Compiler Design
Chapter 2: Parsing

GATE CS PYQ
by Monalisa

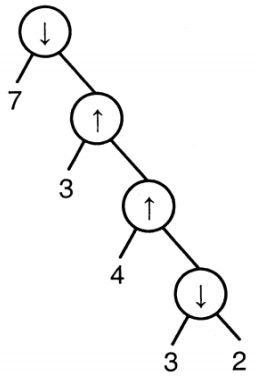
- GATE CS 2010,Q38:The grammar S → aSa | bS | c is
  (A) LL(1) but not LR(1)
  (B) LR(1)but not LR(1)
  (C) Both LL(1)and LR(1)
  (D) Neither LL(1)nor LR(1)
- The LL(1) parsing table for the given grammar is:

	a	**b**	**c**	**$**
S	S → aSa	S → bS	S → c	

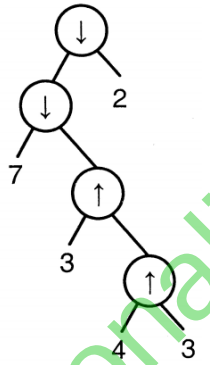
- As there is no multiple entries in LL(1) parsing table, the given grammar is LL(1).
- Every LL(1) is LR(1) also, so the given grammar is LL(1) as well as LR(1).
- Ans : **(C)** Both LL(1)and LR(1)

- GATE CS 2011,Q27: 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)?

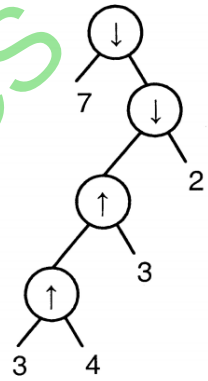
(A)



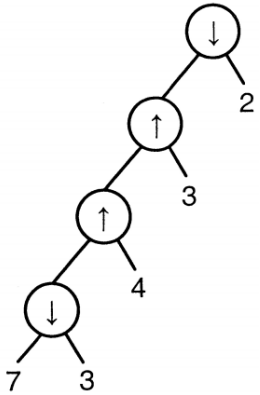
(B)



(C)



(D)



- ↓: left associative, ↑:right associative
- 7 ↓ 3 ↑ 4 ↑ 3 ↓ 2
- = (7 ↓ (3 ↑ (4 ↑ 3))) ↓ 2
- Ans: B

- GATE CS 2012,Q52: 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 **E1**, **E2**, and **E3**. ε is the empty string, $ indicates end of input, and,| separates alternate right hand sides of productions.

S → a A b B | b A a B | ε
A → S
B → S

	a	b	$
S	**E1**	**E2**	S → ε
A	A → S	A → S	error
B	B → S	B → S	**E3**

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

(A)	(B)	(C)	(D)
FIRST(A) ={a, b,ε} = FIRST(B) FOLLOW(A) = {a, b} FOLLOW(B) = {a, b, $}	FIRST(A) ={a, b, $} FIRST(B) = {a, b, ε} FOLLOW(A) = {a, b} FOLLOW(B) = {$}	FIRST(A) ={a, b, ε} = FIRST(B) FOLLOW(A) = {a, b} FOLLOW(B) = ∅	FIRST(A) ={a, b} = FIRST(B) FOLLOW(A) = {a, b} FOLLOW(B) = {a, b}

	FIRST	**FOLLOW**
S	a,b, ε	a,b,$
A	a,b, ε	a,b
B	a,b, ε	a,b,$

- Ans:A

- GATE CS 2012,Q53: 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 **E1**, **E2**, and **E3**. ε is the empty string, $ indicates end of input, and,| separates alternate right hand sides of productions.

