7 Compiler Design

Practice Questions

Q.1 Which of the following statements is false?

(A) Unambiguous grammar has different derivation tree

(B) An LL(1) parser is a top-down parser

(C) LALR is more powerful than SLR

(D) Ambiguous grammar can't be LR(k)

Q.2 Assume that the CLR parser for a grammar G has n_1 states and the LALR parser for G has n_2 states. The relationship between n_1 and n_2 is

(A) n_1 is necessarily less than n_2

(B) n_1 is necessarily equal to n_2

(C) n_1 is necessarily greater than n_2

(D) n_1 is necessarily greater than or equal to n_2

Q.3 Consider the following grammar $G = \{$bexpr, {bexpr, bterm, bfactor}, {not, or, and, ( ), true, false}, $P\}$ with P given below

bexpr → bexpr or bterm | bterm
bterm → bterm and bfactor | bfactor
bfactor → not bfactor | (bexpr) | true | false

The equivalent non-left recursive grammar for the given grammar is

(A) bexpr → bterm E'
E' → or bterm E' | ε
bterm → bfactor F'
F' → and bfactor F'
bfactor → not bfactor | (bexpr) | true | false

(B) bexpr → bterm E'
E' → or bterm E' | ε
bterm → bfactor and F'
F' → bfactor F' | ε
bfactor → not bfactor | (bexpr) | true | false

(C) bexpr → bterm E'
E' → or bterm E' | ε
bterm → bfactor F'
F' → and bfactor F' | ε
bfactor → not bfactor | (bexpr) | true | false

(D) bexpr → bterm E'
E' → or bterm E'
bterm → bfactor F'
F' → and bfactor F' | ε
bfactor → not bfactor | (bexpr) | true | false

Q.4 Consider the following Code fragment

```
int main ()
{
  int x, y, total;
  x = 10, y = 20;
```

$total = x + y;$

$printf("Total = \%d\backslash n", total);$

}

Number of tokens in the given code fragment is ______.

Q.5 Match the description of several parts of a classic optimizing compiler in **List-I,** with the names of those parts in **List-II:**

	List-I		List-II
(a)	A part of a compiler that is responsible for recognizing syntax.	(i)	Optimizer
(b)	A part of a compiler that takes as input a stream of characters and produces as output a stream of words along with their associated syntactic categories.	(ii)	Semantic Analysis
(c)	A part of a compiler that understand the meanings of variable names and other symbols and checks that they are used in ways consistent with their definitions.	(iii)	Parser
(d)	An IR-to-IR transformer that tries to improve the IR program in some way (Intermediate Representation).	(iv)	Scanner

Code :

	(a)	(b)	(c)	(d)
(1)	(iii)	(iv)	(ii)	(i)
(2)	(iv)	(iii)	(ii)	(i)
(3)	(ii)	(iv)	(i)	(iii)
(4)	(ii)	(iv)	(iii)	(i)

(A) 1 (B) 2

(C) 3 (D) 4

Q.6 Consider the following code segment.

$x = u - t;$

$y = x \times V;$

$z = y + w;$

$w = t - z;$

$u = x \times y;$

The minimum number of total variables required to convert the above code segment to static singe assignment form is _____

Q.7 The left factored and non-left recursive grammar for the given grammar is

Numeral : :-

Digits | Digits . Digits

| Digits e Sign Digits

| Digits . Digits e Sign Digits

Digits : :- Digit | Digits Digit

Digit : :- 0 | 1 | 2 | 3

(A) Numeral : := Digits N1 N2

N1 : := θ Sign Digits | ε

N2 : := Digits | ε

Digits : := Digit Digits | Digit

Digit : := 0 | 1 | 2 | 3

(B) Numeral : := Digits N1 N2

N1 : := θ Sign Digits | ε

N2 : := Digits | ε

Digits : := Digit Digits | Digit

Digit : := 0 | 1 | 2 | 3

(C) Numeral : := Digits N

N : := θ Sign Digits | . Digits | ε

Digits : := Digit Digits | Digit

Digit : := 0 | 1 | 2 | 3

(D) Numeral : := Digits N1

N1 : := *e* Sign Digits | ε | .Digits N2

N2 : := θ Sign Digits | ε

Digits : := Digit D

D : := Digits | ε

Digit : := 0 | 1 | 2 | 3

Q.8 Consider the following grammar

SL → SL. S

SL → ε

S → stmt

The given grammar is

(A) LR(0) and SLR(1)

(B) Not LR(0) but SLR(1)

(C) Neither LR(0) nor SLR(1)

(D) LL(1) but not LR(0)

Q.9 The non-left recursive grammar from the given grammar is

$A \to B \mid a \mid CBD$

$B \to C \mid b$

$C \to A \mid c$

$D \to d$

(A) $A \to B \mid a \mid CBD$

$B \to C \mid b$

$C \to A \mid c$

$D \to d$

(B) $A \to aA' \mid bA' \mid cA' \mid cBDA'$

$A' \to \varepsilon \mid BDA'$

$B \to C \mid b$

$C \to A \mid c$

$D \to d$

(C) $A \to aA' \mid bA' \mid cA'$

$A' \to \varepsilon \mid BDA'$

$B \to C \mid b$

$C \to A \mid c$

$D \to d$

(D) $A \to aA' \mid bA' \mid cBDA'$

$A' \to \varepsilon \mid BDA'$

$B \to C \mid b$

$C \to A \mid c$

$D \to d$

Q.10 Consider the following grammar:

$R \to R \mid R$

$R \to RR$

$R \to R*$

$R \to (R)$

$R \to a$

$R \to b$

where the terminals are $\{|, *, (,), a, b\}$

Follow(R) is _____

(A) $\{\$\}$

(B) $\{|, *,)\}$

(C) $\{|, *,), \$\}$

(D) $\{(, a, b, |, *,), \$\}$

Q.11 Consider the following grammar

$S \to CC$

$C \to cC$

$C \to d$

The number of canonical collections of CLR(1) items which are having the same reductions with different lookaheads is

(A) 0 (B) 1

(C) 2 (D) 3

Q.12 Consider the following context-free grammar, the symbols (, a ,) and , are terminals and S is the initial symbol.

$S \to (L)$

$S \to a$

$L \to L, S$

$L \to S$

The closure of the LR(1) item $[S \to (\cdot L)[\$]$ is

(A) $S \to (.L)|\$$

$L \to .L, S|)$

$L \to .S,|)$

(B) $S \to (.L)|\$$

$L \to .L, S|)$

$L \to .S,|)$

$S \to .(L) \,|\,)$

$S \to .a \,|\,)$

(C) $S \to (.L), \$$

$L \to .L, S,)$

$L \to .S,)$

$S \to .(L),)$

$S \to .a,)$

(D) $S \to (.L)|\$$

$L \to .L, S| \,)$

$L \to .S, \,|\,)$

$L \to .L, S \,|,$

$L \to .S \,|,$

$S \to .(L) \,|\,)$

$S \to .a \,|\,)$

$S \to .(L) \,|\, ,$

$S \to .a \,|\, ,$

Q.13 Consider the following grammar with the semantic rules

Grammar	Semantic Rules
$E_1 \to E_2 + T$	E_1.string $= E_1$.string $\|\|$ T.string $\|\|$ '*'
$E_1 \to T$	E_1.string $= T$.string
$T_1 \to T_2 * F$	T_1.string $= T_2$.string $\|\|$ F.string $\|\|$ '+'
$T \to F$	T.string $\to F$.string
$F \to (E)$	F.string $\to E$.string
$F \to$ num	F.string $\to$ num.string

