CD: Mid-2 QB

1. Which is the most powerful parsing mechanism?

A. Canonical Parsing

2. A bottom up parser generates
    a) Right most derivation
    b) Right most derivation in reverse
    c) Left most derivation
    d) Left most derivation in reverse

3. A shift reduce parser carries out the actions specified within braces immediately after reducing with the corresponding rule of grammer

S----> xxW (PRINT "1")

S----> y { print " 2 " }

W----> Sz { print " 3 ")

What is the translation of xxxxyzz using the syntax directed translation scheme described by the above rules ?

A. 23131

4. In which phase normally Type checking is performed?

A. Semantic Analysis

5. What are the different intermediate code forms?

A. Syntax tree,DAG, Postfix notation, Three address code (Quadraple,Triple and Indirect Triple Notations)

6. What is a synthesized attribute and inherited attribute?

A. **Synthesized Attribute:** If the Value of an Node is Calculated from it's Cchildren then it is called Synthesized Attribute.

Inherited Attribute: If the Value of a Node is Calculated from its Parent or Siblings then it is called Inherited Attribute

7. What is the difference between SDD and SDT?

SDD: Specifies the values of attributes by associating semantic rules with the productions.

SDT: Embeds program fragments (also called semantic actions) within production bodies. The position of the action defines the order in which the action is executed (in the middle of production or end).

SDD: It is easier to read; easy for specification.

SDT: It can be more efficient; easy for implementation

8. What is S-attribute definition and L-attribute definition?

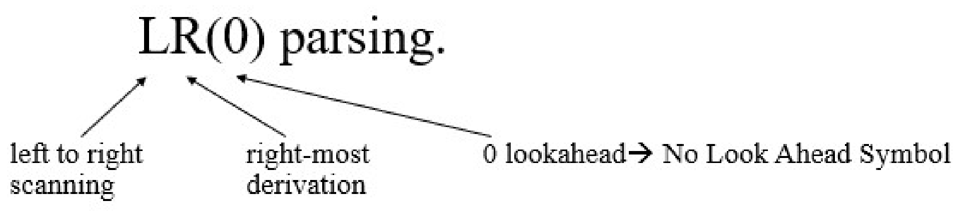
A. Syntax-directed definition that uses only synthesized attributes is called an S-Dttributed Definition.
Syntax-directed definition that uses both synthesized attributes and Inherited attributes but in Inherited the value of a node is Calculated from Either parent or left sibling is called L-Attribute Definition.

9. What is a handle?

A. A **handle** is a substring of a string that matches the right side of a production rule.

10. What is meant by L, R, 0 in LR(0)?

A.



11. What are the possible conflicts in SLR(1)?

A. Shift/Reduce Conflict and Reduce/Reduce Conflict

12. IF the base production is in the form of A→α.Bβ, a the Look Ahead of the derived production B→.γ is First(β,a).

13. 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 *

14. Consider the grammar with the following translation rules and E as the start symbol.

E -> E1 #T {E.value = E1.value * T.value}

| T {E.value = T.value}

T -> T1 & F {T.value = T1.value + F.value}

|F {T.value= F.value}

F -> num {F.value = num.value}

Compute E.value for the root of the parse tree for the expression:2 # 3 & 5 # 6 &4

A. 160

15. Generate the three address code for an expression x: = a + b * c +d ;

A. t1=a+b;
t2=c+d;
t3=t1*t2;
x=t4;

16. Which bottom up parser supports even ambiguous grammars also?

A. Operatr Precedence Parser

17. What is a augmented grammar?

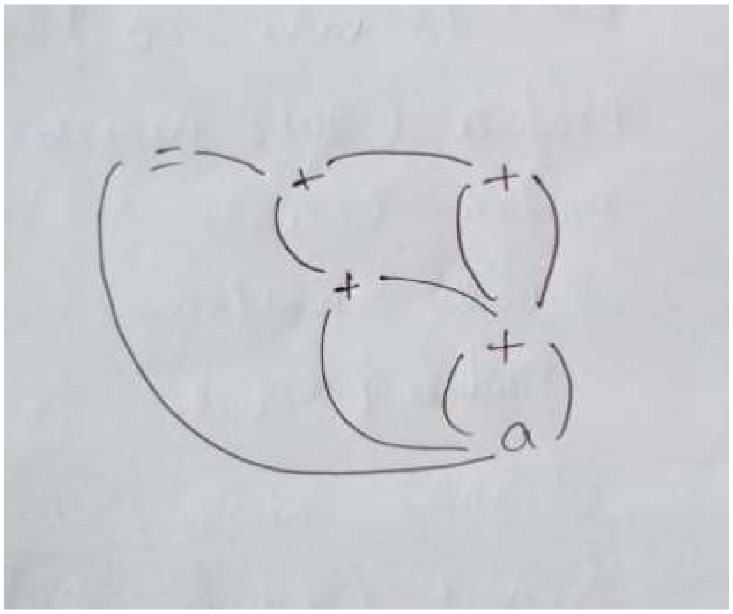
A. A Grammar Containing Augmented Production is called Augmented Grammar. Augmented Production is nothing but a new production, which is not already present in the grammar that derives the start symbol.