S → a A b B | b A a B | ε
A → S
B → S

	a	b	$
S	**E1**	**E2**	S → ε
A	A → S	A → S	error
B	B → S	B → S	**E3**

- The appropriate entries for E1, E2, and E3 are

(A) E1:S →aAbB,A→S
E2:S →bAaB,B→S
E3: B→S

(B) E1:S →aAbB,S→ε
E2:S →bAaB,S→ε
E3:S →ε

(C) E1:S→aAbB, S→ε
E2: S→bAaB, S→ε
E3: B→S

(D) E1: A →S, S →ε
E2: B →S, S →ε
E3: B →S

- S→ aAbB will go under column a.
- S→ bAaB will go under column b.
- S→ ε will go in E1, E2 and under column of $.
- E1 will have: S→ aAbB and S→ ε.
- E2 will have S→ bAaB and S→ ε.
- B→ S will go under {a, b, $}
- So E3 will contain B→ S.
- Ans: C

	FIRST	FOLLOW
S	a,b, ε	a,b,$
A	a,b, ε	a,b
B	a,b, ε	a,b,$

- GATE CS 2013,Q9: What is the maximum number of reduce moves that can be taken by a bottom-up parser for a grammar with no epsilon- and unit-production (i.e., of type A → ϵ and A → a) to parse a string with *n* tokens?
- (A) *n*/2 (B) *n*-1 (C) 2*n*-1 (D) 2^n
- Maximum number of reduce moves that can be taken by Shift reduce parser / bottom up parser for a grammar with no ∈ and unit production to parse a string of n token is n-1.
- Ans : (B) n-1

- GATE CS 2013,Q40: Consider the following two sets of LR(1) items of an LR(1) grammar.

  X → c.X, c/d X → c.X, $
  X → .cX, c/d X → .cX, $
  X → .d, c/d X → .d, $

- Which of the following statements related to merging of the two sets in the corresponding LALR parser is/are **FALSE**?
- 1. Cannot be merged since look aheads are different.
  2. Can be merged but will result in S-R conflict.
  3. Can be merged but will result in R-R conflict.
  4. Cannot be merged since *goto* on *c* will lead to two different sets.
- (A) 1 only (B) 2 only (C) 1 and 4 only (D) 1, 2, 3 and 4
- 1.Two sets in LR(1) items can be merged if they differ with look ahead symbols,1 false.
- 2.No reduction, so after merging it will not have SR conflict,2 false.
- 3.No reduction , so after merging it will not have RR conflict,3 false
- 4.Mearging possible, goto(c) will lead to one state.4 false
- Ans: (D) 1, 2, 3 and 4

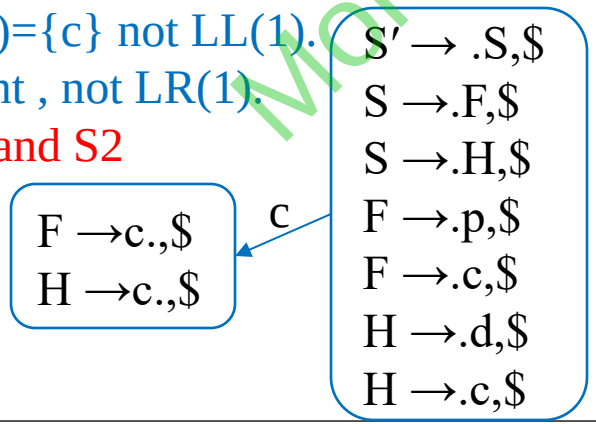
- GATE CS 2014 Set-1,Q34: A canonical set of items is given below
- S → L.>R
- Q → R.
- On input symbol < the set has
  (A) a shift-reduce conflict and a reduce-reduce conflict.
  (B) a shift-reduce conflict but not a reduce-reduce conflict.
  (C) a reduce-reduce conflict but not a shift-reduce conflict.
  (D) neither a shift-reduce nor a reduce-reduce conflict.
- The input symbol is "<" which is not in canonical set of item, so it is neither a shift-reduce nor a reduce-reduce conflict with reference to "<" symbol.
- if symbol ">" then it will be a SR conflict.
- **Ans:(D)** neither a shift-reduce nor a reduce-reduce conflict.

- GATE CS 2014 Set-2,Q17: Consider the grammar defined by the following production rules, with two operators ∗ and +
- S → T * P
- T → U | T * U
- P → Q + P | Q
- Q → Id
- U → Id
- Which one of the following is TRUE?
- **(A)** + is left associative, while ∗ is right associative
  (B) + is right associative, while ∗ is left associative
  (C) Both + and ∗ are right associative
  (D) Both + and ∗ are left associative
- T ⟶ T * U, since T is left recursive, hence * is left associative.
- P ⟶ Q + P, here P is right recursive, so + is right associative.
- **Ans: (B)** + is right associative, while ∗ is left associative