The output produced by the SDT for the input string "$3*4+5*2$" is _____

(A) $34*+52+$ (B) $34*52*+$

(C) $34+52+*$ (D) $34+*52+$

Q.14 Consider the following grammar **[MSQ]**

$S \to A$

$A \to BC \,|\, DBC$

$B \to Bb \,|\, \varepsilon$

$C \to c \,|\, \varepsilon$

$D \to a \,|\, d$

Which of the following is part of the FIRST(A) is

(A) $\{a, b\}$ (B) $\{c, d\}$

(C) $\{a, b, c, d, \varepsilon\}$ (D) $\{a, d\}$

Q.15 Compiler can check _____ error **[MSQ]**

(A) Logical (B) Syntax

(C) Semantic (D) All of them

Q.16 Consider the following SDT

$A \to b$ {print("a")}A

$A \to a$ {print("b")}A

$A \to c$ {print("d")}

The output produced by the SDT for the input string *bbac*

(A) *aabd* (B) *abda*

(C) *dbaa* (D) None of these

Q.17 Consider the following context free grammar

$S \to ABBA$

$A \to a \,|\, \varepsilon$

$B \to b \,|\, \varepsilon$

The entries in the following LL(1) parse table M is

	a	b	\$
S	$S \to ABBA$	$S \to ABBA$	$S \to ABBA$
A			
B		$B \to b$	

The entries for the

$M[A, a], M[A, b], M[A, \$]$ is

(A) $M[A,a] = A \to a,$
$M[A,b] = A \to \varepsilon,$
$M[A,\$] = A \to \varepsilon$

(B) $M[A,a] = \{A \to a, A \to \varepsilon\}$
$M[A,b] = A \to \varepsilon,$
$M[A,\$] = A \to \varepsilon$

(C) $M[A,a] = A \to a,$
$M[A,b] = A \to \varepsilon,$
$M[A,\$] =$ Nil

(D) $M[A,a] = \{A \to a, A \to \varepsilon\}$
$M[A,b] = A \to \varepsilon,$
$M[A,\$] =$ Nil

Q.18 Given the following expression grammar:

$E \to E * F \mid F + E \mid F$

$F \to F - F \mid id$

The output produced by the expression grammar after evaluating the expression $5+3*4-2$

(A) 16 (B) 10
(C) 4 (D) None of these

Q.19 In a bottom-up evaluation of a syntax direction definition, inherited attributes can

(A) Always be evaluated
(B) Be evaluated only if the definition is L-attributed
(C) Evaluation only done if the definition has synthesized attributes
(D) None of these

Q.20 Consider the following statements related to compiler construction:

I. Lexical Analysis is specified by context-free grammar and implemented by pushdown automata.
II. Syntax Analysis is specified by regular expressions and implemented by finite-state machine.

Which of the above statement(s) is/are incorrect?

(A) Only I (B) Only II
(C) Both I and II (D) Neither I nor II

Q.21 Consider the following Grammar

$S \to X$

$X \to Yb \mid aa$

$Y \to a \mid bYa$

The given grammar is

(A) LR(0) and SLR(1)
(B) Not LR(0) but SLR(1)
(C) LR(0) but not SLR(1)
(D) Neither LR(0) nor SLR(1)

Q.22 Consider the following syntax directed translation scheme.

$E \to E_1 * T \{E.val = E_1.val * T.val\}$

$E \to T \{E.val = T.val\}$

$T \to F - T_1 \{T.val = F.val - T_1.val\}$

$T \to F \{T.val = F.val\}$

$F \to 3 \{F.val = 2\}$

$F \to 5 \{F.val = 4\}$

The output produced by the SDTS after evaluating the given expression is $5-3*5*3$.

Assume attribute evaluation with bottom-up parsing, i.e., attributes are evaluated immediately after a reduction.

(A) 15 (B) – 42
(C) 30 (D) None of these

Q.23 Consider the following context free grammar

$S \to P$

$P \to (P)P$

$P \to \varepsilon$

In the LL(1) parse table M, the entries for $M[S,\$]M[P,)]$ are

(A) $S \to P, P \to \varepsilon$

(B) $S \to \varepsilon, P \to \varepsilon$

(C) $P \to \varepsilon, P \to (P)P$

(D) None of these

Q.24 Given the CFG

$G = \{S, \{S, U, V, W\}, \{a, b, c, d), P\}$ with

P given as shown below:

$S \to UVW$

$U \to (S) \mid aSb \mid d$

$V \to aV \mid \varepsilon$

$W \to cW \mid \varepsilon$

Then Follow $(U) =$

(A) $\{a, c\}$ (B) $\{a, c, b,)\}$

(C) $\{a, c, \$, b\}$ (D) $\{a, c, \$, b,)\}$

Q.25 Consider the following context free grammar

$S \to bAB \mid bb \mid C$

$A \to BC \mid aCB \mid \varepsilon \mid a$

$B \to bB \mid C \mid \varepsilon$

$C \to aaC \mid bbC \mid D$

$C \to a \mid b$

The Equivalent simplified (After Elimination of Unit, Null, Useless Symbols) from the grammar is

(A) $S \to bAB \mid bb \mid aaC \mid bbC \mid a \mid b$

$A \to BC \mid aCB \mid a$

$B \to bB \mid aaC \mid bbC \mid a \mid b$

$C \to aaC \mid bbC \mid a \mid b$

$D \to a \mid b$

(B) $S \to bAB \mid bb \mid aaC \mid bbC \mid a \mid b$

$A \to BC \mid aCB \mid a$

$B \to bB \mid aaC \mid bbC \mid a \mid b$

$C \to aaC \mid bbC \mid a \mid b$

(C) $S \to bAB \mid bb \mid aaC \mid bbC$

$A \to BC \mid aCB \mid a$

$B \to bB \mid aaC \mid bbC \mid a \mid b$

$C \to aaC \mid bbC \mid a \mid b$

$D \to a \mid b$

(D) None of these

Q.26 Which one of the following statements is FALSE?

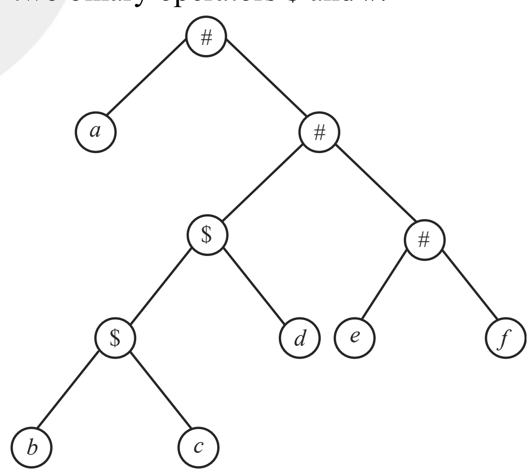
(A) Context-free grammar can be used to specify both lexical and syntax rules.

(B) Type checking is done before parsing.

(C) High-level language programs can be translated to different Intermediate Representations.

(D) Arguments to a function can be passed using the program stack.

Q.27 Consider the following parse tree for the expression $a \# b \$ c \$ d \# e \# f$, 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

Q.28 The attributes of three arithmetic operators in some programming languages are given below:

Operator	Precedence	Associativity	Arity
+	High	Left	Binary
–	Medium	Right	Binary
*	Low	Left	Binary

The value of the expression $2-5+1-7*3$ in this language is

(A) 9 (B) 10

(C) 11 (D) 12

Q.29 Which of the following statements are CORRECT?

1. Static allocation of all data areas by a compiler makes it impossible to implement recursion.
2. Automatic garbage collection is essential to implement recursion.
3. Dynamic allocation of activation records is essential to implement recursion.
4. Both heap and stack are essential to implement recursion.

