcd#e#f, involving two binary operators $ and #.



Which one of the following is correct for the given parse tree?

A. **$ has higher precedence and is left associative; # is right associative**
B. **# has higher precedence and is left associative; $ is right associative**
C. **$ has higher precedence and is left associative; # is left associative**
D. **# has higher precedence and is right associative; $ is left associative**

ANSWERS

A1. A

A2. D

The given grammar is ambiguous as there are two possible leftmost derivations for string "c"

A3. B

A4. D

A5. A

A6. A, B

Ans - A,B.

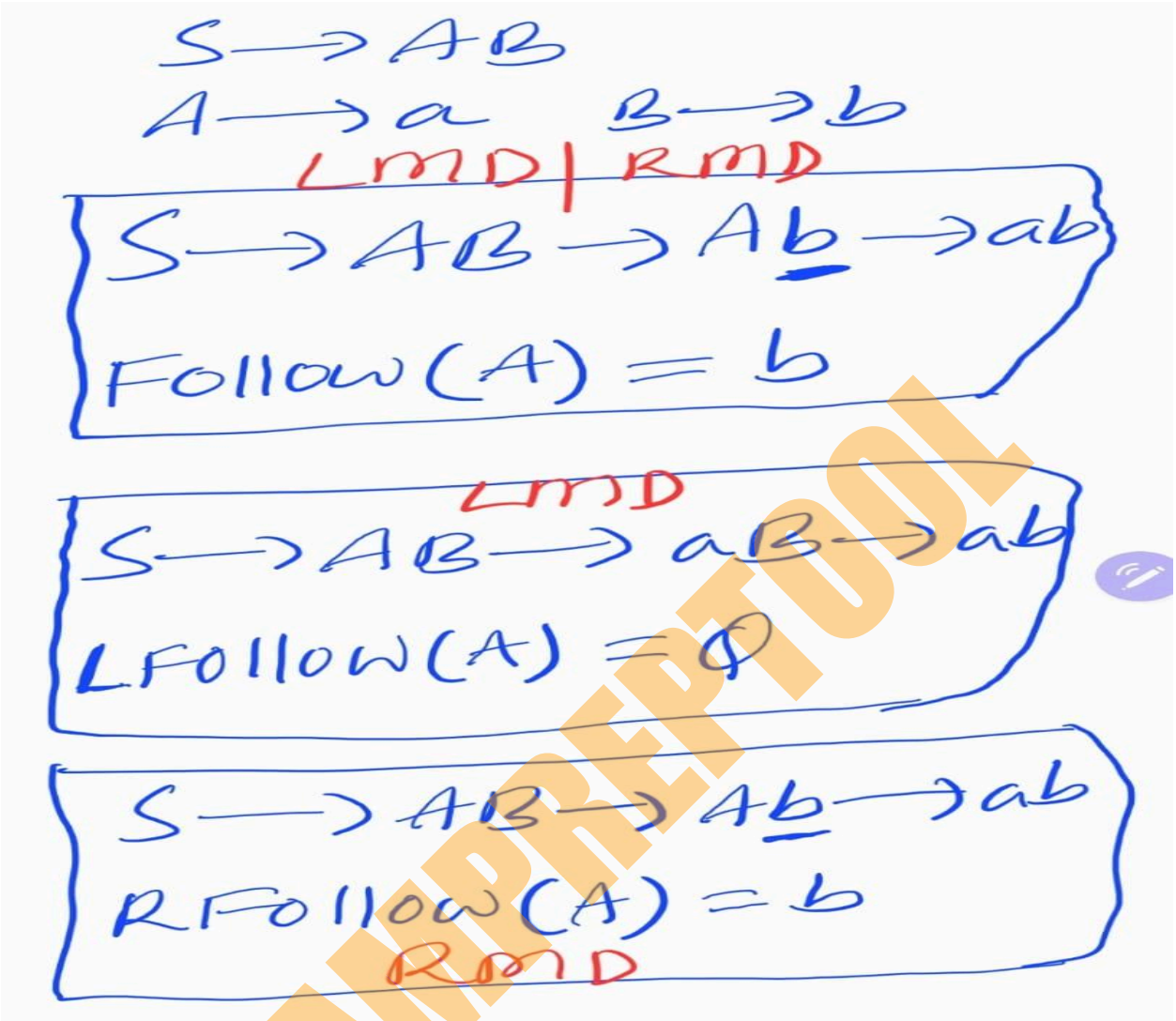
LFOLLOW may be different from FOLLOW but RFOLLOW and FOLLOW will be the same,