- GATE CS 2015 Set-1,Q13: Which one of the following is True at any valid state in shift-reduce parsing?
  (A) Viable prefixes appear only at the bottom of the stack and not inside
  (B) Viable prefixes appear only at the top of the stack and not inside
  (C) The stack contains only a set of viable prefixes
  (D) The stack never contains viable prefixes
- A handle is actually on the top of the stack.
- The prefixes of right sentential forms that can appear on the stack of a shift reduce parser are called *viable prefixes*.
- So set of viable prefixes is in stack.
- **Ans : (C)** The stack contains only a set of viable prefixes

- GATE CS 2015 Set-3,Q16: Among simple LR (SLR), canonical LR, and look-ahead LR (LALR), which of the following pairs identify the method that is very easy to implement and the method that is the most powerful, in that order?
- **(A)** SLR, LALR
  (B) Canonical LR, LALR
  (C) SLR, canonical LR
  (D) LALR, canonical LR
- SLR is very easy to implement and CLR is most powerful method.
- **Ans : (C)** SLR, canonical LR

- GATE CS 2015 Set-3,Q31: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
- The given grammar is ambiguous as there are two possible parse tree for string "c".
- An Ambiguous grammar can neither be LL(1) nor LR(1).
- FIRST(F)∩FIRST(H)={c} not LL(1).
- R-R conflict is present , not LR(1).
- Ans: **(D)** Neither S1 and S2



- GATE CS 2017 Set-1,Q17:Consider the following grammar:
- P→xQRS
- Q→yz|z
- R→w|ε
- S→y
- Which is FOLLOW(Q)?
- **(A)** {R} **(B)** {w} **(C)** {w, y} **(D)** {w, $}
- FOLLOW (Q) = FIRST (R)
- FIRST (R) = {w, ϵ}
- Since FIRST (R) = {ϵ}
- So FOLLOW (Q) = {w} ∪ FIRST(S)
- FIRST(S) = {y}
- FOLLOW (Q) = {w, y}
- Ans : **(C)** {w, y}

- GATE CS 2017 Set-2,Q6 :Which of the following statements about parser is/are CORRECT?
- I. Canonical LR is more powerful than SLR.
  II. SLR is more powerful than LALR.
  III. SLR is more powerful than Canonical LR.
- **(A)** I only **(B)** II only **(C)** III only **(D)** II and III only
- The power in increasing order is: SLR < LALR < CLR
- Ans: **(A)** I only

- GATE CS 2017 Set-2,Q32: Consider the following expression grammar G:
- E → E - T | T
  T → T + F | F
  F → (E) | id
- Which of the following grammars is not left recursive, but is equivalent to G?

(A) E→E-T |T
T→ T+F |F
F → (E) | id

(B) E → TE'
E' → -TE' | ε
T → T + F | F
F → (E) | id

(C) E → TX
X → -TX | ε
T → FY
Y → +FY | ε
F → (E) | id

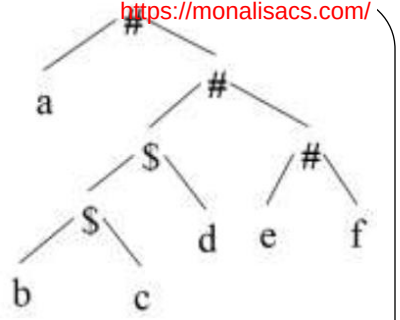
(D) E → TX | (TX)
X → -TX |+TX| ε
T → id

- *LRG*
- $A \rightarrow A\alpha|\beta$
-
- E →TX
- X →-TX| ε
- T →FY
- Y →+FY| ε
- F → (E) | id

RRG
$\Rightarrow A \rightarrow \beta A'$
$\Rightarrow A' \rightarrow \alpha A'|\varepsilon$

Ans : (C)

- GATE CS 2018,Q38: 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
- Since $ will be evaluated before # so $ has higher precedence .
- Left $ in bcd will be evaluated first so it is left associative.
- # is right associative as the right one (e#f) will be evaluated first.
- $>#,$=left associative,#=right associative.
- Ans : (A) $ has higher precedence and is left associative;
- # is right associative

- GATE CS 2019,Q3: Which one of the following kinds of derivation is used by LR parsers?
  (A) Leftmost
  (B) Leftmost in reverse
  (C) Rightmost
  (D) Rightmost in reverse
- Bottom up parser simulates reverse of right most derivation.
- LR parsers have Rightmost derivation in reverse.
- Ans :**(D)** Rightmost in reverse