18. What is the parsing table size of LR (0) items?

A. (Number of Terminals + Number of Non terminals + 1) * (Number of States)

19. DAG stands for Directed Acyclic Graph.

20. Construct DAG for a:=a+(a+a)+(a+a+a+a)



21. What a operator grammar and operator precedence grammar?

A. Operator Grammar is a Grammar that doen't have any Epsilon productions on Right-side and no two nonterminal should be a side.

Operator Precedence Grammar is an Operator Grammar in which Precedence and Associativity rules are clearly Specified without any Ambiguity.

22. Consider the grammar G whose CLR parser has n1 states and LALR parser has n2 states. What is the relation between n1 and n2?

(A) n1 ≤n2

(B) n1 = n2

(C) n1 ≥n2

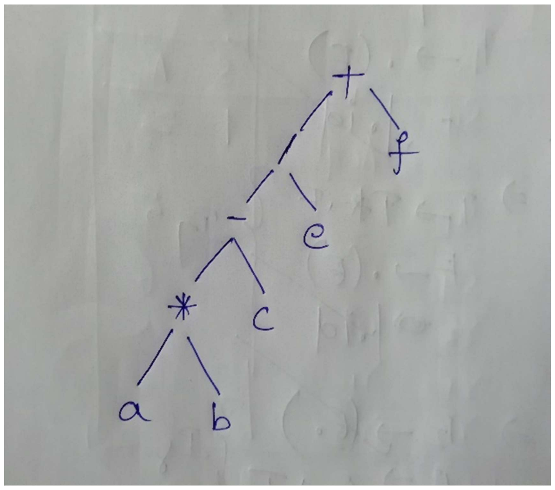
(D) None of the above

23. Define annotated parse tree?

A. A Parse Tree in which each node is Represented by it's value is called Annotated Parse Tree

24. Generate syntax tree for following expression :

a*b-c/e+f



25. Regarding the LR(0) and SLR(1) parse tables on which entries we may have the variation

a. Only shift entries

b. Both shift and reduced entries

c. only reduced entries

d. both reduced and error entries.

26. What is precedence and associatively?

A. Precedence determines which operator is performed first in an expression with more than one operator and Associativity determines which operator is performed first when two operators of same precedence appear in an expression.

27. YACC stand for Yet Another Compiler Compiler.

28. Why Bottom-up parsers called as Shift Reduce parsers?

A. Bottom-up parsing is also known as **shift-reduce parsing,** because its two main actions are shift and reduce.

29. In operator precedence parsing table when will be the 2 operators will get equal precedence i.e. a=b.

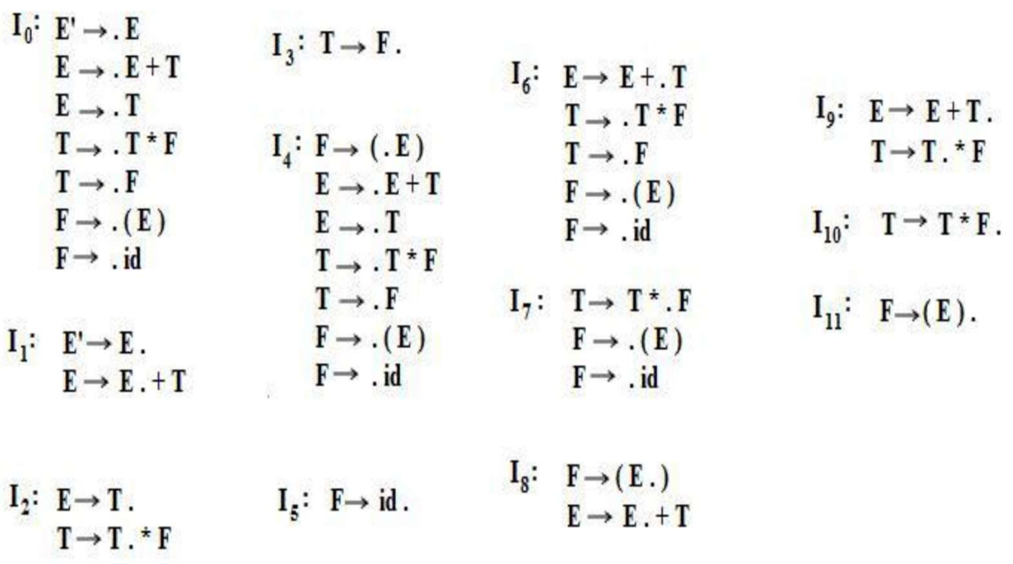
A. When two operators are at same level in the Operator Precedence Grammar then they get equal Precedence.