(A) 1 and 2 only (B) 2 and 3 only

(C) 3 and 4 only (D) 1 and 3 only

Q.30 For the grammar below, a partial LL(1) parsing table is also presented along with the grammar. Entries that need to be filled are indicated as E_1, E_2 and E_3. ε is the empty string, $ indicates end of input, and 1 separate alternate right hand sides of productions.

$S \to aAbB \mid bAaB \mid \varepsilon$

$A \to S$

$B \to S$

	A	b	$
S	E_1	E_2	$S \to \varepsilon$
A	$A \to S$	$A \to S$	error
B	$B \to S$	$B \to S$	E_3

The FIRST and FOLLOW sets for the non-terminals A and B are

(A) $\text{FIRST}(A) = \{a, b, \varepsilon\} = \text{FIRST}(B)$

$\text{FOLLOW}(A) = \{a, b\}$

$\text{FOLLOW}(B) = \{a, b, \$\}$

(B) $\text{FIRST}(A) = \{a, b, \$\}$

$\text{FIRST}(B) = \{a, b, \varepsilon\}$

$\text{FOLLOW}(A) = \{a, b\}$

$\text{FOLLOW}(B) = \{\$\}$

(C) $\text{FIRST}(A) = \{a, b, \varepsilon\} = \text{FIRST}(B)$

$\text{FOLLOW}(A) = \{a, b\}$

$\text{FOLLOW}(B) = \phi$

(D) $\text{FIRST}(A) = \{a, b\} = \text{FIRST}(B)$

$\text{FOLLOW}(A) = \{a, b\}$

$\text{FOLLOW}(B) = \{a, b\}$

Q.31 The appropriate entries for E_1, E_2 and E_3 are

(A) $E_1 : S \to aAbB, A \to S$

$E_2 : S \to bAaB, B \to S$

$E_3 : B \to S$

(B) $E_1 : S \to aAbB, S \to \varepsilon$

$E_2 : S \to bAaB, S \to \varepsilon$

$E_3 : S \to \varepsilon$

(C) $E_1 : S \to aAbB, S \to \varepsilon$

$E_2 : S \to bAaB, S \to \varepsilon$

$E_3 : B \to S$

(D) $E_1 : A \to S, S \to \varepsilon$

$E_2 : B \to S, S \to \varepsilon$

$E_3 : B \to S$

Q.32 The program below uses six temporary variables a, b, c, d, e, f.

a = 1

b = 10

c = 28

d = a + t

e = c + d
f = c + e
b = c + e
e = b + f
d = 5 + e
return d + f

Assuming that all operations like their operands from registers, what is the minimum number of registers needed to execute this program without spilling?

(A) 2 (B) 3
(C) 4 (D) 6

Q.33 Which of the following statements are TRUE?

I. There exist parsing algorithms for some programming languages whose complexities are less than $\theta(n^3)$.

II. A programming language which allows recursion can be implemented with static storage.

III. No L-attributed definition can be evaluated in the framework of bottom-up parsing.

IV. Code improving transformations can be performed at both source language and intermediate code level.

(A) I and II (B) I and IV
(C) III and IV (D) I, III and IV

Q.34 FOLLOW (S) set for the postfix grammar after removing left recursion $S \rightarrow SS+ \mid SS* \mid a$.

(A) $\{a, \in\}$
(B) $\{+, *\}$
(C) $\{a, \$\}$
(D) None of the above

Q.35 Consider the following statement:

S_1: Three address code is linear representation of syntax tree

S_2: With triples representation optimization can change the execution order.

Which of the above is correct?

(A) Only S_1
(B) Only S_2
(C) Both S_1 and S_2
(D) None of these

Q.36 Consider the following grammar.

$$X \rightarrow YZ\{Z.x = X.x$$
$$X.y = Y.y\}$$
$$Z \rightarrow PZ'\{Z'.x = P.x$$
$$Z.y = P.x + Z'.y\}$$

Which of the following is true?

(A) Both x and y are inherited attributes
(B) Both x and y are synthesized attributes
(C) x is inherited and y is synthesized
(D) x is synthesized and y is inherited

Q.37 Choose the correct sequence of occurrence during compilation process

(A) Parse tree $\rightarrow$ Token stream $\rightarrow$ intermediate code
(B) Parse tree $\rightarrow$ 3 address code $\rightarrow$ character stream
(C) Character stream $\rightarrow$ Parse tree $\rightarrow$ SDT tree
(D) Token stream $\rightarrow$ SDT tree $\rightarrow$ Parse tree

Q.38 Consider the following code.

```
main ()
{
    int temp = 200, 10;
    int l1, l2;
    temp = ++temp;
    l1+ = l2
    printf("%d", temp + l1);
}
```

The number of tokens in the above code is______.

Q.39 Consider the regular expression with the respective token number in the table.

REX	Token No.
$(a+b)*c$	1
$ca*b$	2
$c*$	3

Choose the correct output when lexical analyzer scans the following input: "cabacccab" Note: The analyzer tries to output the token that matches the longest possible prefix.

(A) 3122

(B) 2132

(C) 1132

(D) Generates lexical error

Q.40 Consider the following translation scheme:

$S \to XY$

$X \to X*Y[\text{Print}[('*')]$

$X \to \text{id}[\text{Print}(\text{id})]$

$X \to \text{id}\ [\text{Print}\ (-)]$

$Y \to +Y[\text{Print}(-)]$

$Y \to \text{id}\{\text{Print}\ (\text{id})\}$

Here id is a token which represent on integer id represent the value of that integer. For an input $6*4*5+7$, this translation scheme prints.

(A) $64*5*7+$ (B) $6*4*5-7$

(C) $64*5*7-$ (D) $64*5-*7-$

Q.41 Consider the following statements:

S_1: While program in execution, access to heap memory is slower as compared to accessing variables allocate on stack.

S_2: While program in execution, in a multithread situation, each threads has its own stack and share a common heap memory.

S_3: During a program execution, heap is stored in main memory and stack is present in secondary memory.

Which of the above is incorrect?

(A) Only S_1 and S_2

(B) Only S_2 and S_3

(C) All of these

(D) Only S_3

Q.42 Consider the basic block given below:

$X \to X*Y$

$Z \to X+Z$

$P \to Z/P$

$X \to Z+P$

Minimum number of edges present in the DAG representation of the above block is________.

Q.43 Consider the following statement:

I. Three address code is a linearized representation of syntax tree.

II. Type checking is done during all the phases especially in syntax analysis phase.

III. Target code generation phase is machine independent code generation

IV. Symbol table is accessed during lexical, syntax and semantic analysis phase.

The number of the correct statement is/are________.

Q.44 In SLR parsing for the grammar.

$E \to E$

$E \to aEbE \backslash bEaE \backslash \in$

In state 0, for input 'a' and 'b'

(A) Both will have shift reduce conflict

(B) Only 'a' will have shift reduce conflict

(C) Only 'b' will have shift reduce conflict

(D) Neither of the other options

Q.45 Which of the following statements is true?

(A) $S \rightarrow$ *aabc/ab,* this grammar is not LL(1) but it is LL(2).

(B) Every regular language is LL (1)

(C) Every regular grammar is LL (1)

(D) Both (a) and (b)

Q.46 Consider the following grammar G1 and G2 with S, A, B, C as non-terminals and a, b, c $\in$ as terminals.

$$G_1 : S \rightarrow A + B \mid A \mid B \mid AB$$
$$A \rightarrow A * C \mid a$$
$$B \rightarrow B + C \mid b$$
$$C \rightarrow c$$

$$G_2 : S \rightarrow A * B \mid \in$$
$$S \rightarrow B - C$$
$$A \rightarrow a$$
$$B \rightarrow b$$
$$C \rightarrow c$$

Which of the above grammar is operator grammar?

