CS3300 : Compiler Design
Assignment - 1

By CS15B049   Lokesh Koshale

Ans. 1 - a -(i) - Regular expression for C comments -
single line comment : “//”.*

(ii) string literals : \"(\\. | [^\\"])*\"
\” for (“) and \\. to match string like “\a” then [^\\"] matches all char except \ and “

Ans 1- b - .*(“abb”).*   matches all string with abb as substring.

Ans 1- d- regular expression for pascal keywords which are case in sensitive

Begin : [bB][eE][gG][iI][nN] ;

End   : [eE][nN][dD];

Function : [fF][uU][nN][cC][tT][iI][oO][nN];

Absolute : [Aa][bB][sS][Oo][Ll][uU][tT][eE] ;

Ans : 2 :   Scheme to convert a Grammar ϵ -free is :

For all Non- terminal Z which has production like Z **-> ϵ ,** we look for all productions whose right side contains Z and replace each occurrence of Z in each of these productions to obtain the non ϵ-productions.
The resultant non- ϵ production are then added to the grammar to retain the language.
If the goal goes to ϵ production we add ϵ production to the goal.

Ex:
S’ → S ;
S → aSbS | bSaS | ϵ .

Here S -> ϵ so we replace all occurrence of right side of S as ϵ which gives
S -> abS | aSb | ab   for   S-> aSbS
S -> baS | bSa | bs   for   S -> bSaS
S’ -> ϵ   for   S’ -> ϵ

our grammar becomes :

S’ -> S | ϵ ;
S -> aSbS | bSaS | abS | aSb | baS | bSa | ab| bs ;

Ans : 3 : **c)** CFG for palindromes with 0,1 and 2

S → 0 S 0
| 1 S 1
| 2 S 2
| 1
| 2
| 0
| ϵ

Ans : 3 : **d)** CFG for all strings of 0's and 1's that doesn't contain 011 as substring

A → 1 A
| B

B → 0 B
| C

C → 01 C
| 0 C
| ϵ

Ans : 3 : **b)** CFG for java comments

allChar belongs to the set of all valid characters except '\n' new-line

S → // A
| /* B

A → allChar A
| ϵ

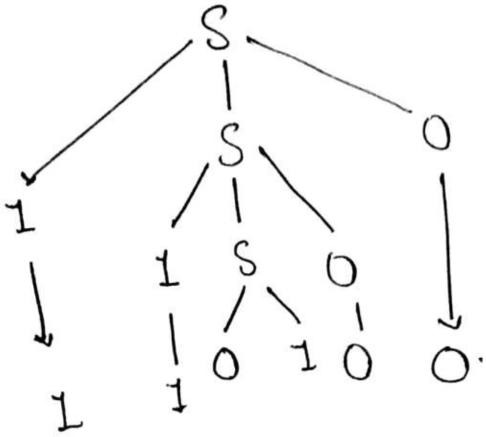
B → A N B
| */

N → \n

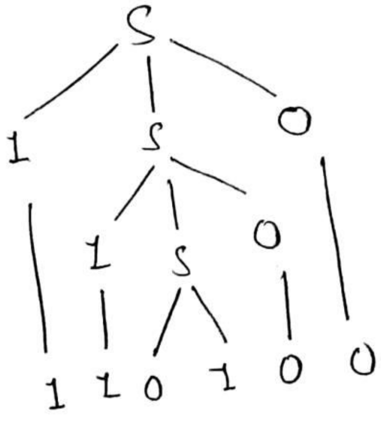
Ans-9: (a) $S \rightarrow 1S0 \mid 01$

given String: 110100.

Left-derivation Tree



Right-Derivation Tree:

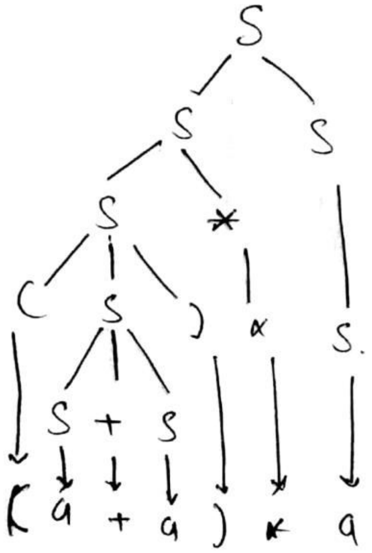


The grammar is not ambigious as left and right subtree ~~g~~ are same.