30. Which conflict is produced by a state I_k having $A \rightarrow \alpha$. And $B \rightarrow \gamma$.

A. Reduce -Reduce Conflict

TEN MARK QUESTIONS

1. Construct SLR(1) Parsing table for the following grammar E->E+T| T; T->T*F |F; F->(E) | id and show the parsing action for the string id +id +id.



I_0: $E' \rightarrow .E$
$E \rightarrow .E+T$
$E \rightarrow .T$
$T \rightarrow .T*F$
$T \rightarrow .F$
$F \rightarrow .(E)$
$F \rightarrow .id$

I_1: $E' \rightarrow E.$
$E \rightarrow E.+T$

I_2: $E \rightarrow T.$
$T \rightarrow T.*F$

I_3: $T \rightarrow F.$

I_4: $F \rightarrow (.E)$
$E \rightarrow .E+T$
$E \rightarrow .T$
$T \rightarrow .T*F$
$T \rightarrow .F$
$F \rightarrow .(E)$
$F \rightarrow .id$

I_5: $F \rightarrow id.$

I_6: $E \rightarrow E+.T$
$T \rightarrow .T*F$
$T \rightarrow .F$
$F \rightarrow .(E)$
$F \rightarrow .id$

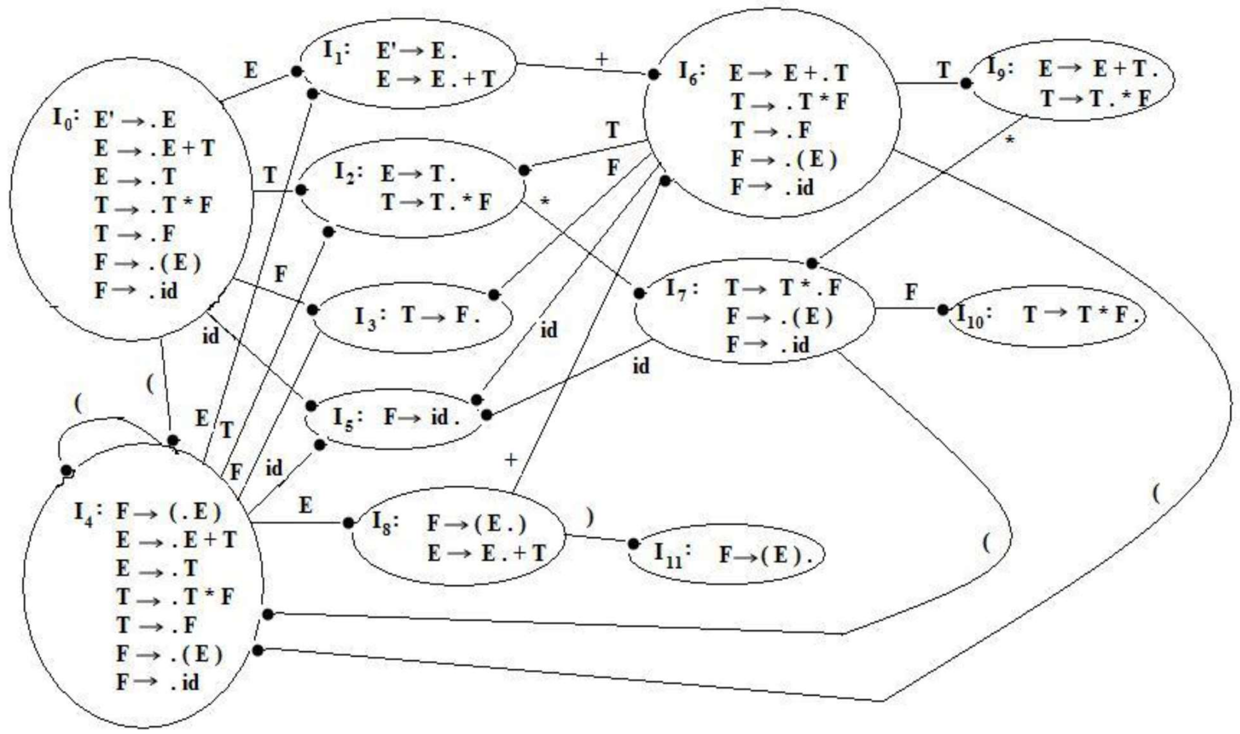
I_7: $T \rightarrow T*.F$
$F \rightarrow .(E)$
$F \rightarrow .id$