(A) Only G_1

(B) Only G_2

(C) Both G_1 and G_2

(D) None of these

Q.47 If we merge states in LR(1) parser to form a LALR(1) parser, we may introduce

(A) shift-reduce conflict

(B) reduce-reduce conflict

(C) no extra conflict

(D) both shift-reduce as well as reduce-reduce

Q.48 Suppose we have a rightmost derivation which proceeds as follows:

$$S \rightarrow Aabw$$
$$\rightarrow ABw$$

Which of the following is a possible handle for it?

(A) $A \rightarrow ab$ (B) $A \rightarrow a$

(C) $S \rightarrow A$ (D) $B \rightarrow ab$

Q.49 Which of the following statements is FALSE?

(A) In a SLR(1) parser, it is allowable for both shift and reduce items to be in the same state

(B) In a SLR(1) parser, it is allowable for multiple reduce items to be in the same state

(C) All SLR(1) grammars are LR(0)

(D) All LR(0) grammars are SLR(1)

Q.50 Which of the followign statements regarding LR(0) parser is FALSE?

(A) A LR(0) configurating set cannot have multiple reduce items

(B) A LR(0) configurating set cannot have both shift as well as reduce items

(C) If a reduce item is present in a LR(0) configurating set it cannot have any other item

(D) A LR(0) parser can parse any regular grammar

Q.51 Which of the following sentences is CORRECT?

(A) A top-down parse produces a leftmost derivation of a sentence

(B) A bottom-up parse produces a rightmost derivation of a sentence

(C) A top-down parse produces a rightmost derivation of a sentence

(D) A bottom-up parse produces a leftmost derivation of a sentence

Q.52 Which of the following is TRUE regarding (LL0) grammar?

(A) We can have a LL(0) grammar for any regular language

(B) We can have a LL(0) grammar for a regular language only if it does not contain empty string

(C) We can have a LL(0) grammar for any regular language if and only if it has prefix property

(D) We can have a LL(0) grammar for only single string languages

Q.53 Match the following :

(i) LL(1)	(a) bottom-up
(ii) Recursive Descent	(b) Predictive
(iii) Recursive Ascent	(c) Top-down
(iv) LR(1)	(d) Deterministic CFL

(A) i-b; ii-c; iii-a; iv-d

(B) i-d; ii-a; iii-c; iv-d

(C) i-c; ii-b; iii-d; iv-a

(D) i-a; ii-c; iii-b; iv-d

Q.54 Which of the below relations does hold TRUE regarding GRAMMARS?

(A) $\text{LL}(1) \subset \text{SLR}(1) \subset \text{LR}(1)$

(B) $\text{SLR}(1) \subset \in -\text{free LL}(1) \subset \text{LR}(1)$

(C) $\in -\text{free LL}(1) \subset \text{SLR}(1) \subset \text{LR}(1)$

(D) $\text{LL}(1) \subset \text{SLR}(1) = \text{LR}(1)$

Q.55 The worst case space complexity of operator function table and operator relation table is?

(A) $O(n) \& O(n)$

(B) $O(n^2) \& O(n^2)$

(C) $O(n) \& O(n^2)$

(D) $O(n^2) \& O(n)$

Q.56 Consider the grammar shown below

$E \to E + T / T$

$T \to TF / F$

$F \to F * |a| b$

The minimum number of states are required in SLR (1) parsing table is____.

Q.57 Find the number of SR and RR conflicts in DFA with LR (0) items $S \to SS \mid a \mid \in$

(A) 2, 2 (B) 2, 1

(C) 3, 1 (D) 3, 2

Q.58 Consider the syntax direction definition shown below

$N \to L$ {N. val = L. val;}

$L \to LB$ {L.val = 2 * L.val + B. val;}

|B {L. val = B. val;}

$B \to 0$ {B.val = 0;}

| 1 {B.val= 1; }

If the input is 1010101 then its output are:-

Answers Compiler Design

1.	A	2.	D	3.	C	4.	34	5.	A
6.	7	7.	D	8.	A	9.	B	10.	D
11.	C	12.	C	13.	A	14.	A, B, C, D	15.	B, C
16.	A	17.	B	18.	A	19.	B	20.	C
21.	D	22.	D	23.	A	24.	D	25.	D
26.	B	27.	A	28.	A	29.	D	30	A
31.	C	32.	B	33.	B	34.	B	35.	A
36.	C	37.	C	38.	34	39.	D	40	C
41.	D	42.	8	43.	2	44.	A	45.	D
46.	D	47.	B	48.	D	49.	C	50.	D
51.	A	52.	D	53.	A	54.	C	55.	C
56.	10	57.	B	58.	85				

Explanations Compiler Design

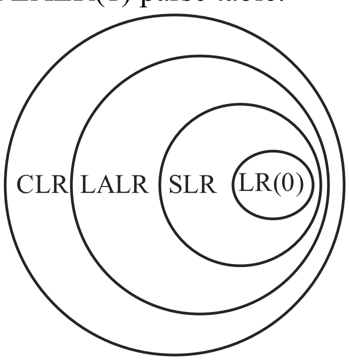
1. (A)

Unambiguous grammar has both kinds of derivation : False

In Unambiguous grammar both LMD and RMD generates the unique parse tree for the given input string

2. (D)

Number of states in a CLR parser table greater than equal LALR(1) parse table.



3. (C)

Non-left recursive grammar for the given grammar is

bexpr → bterm E'

E' → or bterm E' | ε

bterm → bfactor F'

F' → and bactor F' | ε

bfactor → not bfactor | (bexpr) | true | false

4. 34

int main() –4

{ –1

int x, y, total; –7

$x = 10, y = 20;$ –8

total $= x + y;$ –6

printf ("Total = %d\n", total); –7

} –1

Total number of tokens are

$$4+1+7+8+6+7+1=34$$

5. (A)

Parser is a part of compiler and responsible for syntax recognition. Scanner (or tokenization) used by the lexical analyser. In Semantic

analysis consistency and definition of syntax is checked. An optimizer is used improve the IR program.
Hence, the correct option is (A).

6. **7**

Static Single Assignment is used for intermediate code in compiler design, In static single Assignment form (SSA) each assignment to a variable should be specified with distinct names. We use subscripts to distinguish each definition of variables.
In the given code segment each definition is distinct.
So, the total number of variable is

(x, u, t, y, v, z, w).

7. **(D)**

The given grammar is
Numeral : := Digits / Digits. Digits / Digits e Sign Digits / Digits. Digits e
Sign Digits
Digits : := Digit Digits | Digit
Digit : := 0 | 1 | 2 | 3
Apply Left Factoring
Numeral : := Digits N1 N2
N1 : := e Sign Digits | ℇ
N2 : := Digits | ℇ
Digits : := Digit D
D : := Digits | ℇ
Digit : := 0 | 1 | 2 | 3

8. **(A)**

LR(0) Items : No conflicts in LR(0), SLR(1)

s0 :
[S' –> .SL]
[SL–> .SL; S]
[SL–> .] goto [s0, SL] = s1;
s1 :
[S' –> SL.]
[SL–> SL. ; S] goto (s1, ;) = s2;
s2 :
[SL –> SL;.S]
[S –> .stmt] goto (s2, S) = s3;
s3 :
[SL –> SL; S.] goto (s2, stmt) = s3;
s4 :
[S –> stmt .] goto (s2, stmt) = s4;

The given grammar is LR(0) and SLR(1)
SLR(1) parse table is

		stmt	S	SL	S
s0	r2		r2		
s1	s2		Accept		
s2		s4			3
s3	r1		r1		
s4	r3		r3		

9. **(B)**

Given grammar is

$A \to B \mid a \mid CBD$
$B \to C \mid b$
$C \to A \mid c$
$D \to d$

It is having the indirect recursion.