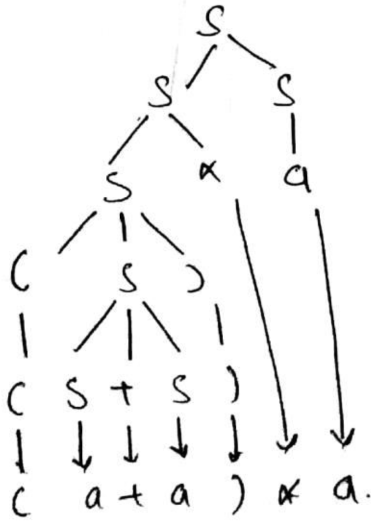
4(b) - $S \to S + S$
$\mid SS$
$\mid (S)$
$\mid S*$
$\mid a.$

given String : (a+a)*a

Left-derivation tree:



Right derivation: tree.

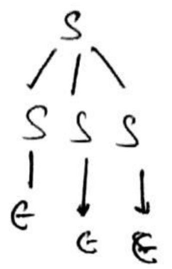


The above grammar is ambiguous left derivation tree is not same as right derivation tree .

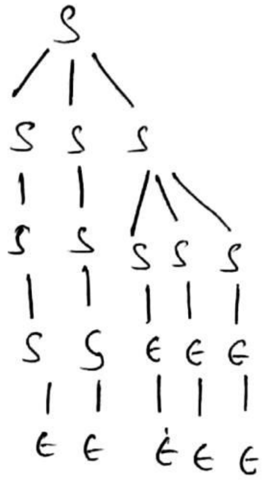
4- (c) $S \rightarrow S\ S\ S$
$\quad\quad | \in$

String = $\in$ (empty string)

Left Derivation:



Right Derivation!



The given grammar is ambigious as right-derivation tree is different from left derivation tree.

Ans6:

$S \longrightarrow SS+ \mid SS* \mid T$

$T \longrightarrow id \mid num$

(a) building LR(0) parsing Table

we add $S' \longrightarrow S\$$

States.

I_0: $S' \longrightarrow \cdot S\$$
$S \longrightarrow \cdot SS+$
$S \longrightarrow \cdot SS*$
$S \longrightarrow \cdot T$
$T \longrightarrow \cdot id$
$T \longrightarrow \cdot num$

I_1: $S' \longrightarrow S \cdot \$$
$S \longrightarrow S \cdot S+$
$S \longrightarrow S \cdot S*$
$S \longrightarrow \cdot SS+$
$S \longrightarrow \cdot SS*$
$S \longrightarrow \cdot T$
$T \longrightarrow \cdot id$
$T \longrightarrow \cdot num$

I_2: $S \longrightarrow T \cdot$

I_3: $T \longrightarrow id \cdot$

I_4: $T \longrightarrow nom \cdot$

I_5: $S' \longrightarrow S\$ \cdot$

I_6: $S \longrightarrow SS \cdot +$
$S \longrightarrow SS \cdot *$
$S \longrightarrow S \cdot S+$
$S \longrightarrow S \cdot S*$
$S \longrightarrow \cdot SS+$
$S \longrightarrow \cdot SS*$
$S \longrightarrow \cdot T$
$T \longrightarrow \cdot id$
$T \longrightarrow \cdot nom$

I_7: $S \longrightarrow SS+ \cdot$

I_8: $S \longrightarrow SS* \cdot$

Parse Table :

state	Actions					GOTO	
	+	*	Id	num	$	S	T
0	-	-	s3	s4	-	1	2
1	-	-	s3	s4	s5	6	2
2	r3	r3	r3	r3	r3	-	-
3	r4	r4	r4	r4	r4	-	-
4	r5	r5	r5	r5	r5	-	-
5	acc	acc	acc	acc	acc	-	-
6	s7	s8	s3	s4	-	6	2
7	r2	r2	r2	r2	r2	-	-
8	r3	r3	r3	r3	r3	-	-

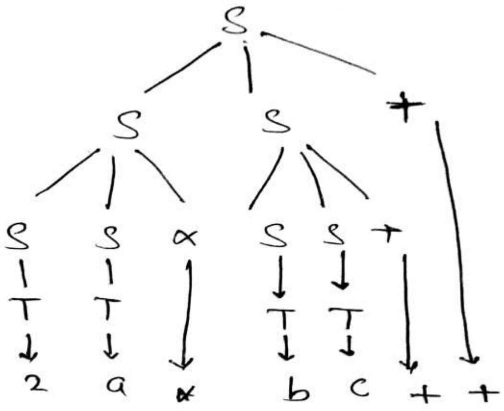
The given grammar is LR(0) , as there is no shift reduce conflicts in parse table.

The Sequence of action for String : 2 a * b c + +

The sequence of actions.

Stack	input.	Actions.
0	2 a * b c + + $	s4
0 4	a * b c + + $	r5
0 2	a * b c + + $	r3
0 1	a * b c + + $	s3.
0 1 3	* b c + + $	r4
0 1 2	* b c + + $	r3
0 1 6	* b c + + $	s8.
0 1 6 8	b c + + $	r2.
0 1	b c + + $	s6
0 1 6	c + + $	s6
0 1 6 6	+ + $	s7
0 1 6 6 7	+ $	r1.
0 1 6	+ $	s7
0 1 6 7	$	r1.
0 1	$	s5
0 1 5	$	acc.