In FOLLOW(A), we add all terminals which appear on the immediate right of A in some sentential form. (LMD/RMD)

In RFOLLOW(A),we add all terminals which appear on the immediate right of A in some right sentential form. (RMD)

In LFOLLOW(A),we add all terminals which appear on the immediate right of A in some left sentential form. (LMD)

Consider the following grammar



The above example proves that

LFOLLOW may not always be the same as FOLLOW but d prove that RFOLLOW and FOLLOW will always be the same.

A7. C
A8. C
A9. C

P is false

S->Sa/b

S->aS/a

These are regular grammars but not LL(1)

Q is true

Every Regular Language is DCFL

Every DCFL has LR(1) Grammer, hence all regular languages are parsed by LR(1) parser.

Hence, Statement Q is True.

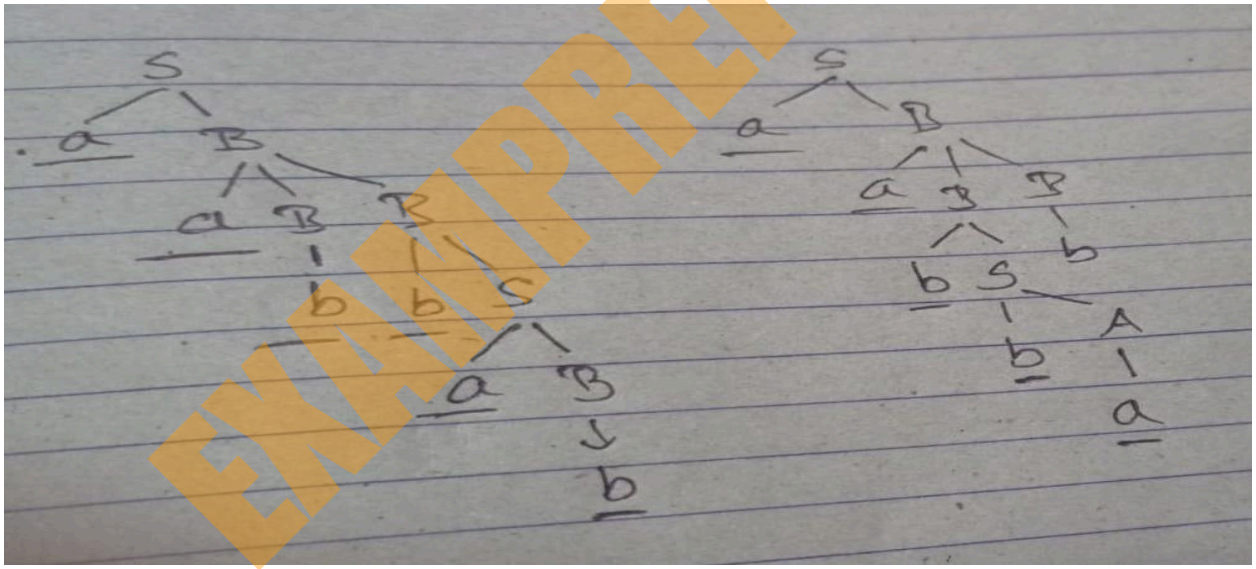
Note also that

Every LR(1) grammar is Unambiguous

So Every DCFL is unambiguous

So Every regular language is also Unambiguous

A10. B



A11. D
A12. D
A13. B
A14. A
A15. A