$A \to C \mid b \mid a \mid CBD$
$B \to C \mid b$
$C \to A \mid c$
$D \to d$

$\to$ $A \to A \mid c \mid b \mid a \mid cBD \mid ABD$
$B \to C \mid b$
$C \to A \mid c$
$D \to d$

$\Rightarrow$ $A \to A \mid ABD \mid a \mid b \mid c \mid cBD$
$B \to C \mid b$
$C \to A \mid c$
$D \to d$

$\to$ $A \to aA' \mid bA' \mid cA' \mid cBDA'$

$$A' \to \varepsilon \mid BDA'$$
$$B \to C \mid b$$
$$C \to A \mid c$$
$$D \to d$$

10. (D)

As R is the start symbol of the grammar, add $\{\$\}$ to the following set.

Follow(R) $= \{\$\} u \{|,*,)\}$

$$R \to RR$$

Follow(R) = First(R)

First(R) $= \{(, a, b\}$

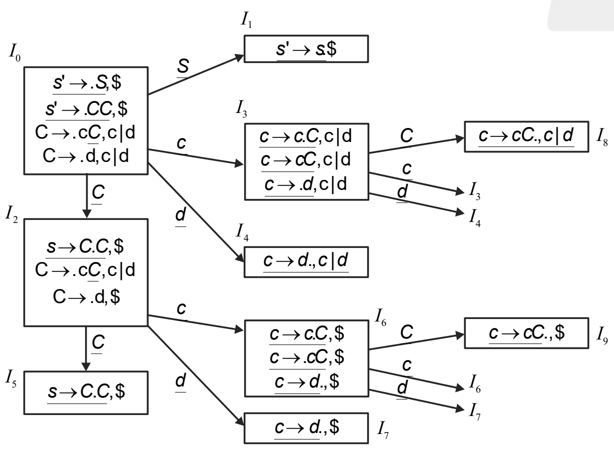
$\Rightarrow$ Follow(R) $= \{(, a, b, |, *,), \$\}$

11. (C)

Augmented grammar for the given grammar is

Let, $r_1 = S \to CC, r_2 = C \to cC$ and

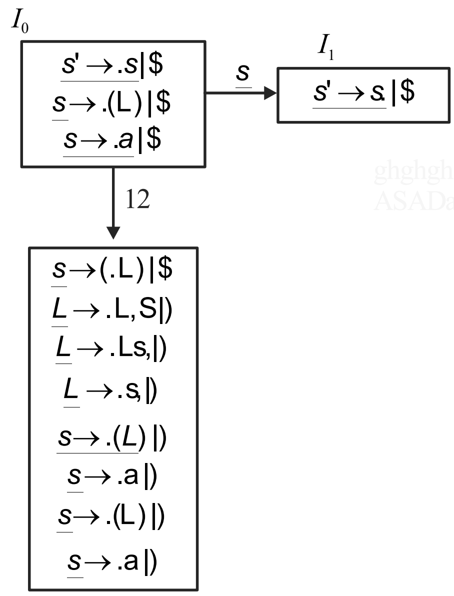
$$r_3 = C \to d$$
$$S' \to S$$
$$S \to CC$$
$$C \to cC$$
$$C \to d$$



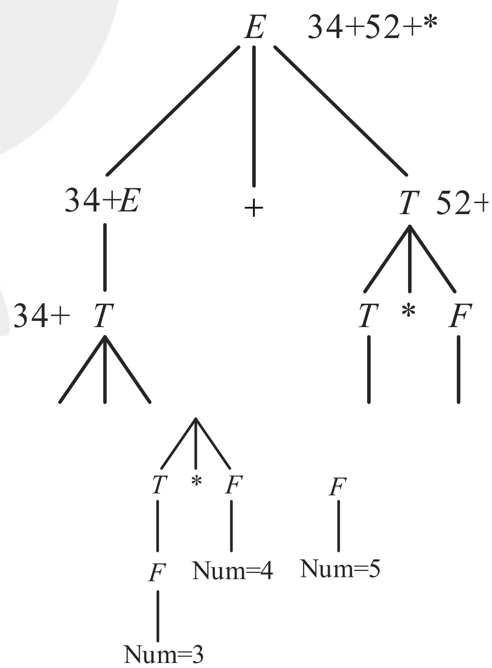
14 and 17 contain the same reductions (r_3) with different lookaheads.

18 and 19 contain the same reductions (r_2) with different lookaheads.

12. (C)



13. (C)



14. A,B,C,D

First(A) = First(B) u First(D)

First(D) = $\{a, d\}$

First(B) = $\{\varepsilon\}$

Substitute ε in place of B

First(B) = $\{b\}$

$\Rightarrow$ First(A) = First(C) = $\{c, \varepsilon\}$

Substitute ε in place of C

$\Rightarrow$First(A) $= \{\varepsilon\}$

$\Rightarrow$First(A) $= \{a, b, c, d, \varepsilon\}$

First(A) contains all the symbols $\{a, b, c, d, \varepsilon\}$

All the options are correct.

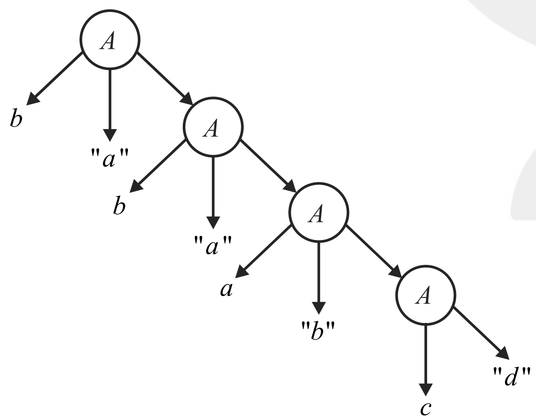
15. **B,C**

(B) Compiler will recognize all the syntax and semantic errors in the code, May not detect the logical errors.

(C) Compiler will recognize all the syntax and semantic errors in the code, May not detect the logical errors.

16. **(A)**

Given input string is *bbac* and the output produced by the SDT is *aabd*



17. **(B)**

First(S) $= \{a, b, \varepsilon\}$

First(A) $= \{a, \varepsilon\}$

First(B) $= \{b, \varepsilon\}$

Follow(S) $= \{\$\}$

$= \text{First}(B) = \{b\} \cup \text{First}(B)$

Follow(A) $= \{b\} \cup \text{First}(A)$

$= \{b, a\} \cup \text{follow}(S) = (a, b, \$\}$

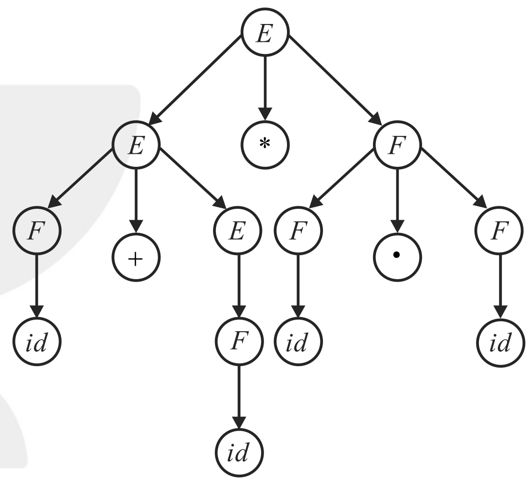
Follow(B) = First(B) $= \{b, a, \$\}$

LL(1) parse table is

	a	b	\$
S	$S \to ABBA$	$S \to ABBA$	$S \to ABBA$
A	$A \to a, A \to \varepsilon$	$A \to \varepsilon$	$A \to \varepsilon$
B	$B \to \varepsilon$	$B \to b, B \to \varepsilon$	$B \to \varepsilon$