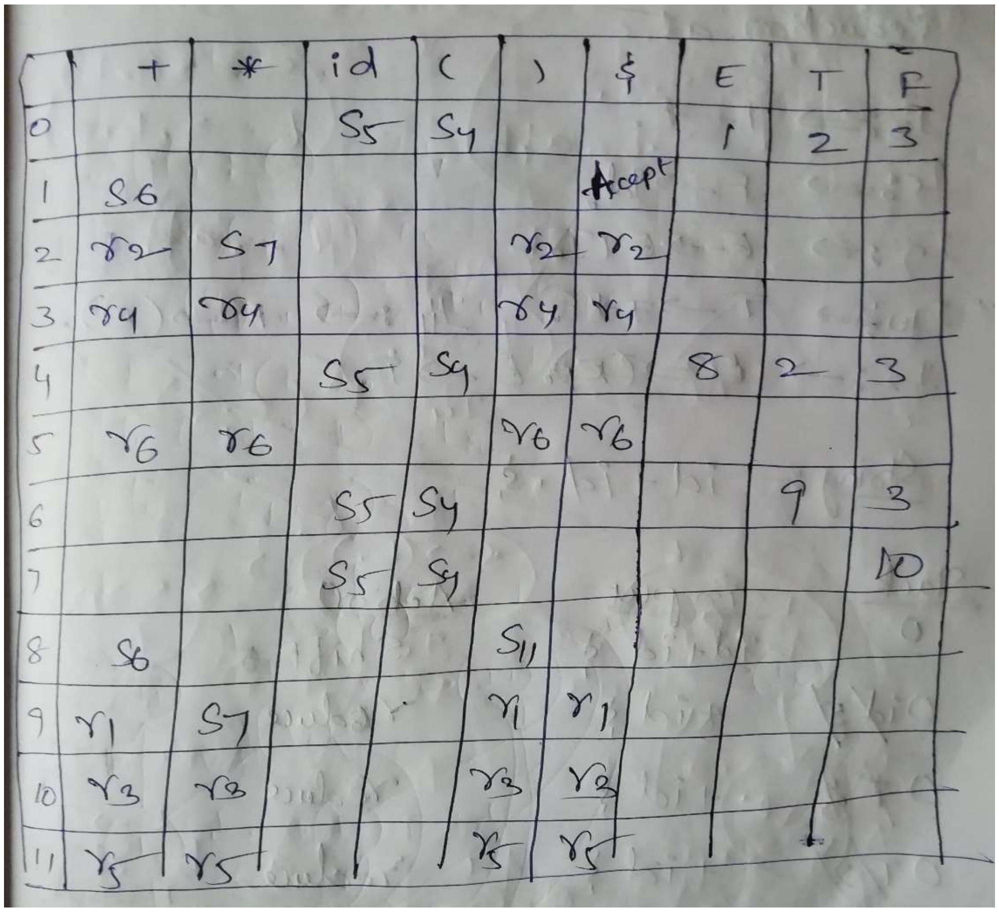
I_8: $F \rightarrow (E.)$
$E \rightarrow E.+T$

I_9: $E \rightarrow E+T.$
$T \rightarrow T.*F$

I_{10}: $T \rightarrow T*F.$

I_{11}: $F \rightarrow (E).$





	+	*	id	(	)	$	E	T	F
0			S5	S4			1	2	3
1	S6					Accept			
2	r2	S7			r2	r2			
3	r4	r4			r4	r4			
4			S5	S4			8	2	3
5	r6	r6			r6	r6			
6			S5	S4				9	3
7			S5	S4					10
8	S6				S11				
9	r1	S7			r1	r1			
10	r3	r3			r3	r3			
11	r5	r5			r5	r5			

Derive the String id+id+id;

2. Construct a canonical parsing table for the grammar given below

S -> CC

C -> cC | d

I_0 : $S' \rightarrow .\,S, \$$
$S \rightarrow .\,CC, \$$
$C \rightarrow .\,cC, c/d$
$C \rightarrow .\,d, c/d$

I_1: $S' \rightarrow S\,., \$$

I_2 : $S \rightarrow C\,.\,C, \$$
$C \rightarrow .\,cC, \$$
$C \rightarrow .\,d, \$$

I_3 : $C \rightarrow c\,.\,C, c/d$
$C \rightarrow .\,cC, c/d$
$C \rightarrow .\,d, c/d$