- GATE CS 2019,Q19: Consider the grammar given below:
- $S \rightarrow Aa$
- $A \rightarrow BD$
- $B \rightarrow b|\varepsilon$
- $D \rightarrow d|\varepsilon$
- Let a, b, d and $ be indexed as follows:

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).
- Answer_____.
- Follow(B) = First(D) ∪ Follow(A)
- Follow(B) = {d} ∪ {a} = {a,d}
- Ans : 31

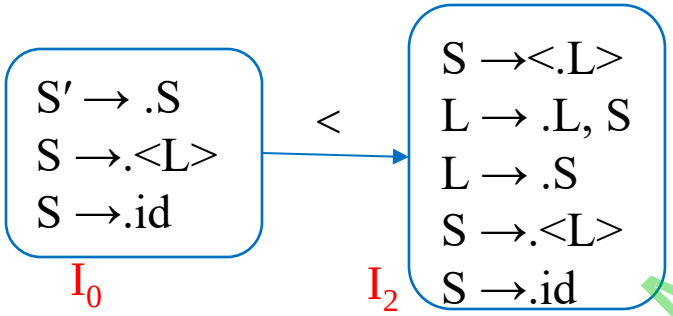
- GATE CS 2019,Q43: Consider the augmented grammar given below:

  $S' \rightarrow S$

  $S \rightarrow <L> | id$

  $L \rightarrow L, S | S$

  Let I_0 = CLOSURE ({[$S' \rightarrow .S$]}). The number of items in the set GOTO (I_0, <) is _____.



- The set GOTO (I_0 , <) has 5 items
- Ans: 5

- GATE CS 2020,Q24: Consider the following grammar.

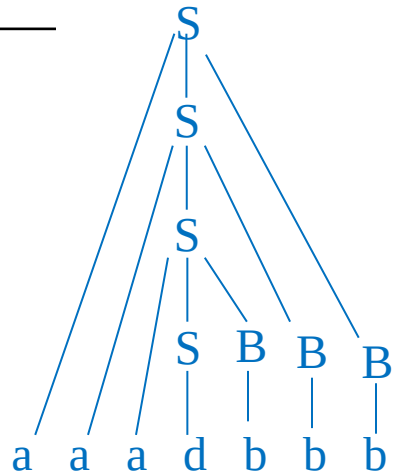
  S → aSB| d

  B → b

  The number of reduction steps taken by a bottom-up parser while accepting the string *aaadbbb* is _____.

- aaadbbb [1.S →d]
- aaaSbbb [2.B →b]
- aaaSBbb [3. S → aSB]
- aaSbb [4. B →b]
- aaSBb [5. S → aSB]
- aSb [6. B →b]
- aSB [7. S → aSB]
- S
- Ans : 7

Stack	i/p string	Action
$	aaadbbb$	Shift
$a	aadbbb$	Shift
$aa	adbbb$	Shift
$aaa	dbbb$	Shift
$aaad	bbb$	Reduce S →d
$aaaS	bbb$	Shift
$aaaSb	bb$	Reduce B →b
$aaaSB	bb$	Reduce S → aSB
$aaS	bb$	Shift
$aaSb	b$	Reduce B →b
$aaSB	b$	Reduce S → aSB
$aS	b$	Shift
$aSb	$	Reduce B →b
$aSB	$	Reduce S → aSB
$S	$	Accept



- GATE CS 2021,Set-1,Q5:Consider the following statements.
- S_1:Every SLR(1) grammar is unambiguous but there are certain unambiguous grammars that are not SLR(1).
- S_2:For any context-free grammar , there is a parser that takes at most $O(n^3)$ time to parse a string of length n.
- Which one of the following options is correct?
- (A) S_1 is true and S_2 is false
- (B) S_1 is false and S_2 is true
- (C) S_1 is true and S_2 is true
- (D) S_1 is false and S_2 is false
- S_1:True
- S_2: Top down & bottom up parser time complexity $O(n^3)$ .True
- Ans: (C) S_1 is true and S_2 is true