18. **(A)**

The parse tree for the given string is $5+3*4-2$



The output is $(5+3)*(4-2) = 8*2 = 16$

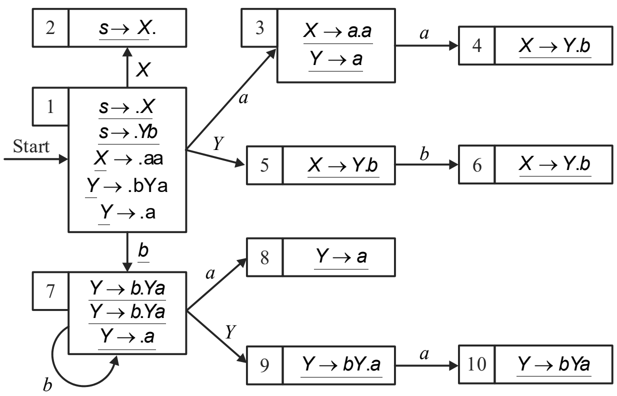
19. **(B)**

A Syntax Directed Definition (SDD) is called S Attributed if it has only synthesized attributes L. Attributed Definitions contain both synthesized and inherited attributes but do not need to build a dependency graph to evaluate them.

20. **(C)**

Both the statements are Incorrect. Lexical Analysis is specified by the Regular Expression and implemented by the finite state-machine and Syntax Analysis is specified by the CFG and implemented by the PDA.

21.	(D)



State 13 contains S/R conflict in LR(0). Apply reduction operation on state 13 and shift operation for the input symbol $'a'$.

Need to check for the S/R conflict in SLR(1)

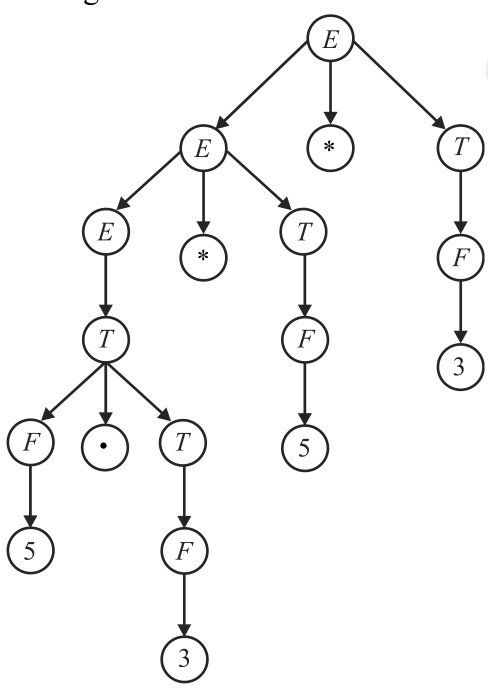
First(a) intersection Follow(Y) $= \{a\}$ intersection $\{a,b\} = \{a\} \neq \Phi$

S/R connect in SLR(1).

The given grammar is neither LR(0) nor SLR(1).

22.	(D)

Given string is $5-3*5*3$



$(((5-3)*5)*3)$

$\Rightarrow \quad F \to 3\{F.val = 2\}$

$F \to 5\{F.val = 4\}$

$(((4-2)*4)*2) \Rightarrow 16$

23.	(A)

First(S) $= \{(, \varepsilon\}$

First(P) $= \{(, \varepsilon\}$

Follow(S) $= \{\$\}$

Follow(P) $= \{), \$\}$

	(	)	$
S	$S \to P$		$S \to P$
P	$P \to (P)P$	$P \to \varepsilon$	$P \to \varepsilon$

24.	(D)

Follow(U) $=$ First(V) $= \{a, \varepsilon\}$

$\Rightarrow \quad \{a\} \cup \text{First}(W) = \{a\} \cup \{c, \varepsilon\}$

Substitute $\{\varepsilon\}$ in place of W

$\Rightarrow \quad \{a\} \cup \{c\}\,\text{Follow}(S)$

$\Rightarrow \quad \{a, c, \$, b,)\}$

25.	(D)

$S \to bAB \mid bb \mid C$

$A \to BC \mid aCB \mid \varepsilon \mid a$

$B \to bB \mid C \mid \varepsilon$

$C \to aaC \mid bbC \mid D$

$D \to a \mid b$

Eliminate Null productions

$S \to bAB \mid bb \mid C \mid bB \mid bA \mid b$

$A \to BC \mid aCB \mid a \mid C \mid aC$

$B \to bB \mid C \mid b$

$C \to aaC \mid bbC \mid D$

$D \to a \mid b$

Eliminate the Unit productions

$S \to bAB \mid bb \mid aaC \mid bbC \mid a \mid b \mid bB \mid bA \mid b$

$A \to BC \mid aCB \mid a \mid aaC \mid bbC \mid a \mid b \mid aC$

$B \to bB \mid aaC \mid bbC \mid a \mid b \mid b$

$C \to aaC \mid bbC \mid a \mid b$

$D \to a \mid b$

Eliminate Useless Symbols

$S \to bAB \mid bb \mid aaC \mid bbC \mid a \mid b \mid bB \mid bA \mid b$

$A \rightarrow BC \mid aCB \mid a \mid aaC \mid bbC \mid a \mid b \mid aC$

$B \rightarrow bB \mid aaC \mid bbC \mid a \mid b \mid b$

$C \rightarrow aaC \mid bbC \mid a \mid b$

26. (B)

Type checking is done in semantic analysis phase after syntax analysis phase (i.e., after parsing).

27. (A)

Since \$ will be evaluated before # so \$ has higher precedence and the left \$ i.e., in $b\$c\$d$ the left "\$" (i.e., $b\$c$) will be evaluated first so it is left associative, whereas # is right associative (as in $d\#e\#f$), the right one (i.e., e # f) will be evaluated first.

28. (A)

+ has highest precedence, so it will be evaluated first.

$$2-5+1-7*3 = 2-(5+1)-7*3$$
$$= 2-6-7*3$$

Now, - has more precedence than *, so sub will be evaluated before * and – has right associative so (6-7) will be evaluated first.

$$2-6-7*3 = (2-(6-7))*3$$
$$= (2-(-1))*3$$
$$= 3*3 = 9$$

29. (D)

The statement, static allocation of all data areas by a compiler makes it impossible to implement recursion is true, as recursion requires memory allocation at run time, so it requires dynamic allocation of memory.

Hence, Dynamic allocation of activation records is essential to implement recursion is also a true statement.

30. (A)

FIRST(P): is the set of terminals that begin the strings derivable from non-terminal P. If P derives epsilon, then we include epsilon in FIRST(P).

FOLLOW(P): is the set of terminals that can appear immediately to the right of P in some sentential form.