I_4 : $C \rightarrow d\,., c/d$

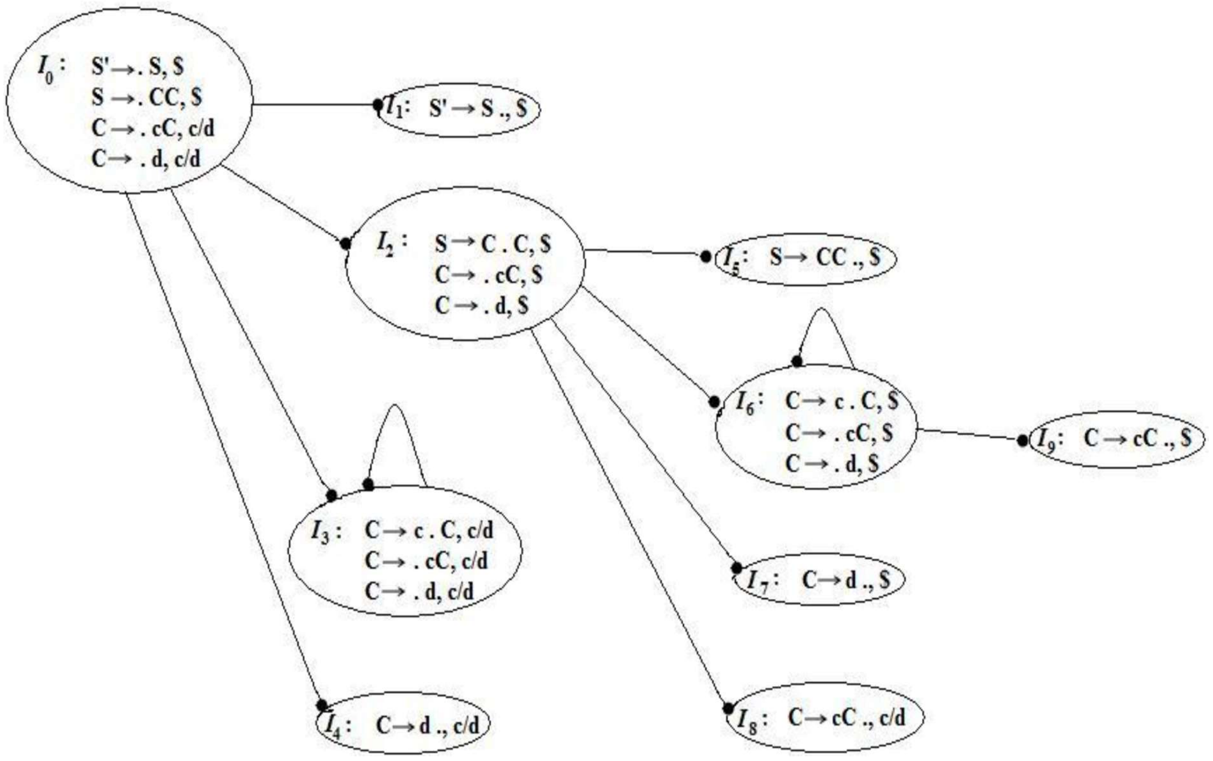
I_5: $S \rightarrow CC\,., \$$

I_6 : $C \rightarrow c\,.\,C, \$$
$C \rightarrow .\,cC, \$$
$C \rightarrow .\,d, \$$

I_7: $C \rightarrow d\,., \$$

I_8: $C \rightarrow cC\,., c/d$

I_9: $C \rightarrow cC\,., \$$



I0: S'→. S, $ / S →. CC, $ / C→. cC, c/d / C→ . d, c/d
I1: S'→S ., $
I2: S→C . C, $ / C→ . cC, $ / C→ . d, $
I5: S→ CC ., $
I6: C→ c . C, $ / C→ . cC, $ / C→ . d, $
I9: C→cC ., $
I3: C→ c . C, c/d / C→ . cC, c/d / C→ . d, c/d
I7: C→d ., $
I4: C→d ., c/d
I8: C→cC ., c/d

3. Consider the grammar

   S-> L=R
   S->R
   L->*R
   L->id
   R->L
   using operator precedence parser, parse the string id= id * id and also find out the function table.

4. Construct Quadruples, Triples, and Indirect Triples of the following expression:
   (a + b) * (c + d) - (a + b + c).

5. Discuss about
   i) Postfix Notation 3M
   ii) Syntax Trees. 3M
   iii) Generate three address code for the following code construct. 6M
   a) **a < b or c < d**
   b) for(i=0;i<10;i++)
      a=a+10