- GATE CS 2021,Set-1,Q31:Consider the following context-free grammar where the set of terminal is {*a* ,*b* ,*c* ,*d* ,*f*}
- S →*da*T | R*f*
- T →*a*S | *ba*T | ϵ
- R →*ca*TR | ϵ
- The following is a partially-filled LL(1) parsing table.

	a	***b***	***c***	***d***	***f***	***$***
S			(1)	S →*da*T	(2)	
T	T →*a*S	T →|*ba*T	(3)		T →ϵ	(4)
R			R →*ca*TR		R →ϵ	

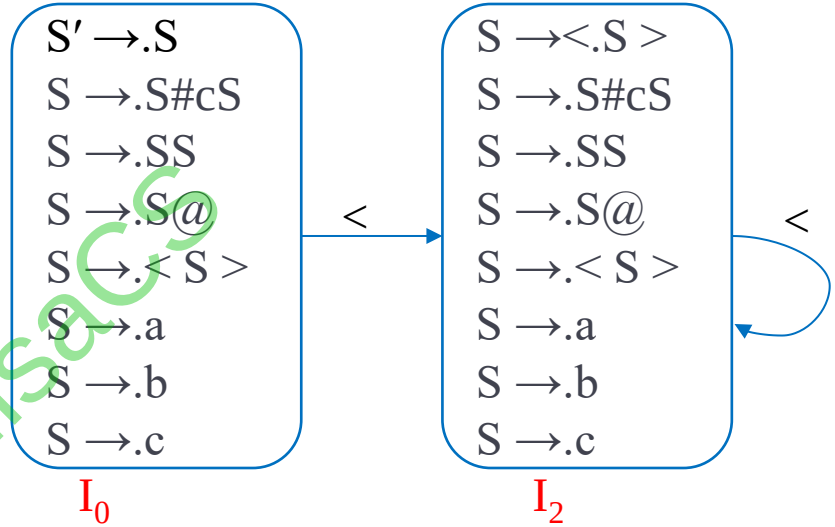
	FIRST	FOLLOW
S	{d,c,f}	{c,f,$}
T	{a,b,ϵ}	{c,f,$}
R	{c,ϵ}	{f}

- (1) S →R*f*
- (2) S →R*f*
- (3) T →ϵ
- (4) T →ϵ

- Which one of the following choices represents the correct combination for the numbered cells in the parsing table("blank" denotes that corresponding cell is empty)?
- (A) (1) S →R*f* (2) S →R*f* (3) T →ϵ (4) T →ϵ
- (B) (1) blank (2) S →R*f* (3) T →ϵ (4) T →ϵ
- (C) (1) S →R*f* (2) blank (3) blank (4) T →ϵ
- (D) (1) blank (2) S →R*f* (3) blank (4) blank

- Ans : (A)

- GATE CS 2021,Set-2,Q51:Consider the following augmented grammar with {#, @, <, >, a, b, c} as the set of terminal.
- S′ → S
- S →S#cS
- S →SS
- S →S@
- S →< S >
- S → a
- S → b
- S → c

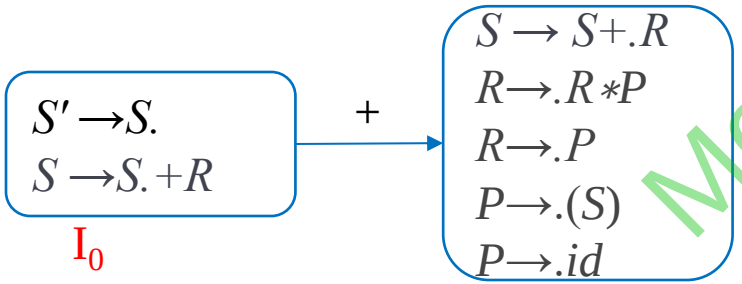


- Let I_0 = CLOSURE ({S′ → .S}). The number of items in the set GOTO (GOTO (I_0,<),<) is _____ .
- Ans : 8