FIRST(A) = FIRST(S)

FIRST(S) = FIRST($aAbB$) and FIRST($bAaB$) and FIRST ($\in$)

FIRST(S) $= \{a, b, \in\}$

FIRST(B) = FIRST(S) $= \{a, b, \in\} =$ FIRST(A)

FOLLOW(A) $= \{b\}$ // because of production $S \rightarrow a\,A\,b\,B$

FOLLOW(A) $= \{a\}$ // because of production $S \rightarrow b\,A\,a\,B$

So, FOLLOW(A) $= \{a, b\}$

FOLLOW(B) = FOLLOW(S) // because of production $S \rightarrow a\,A\,b\,B$

FOLLOW(S) = FOLLOW(A) // because of production $S \rightarrow A$

So, FOLLOW(S) $= \{\$, a, b\} =$ FOLLOW(B)

31. (C)

The entries in E_1, E_2 and E_3 is related to S and B, so we have to take only those production which have S and B in LHS.

$$S \rightarrow aAbB \mid bAaB \mid \varepsilon$$

The production $S \rightarrow aAbB$ will go under column

FIRST($aAbB$) $= a$, so $S \rightarrow aAbB$ will be in E_1.

$S \rightarrow bAaB$ will go under column

FIRST($bAaB$) $= b$, so $S \rightarrow bAaB$ will be in E_2.

$S \rightarrow \varepsilon$ will go under

$\text{FOLLOW}(S) = \text{FOLLOW}(B) = \{a, b, \$\}$, So

$S \to \varepsilon$ will go in E_1, E_2 and under column of \$.

So E_1 will have: $S \to aAbB$ and $S \to \varepsilon$.

E_2 will have $S \to bAaB$ and $S \to \varepsilon$.

Now, $B \to S$ will go under

$\text{FIRST}(S) = \{a, b, \varepsilon\}$

Since $\text{FIRST}(S) = \varepsilon$ So $B \to S$ will go under $\text{FOLLOW}(B) = \{a, b, \$\}$

So E_3 will contain $B \to S$.

32. (B)

Here a, b, and c all have 3 different values so we need at least 3 registers r1, r2 and r3.
Assume 'a' is mapped to r1, 'b' to r2 and 'c' to 13.
d = a – b, after this line if u notice 'a' is never present on right hand side, so we can map 'd' to r1.
e = c + d, after this line 'd' is never present on rhs, so we can map 'e' to r1.
at this time mapping is

r1________e
r2________b
r3________c

We have 3 registers for a, b and c.

f = c - e
b = c + e

These two are essentially doing same thing, after these two line 'b' and T are same so we can skip computing 'f' or need not give any new register for 'i'. And wherever 'f' is present we can replace it with 'b', because neither of 'f' and 'b' are changing after these two lines, so value of these will be "c + e' till the end of the program.
At second last line "d = 5 - e"
Here 'd' is introduced, we can map it to any of the register r1 or r3, because after this line neither of 'e' or ' c' is required. Value of 'b' is required because we need to return 'd + f', and 'f' is essentially equal to 'b'
finally, code becomes

r1 = 1
r2 = 10
r3 = 20
r1 = r1 + r2
r1 = r3 + r1
r2 = r3 + r1
r2 = r3 + r1
r1 = r2 + r2
r3 = 5 + r1
return r3 + r2

Therefore minimum 3 registers needed.

33. (B)

Statement II is false, as a programming language which allows recursion requires dynamic storage allocation. Statement III is false, as L-attributed definition (assume for instance the L-attributed definition has synthesized attribute only) can be evaluated in bottom-up framework.
Statement I is true, as the bottom-up and top-down parser take O (n) time to parse the string, i.e. only one scan of input is required.
Statement IV is true, Code improving transformations can be performed at both source language and intermediate code level. For example implicit type casting is also a kind of code improvement: which is done during semantic analysis phase and intermediate code optimization is a topic itself which uses various techniques to improve the code such as loop unrolling, loop invariant.

34. (B)

(after removing left recursion)

$S \to S'$

$S' \to S + S' / S * S' / a / \in$

$FIRST(S) = \{a, \in\}$

$FIRST(S') = \{a, \in\}$

$FOLLOW(S) = \{+, *\}$

$FOLLOW(S') = \{+, *\}$

35. (A)

S_1 is correct

With triple optimization we cannot change the execution order but with indirect triple we can.

36. (C)

x is inherited

y is synthesized.

37. (C)

Lexical analyzer → Syntax analyzer → Semantic analyzer → intermediate code → Code optimizer.

38. 34

main ( )
1 2 3

{
4

int temp = 200, 10 ;
5 6 7 8 9 10 11

int $l1$, $l2$;
12 13 14 15 16

temp = ++ temp ;
17 18 19 20 21

$l1$ + = $l2$
22 23 24

printf("%d", temp + $l1$) ;
25 26 27 28 29 30 31 32 33

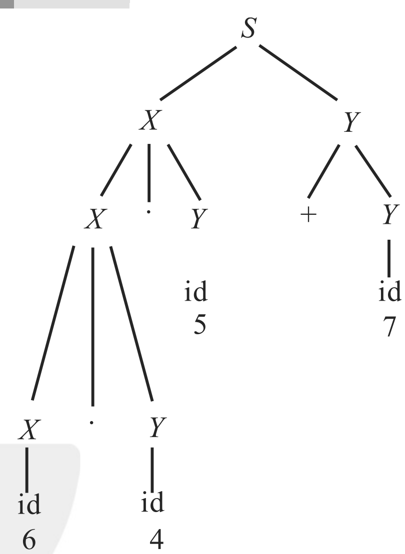
}
34

39. (D)

$\underbrace{ca}_{2}b\ \underbrace{ac}_{1}\ \underbrace{cc}_{3}\quad \underbrace{ab}_{\text{not generated}}$

Hence, lexical error will generate.

40. (C)



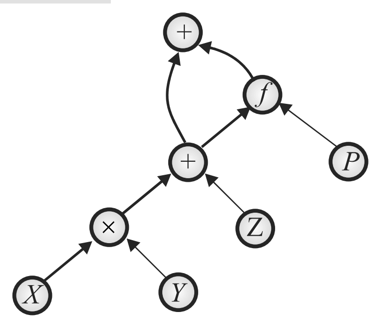
Output: $64*5*7-$

41. (D)

Statement S_1 and S_2 are correct

Statement S_3 in incorrect. Heap and stack both are present in main memory

42. 8



Total 8 edges

43. 2

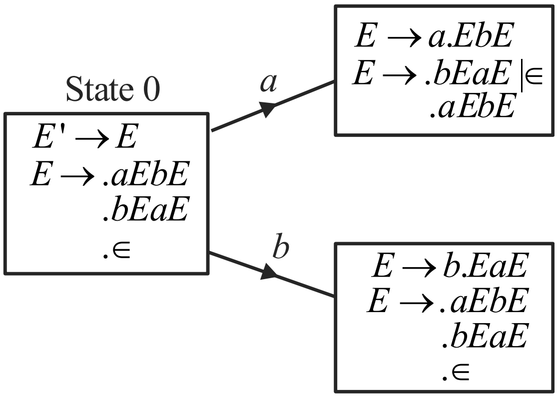
Statement I and IV is correct

Type checking is done at semantic analysis phase

Target code generation is dependent based on the machine

Symbol table is accessed during lexical, syntax and semantic analysis phase.

44. (A)



In state 0, there is reduce $E \to \in$ which will go under of 'E' which $\{a, b\}$ and also at state 0,

There is shift at 'a' and 'b'. Hence, there is shift reduce conflict.

45. (D)

$S \to aabc \mid ab$

There is left factoring in LL (1). Hence, not LL (1) but it is LL (2).

Every regular language is LL (1) is true. There exist a regular grammar which is LL (1).

Every regular grammar is LL (1) is false, because regular grammar may contain left recursion, left factoring ambiguity.

46. (D)

A grammar G is said to be operator grammar if

(a) it does not contain null production

(b) it does not contain 2 adjacent variables on right hand side

So, both G_1 and G_2 are not operator grammar.

