COMPILER DESIGN

S → aSAb | bSBc
A → +AB | ε
B → *BC | ε
C → aC | d

1. What is in the Follow(S)?

(a) {a, b, c, +, $}
(b) {a, c, +, *, $}
(c) {b, c, +, *, $}
(d) {a, b, d, *, $}

2. What is in the Follow(B)?

(a) {a, b, c, d, *}
(b) {a, b, d, ε, $}
(c) {a, c, d, *, $}
(d) {c, d, b, +, *}

3. Choose the False statement.

(a) No left recursive/ ambiguous grammar can be LL(1)
(b) The class of grammars that can be parsed using LR methods is proper subset of the class of grammar that can be parsed by LL method
(c) LR parsing is non-backtracking method
(d) LR parsing can describe more languages than LL parsing

4. Consider the following SDT.

A → BC *(I) B.i = f(A.i)
(II) B.i = f(A.S)
(III) A.S = f(B.s)

Which of the above is violating L – attributed definition?

(a) I only
(b) II only
(c) I, II
(d) I, II, III

5.

$X \rightarrow YZ$

$Y \rightarrow Y + Z$ {print ('+');}
T {Y.val = T.val}

$Z \rightarrow *Y$ {print ('*');} Z
T {Z.val = T.val}
ε

$T \rightarrow$ num {print(num.val);}

For 2+3*2, the above translation scheme prints

(a) 2+3*2
(b) 23+2*
(c) 232*+
(d) 23*2+

6. Consider the following expression

x = a*b – c*d+e

For generating target code how many register will be required apart from accumulator A?

(a) 1
(b) 2
(c) 3
(d) 5

7. Consider the following two grammars

G_1: $A \rightarrow A1 \mid 0A1 \mid 01$
G_2: $A \rightarrow 0A \mid 1$

Which of the following is True regarding above grammars?

(a) L_1 is LR(k)
(b) L_2 is LR(k)
(c) Both L_1 and L_2 is LR(k)
(d) None is LR(k)

8. Consider the following grammar.

$S \rightarrow aB \mid aAb$
$A \rightarrow bAb \mid a$
$B \rightarrow aB \mid \varepsilon$

How many back tracks are required to generate the string aab from the above grammar?

(a) 1
(b) 2
(c) 3
(d) 4