- **GATE CS 2022 | Question: 3**
- Which one of the following statements is TRUE?
- (A)The *LALR*(1) parser for a grammar *G* cannot have reduce-reduce conflict if the *LR*(1) parser for *G* does not have reduce-reduce conflict.
- (B) Symbol table is accessed only during the lexical analysis phase.
- (C) Data flow analysis is necessary for run-time memory management.
- (D)*LR*(1) parsing is sufficient for deterministic context-free languages.
- (A) If LR(1) have RR conflict or may not have RR conflict ,still LALR(1) may have RR Conflict . LALR(1) have SR conflict if and only if LR(1) have SR conflict .False
- (B) It is the abstract data structure use by compiler to store all the information about identifiers used in the program. Every phases of compiler interact with symbol table.
- Whenever an identifier is detected in any of the phases, it is stored in the symbol table. False
- (C) *Data-flow analysis* derive information about the flow of data along program execution paths .Flow graph tells us about the possible execution paths.
- The compiler creates and manages a run-time environment in which it assumes its target programs are being executed. Its not related with data flow analysis .False
- (D) Every LR(k) language is deterministic context-free. Every deterministic context-free language is LR(1).
- Hence, *LR*(1) parsing is sufficient for DCFL . True
- Ans:(D)*LR*(1) parsing is sufficient for deterministic context-free languages.

GATE CS 2022 | Question: 19

- Consider the augmented grammar with $\{+,*,(,),id\}$ as the set of terminals.
- $S' \rightarrow S$
- $S \rightarrow S+R \mid R$
- $R \rightarrow R*P \mid P$
- $P \rightarrow (S) \mid id$
- If I_0 is the set of two $LR(0)$ items $\{[S' \rightarrow S.], [S \rightarrow S.+R]\}$, then $goto(closure(I_0),+)$ contains exactly 5 items.



- **GATE CS 2024 | Set 1 | Question: 16**
- Which of the following is/are Bottom-Up Parser(s)?
- (A)Shift-reduce Parser (B)Predictive Parser
- (C)LL(1) Parser (D)LR Parser
- **Classification of bottom-up parser**
- Shift-Reduce Parsing
- Operator Precedency parser
- LR Parser
  - LR(0) item: LR(0), SLR(1)
  - LR(1) item: CLR(1),LALR(1)
- **Ans :** (A)Shift-reduce Parser, (D)LR Parser
- (B)Predictive Parser & (C)LL(1) Parser are top-down parser

GATE CS 2024 | Set 1 | Question: 28

Consider the following grammar G, with S as the start symbol. The grammar G has three incomplete productions denoted by (1),(2), and (3).

$S \rightarrow daT|(1) \quad T \rightarrow aS|bT|(2) \quad R \rightarrow (3)|\epsilon$

The set of terminals is $\{a,b,c,d,f\}$. The FIRST and FOLLOW sets of the different non-terminals are as follows.

FIRST(S)=$\{c,d,f\}$, FIRST(T)=$\{a,b,\epsilon\}$, FIRST(R)=$\{c,\epsilon\}$

FOLLOW(S)=FOLLOW(T)=$\{c,f,\$\}$, FOLLOW($R$)=$\{f\}$

Which one of the following options CORRECTLY fills in the incomplete productions?

(A)(1) $S \rightarrow Rf$ (2) $T \rightarrow \epsilon$ (3) $R \rightarrow cTR$

(B)(1) $S \rightarrow fR$ (2) $T \rightarrow \epsilon$ (3) $R \rightarrow cTR$

(C)(1) $S \rightarrow fR$ (2) $T \rightarrow cT$ (3) $R \rightarrow cR$

(D)(1) $S \rightarrow Rf$ (2) $T \rightarrow cT$ (3) $R \rightarrow cR$

(A) $S \rightarrow daT|\ Rf \quad T \rightarrow aS|bT|\ \epsilon \quad R \rightarrow cTR|\epsilon$

We can get same FIRST & FOLLOW

(B) $S \rightarrow daT|\ fR \quad T \rightarrow aS|bT|\epsilon \quad R \rightarrow cTR|\epsilon$

FIRST(S)={d,f}$\neq\{c,d,f\}$

(C) $S \rightarrow daT|\ fR \quad T \rightarrow aS|bT|\ cT \quad R \rightarrow cR|\epsilon$

FIRST(S)={d,f} $\neq\{c,d,f\}$, FIRST(T)={a,b,c} $\neq\{a,b,\epsilon\}$

(D) $S \rightarrow daT|\ Rf \quad T \rightarrow aS|bT|\ cT \quad R \rightarrow cR|\epsilon$

FIRST(T)={a,b,c} $\neq\{a,b,\epsilon\}$

Ans : (A)(1) $S \rightarrow Rf$ (2) $T \rightarrow \epsilon$ (3) $R \rightarrow cTR$

GATE CS 2024 | Set 2 | Question: 30

Consider the following context-free grammar where the start symbol is S and the set of terminals is $\{a,b,c,d\}$.