47. (B)

To go from CLR(1) parsing table to LALR(1) parsing table, we merge the states that have the same final items but different lookaheads.

In doing so, we can only introduce RR conflicts.

48. (D)

Handle is part of the string in sentential form that will be reduced to non-terminal i.e left hand side of a production

In the above derivation, sentential form $Aabw$ is reduced to ABw so has to be a production with $B \to ab$ and that is the handle at this point of derivation.

49. (C)

1. In a SLR(1) parser, it is allowable for both shift and reduce items to be in the same state even though it leads to sr conflict but it is allow

2. In a SLR(1) parser, it is allowable for multiple reduce items to be in the same state even though it leads to sr conflict but it is allow

3. All SLR(1) grammars are LR(0) this statement is wrong Reason is $\mathrm{LR}(0) < \mathrm{SLR}(1) < \mathrm{LALR}(1) < \mathrm{CLR}(1)$

→ If a grammar is LR(0) then it is also SLR(1), LALR(1), CLR(1).

→ If a grammar is SLR(1) then it is also LALR(1), CLR(1).

→ If a grammar is LALR(1) then it is also CLR(1).

4. **All LR(0) grammars are SLR(1)**

Therefore C is incorrect.

50. (D)

Since LR(0) parser places reduce-moves in the entire row of "Action", having anything more than just the reduce-move in the state having final-item would lead to SR or RR conflict.

So, Options A, B, and C are true.

51. (A)

A top-down parse produce a leftmost derivation of a sentence.

A bottom-up parse produces a rightmost derivation of a sentence but in reverse.

52. (D)

LL(0) grammars have no lookhead. And since they follow Leftmost derivation, at each step the parser has to derive the string by seeing 0 symbols $\Rightarrow$ Parser sees nothing.

So whenever we have multiple choices for any Variable in the grammar, LL(0) fails.

Hence, LL(0) parser can only parse grammars that strictly generate one single string.

53. (A)

LL(1) is a top-down or predictive parser

REC Decent is predictive

REC Ascent is a technique for implementing an LALR parser so Bottom up

LR(1) is bottom-up or DCFL, since LR(K) accepts DCFL

54. (C)

$$\in\text{-free LL(1)} \subset \text{SLR(1)} \subset \text{LR(1)}$$

Because every $\in$ –free LL(1) are SLR(1) and every SLR(1) are LR(1)

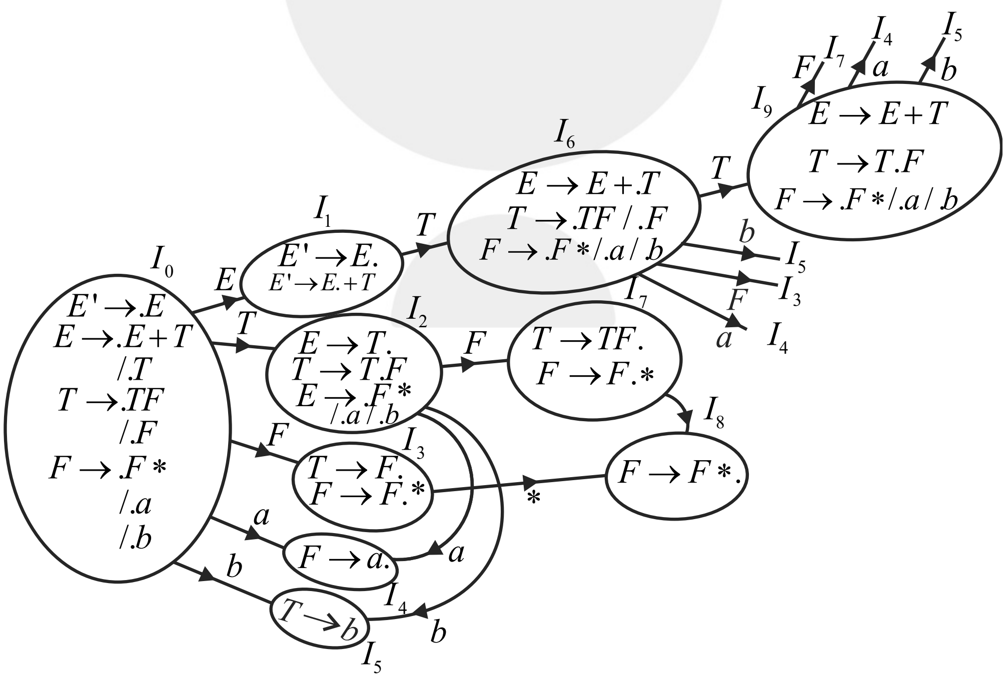
55. (C)

In operator relation table if we have 4 operator then we got is 16 cells so if we have n operator then n^2 is table size.

But in operator function table if we have 4 operators then we got 8 cells so if we have n operator then we got $2n$ table size.

So the option C is the correct option

56. 10



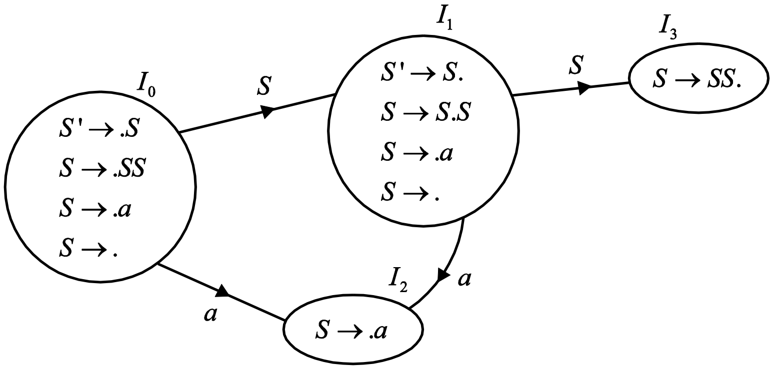
So total 10 states are required

57. (B)

RR conflict:- Means reduce reduce conflict, that means a single state have more than one final production.

SR conflict means in a state there is a final production and here shift more are also occurred. So we find RR & SR conflict in our LR (0) item

$S \to SS \mid a \mid \in$



In the above LR (0) item I_0 has a final item and it shift to I_2 So I_0 have one SR conflict

in the state I_1 has two Final item so here RR conflict occurred and in I_1 have shift move also, it moves

to I_2 So it's have one SR conflict state

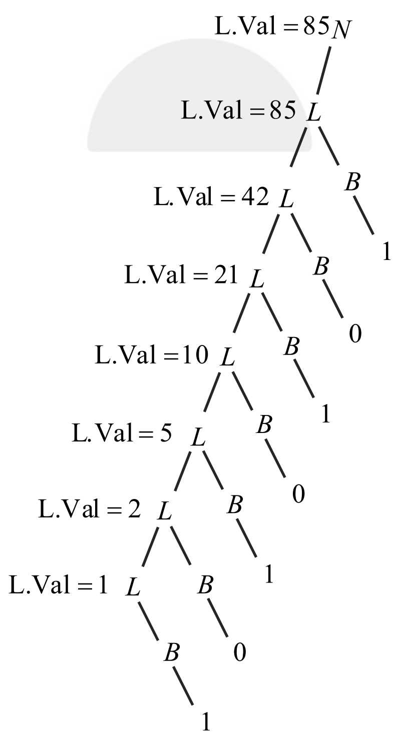
I_2 & I_3 have no conflict so total SR conflicts are 2 (1 in I_0 & 1 in I_1)

And RR conflicts are only 1 (in I_1)

So the option B is correct option.

58. **85**

So it is a logic of binary to decimal conversion



Finally N.val is 85 so the answer is 85