Ans6' Parse Tree:



ANSWER 7 -- LR(1) Parse Table for the given grammar :

Ans7: 1. Crownoise → kaa Crownoise
2. | kaa.

we add
0. S' → crownoise $

I₀: S' → .Crownoise, $
Crownoise → .kaa Crownoise, $
Crownoise → .kaa , $

I₁: S' → Crownoise. , $

I₂: Crownoise → kaa. Crownoise ,$
Crownoise → kaa. $
Crownoise → .kaa crownoise, $
Crownoise → .kaa , $

I₃: Crownoid → kaa Crownoise. , $

Parse Table LR(1)

●	Actions		GOTO
	kaa	$	Crownoise.
0	s2	—	1
1	—	acc	—
2	s2	r2	3
3	—	r1	—

for string "kaa kaa kaa"

Stack	input	Actions.
0	kaa kaa kaa $	s 2
0 2	kaa kaa $	s 2
0 2 2	kaa $	s 2
0 2 2 2	$	r 2
0 2 2 3	$	r 1
0 2 3	$	r 1.
0 1	$	accept.

Ans-5-(a) SLR(1) items and goto functions for

(1) 4(b) $S \to S+S$ 1

$| SS$ 2

$| (S)$ 3

$| S\alpha$ 4

$| a$ 5.

We have $S' \to S\$$ 0.

I_0: $S' \to .S\$$

$S \to .S+S$

$S \to .SS$

$S \to .S*$

$S \to .(S)$

$S \to .a$

I_1: $S' \to S.\$$

$S \to S.+S$

$S \to S.S$

$S \to S.*$

$S \to .S+S$

$S \to .SS$

$S \to .S\alpha$

$S \to .(S)$

$S \to .a$.

I_2: $S \to (.S)$

$S \to .S+S$

$S \to .SS$

$S \to .S\alpha$

$S \to .(S)$

$S \to .a$.

I_3: $S \to a.$

I_5: $S \to SS.$

$S \to S.+S$

$S \to S.S$

$S \to S.\alpha$.

$S \to .S+S$

$S \to .SS$

$S \to .S*$

$S \to .(S)$

$S \to .a$

I_6: $S \to S*.$

I_7: $S \to (S.)$

$S \to S.+S$

$S \to S.S$

$S \to S.*$

$S \to .S+S$

$S \to .SS$

$S \to .S*$

Continue

$\dashrightarrow I_7$: $S \to .(S)$

$S \to .a$

I_4: $S \rightarrow S+.S$

$S \rightarrow .SS$

$S \rightarrow .S+S$

$S \rightarrow .S*$

$S \rightarrow .(S)$

$S \rightarrow .a$

I_8: $S \rightarrow S+S.$

$S \rightarrow S.S$

$S \rightarrow S.+S$

$S \rightarrow S.*$

$S \rightarrow .S+S$

$S \rightarrow .SS$

$S \rightarrow .S*$

$S \rightarrow .(S)$

$S \rightarrow .a$

I_9: $S \rightarrow (S).$

Parse Table: Follow of S = +, *, a. (

	Actions.						GoTo.
	a	+	*	(	)	$	S.
0	s3	-	-	s2	-	-	1
1	s3	s4	s6	s2	-	acc	5
2	s3	-	-	s2	-	-	7
3	r5	r5	r5	r5	-	-	-
4	s3	-	-	s2	-	-	8.
5	s4/r2	r2	r2/s6	r2/s2	-	-	5.
6	r4	r4	r4	r4	-	-	-
7	s3	s4	s6	s2	s9	-	5
8.	s3/r1	r1/s4	r1/s6	s2/r1	-	-	5
9.	r3	r3	r3	r3	-	-	-

The grammar is ambigious as it has sift reduce Conflicts.

Ans 5 -(b) $S \rightarrow S\ S\ S$. 1

$\mid \epsilon$ 2

$S' \rightarrow S\ \$$ 0.

I_0: $S' \rightarrow .S\ \$$

$S \rightarrow .SSS$

$S \rightarrow .\epsilon$

I_1: $S \rightarrow S.\$$

$S \rightarrow S.SS$

$S \rightarrow .SSS$

$S \rightarrow .\epsilon.$

I_2: $S \rightarrow \epsilon.$

I_3: $S \rightarrow SS.S$

$S \rightarrow .SSS$

$S \rightarrow .e.$

$S \rightarrow S.SS.$

I_4: $S \rightarrow SSS.$

$S \rightarrow S.SS$

$S \rightarrow .e.$

$S \rightarrow SS.S$

The grammar is ambiguous in state 4 we can reduce using rule 1 for e as well as shift to stat 2 so there is a shift / reduce conflict.