$S \rightarrow AaAb|BbBa \qquad A \rightarrow cS|\epsilon \qquad B \rightarrow dS|\epsilon$

The following is a partially-filled LL(1) parsing table.

	a	b	c	d	$
S	$S \rightarrow AaAb$	$S \rightarrow BbBa$	(1)	(2)	
A	$A \rightarrow \epsilon$	(3)	$A \rightarrow cS$		
B	(4)	$B \rightarrow \epsilon$		$B \rightarrow dS$	

Which one of the following options represents the CORRECT combination for the numbered cells in the parsing table?

Note: In the options, "blank" denotes that the corresponding cell is empty.

(A)(1) $S \rightarrow AaAb$ (2) $S \rightarrow BbBa$ (3) $A \rightarrow \epsilon$ (4) $B \rightarrow \epsilon$

(B)(1) $S \rightarrow BbBa$ (2) $S \rightarrow AaAb$ (3) $A \rightarrow \epsilon$ (4) $B \rightarrow \epsilon$

(C)(1) $S \rightarrow AaAb$ (2) $S \rightarrow BbBa$ (3) blank (4) blank

(D)(1) $S \rightarrow BbBa$ (2) $S \rightarrow AaAb$ (3) blank (4) blank

FIRST(S)={a,b,c,d}, FIRST(A)={c,ϵ}, FIRST(B)={d,ϵ}

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

	a	b	c	d	$
S	$S \rightarrow AaAb$	$S \rightarrow BbBa$	$S \rightarrow AaAb$	$S \rightarrow BbBa$	
A	$A \rightarrow \epsilon$	$A \rightarrow \epsilon$	$A \rightarrow cS$		
B	$B \rightarrow \epsilon$	$B \rightarrow \epsilon$		$B \rightarrow dS$	

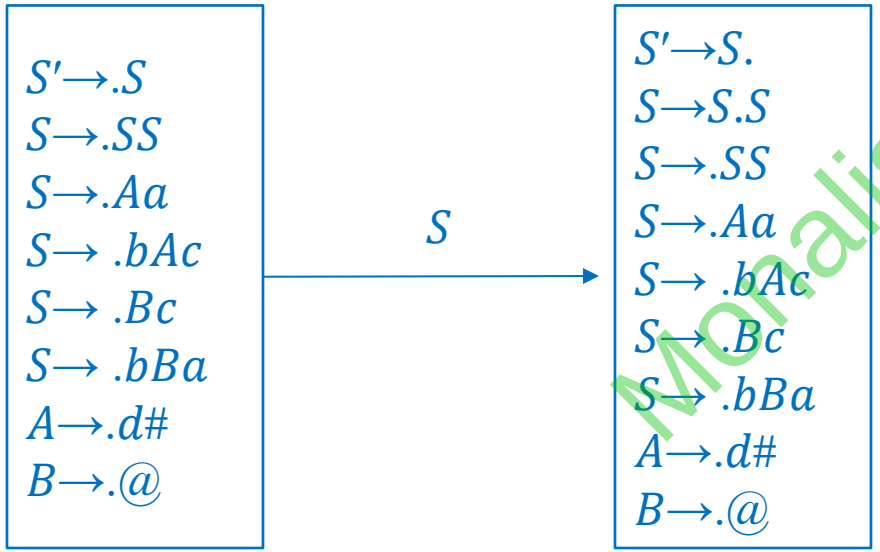
Ans: (A)

GATE CS 2024 | Set 2 | Question: 55

- Consider the following augmented grammar, which is to be parsed with a SLR parser. The set of terminals is $\{a,b,c,d,\#,@\}$

  $S' \rightarrow S \quad S \rightarrow SS|Aa|bAc|Bc|bBa \quad A \rightarrow d\# \quad B \rightarrow @$

  Let I_0=CLOSURE($\{S' \rightarrow \cdot S\}$). The number of items in the set GOTO(I_0,S) is _____.



- The number of items in the set GOTO(I_0,S) is 9.
- Ans: 9