Index No. _____ Continuation Sheet No. _____

Q #2

(a)

```
bool S()
{
    if (A())
        return true;
    else return false;
}

bool A()
{
    if ( token == '(' && A() && token == ')')
        return true;
    else if ( A() && token == '*' && X())
        return true;
    else if (X())
        return true;
    else return false;
}
```

```
bool X()
{
    if (X() && token == '-' && A())
        return true;
    else if (token == 'a')
        return true;
    else if (token == 'b' && X())
        return true;
    else
        return false;
}
```

input: a-(ba* a)

parsing order:

S called
A "
X "
~~token match with a, return true.~~
X " (X-A)
token match with a, return true.
token " " -, ~~return true~~
A called
token match with (, ~~return true~~

A called (A * X)

X called (X - A)

token match with b, ~~Return true~~.

X called

token match with a, return true

token match " *, "

X called

token match with a, return true.

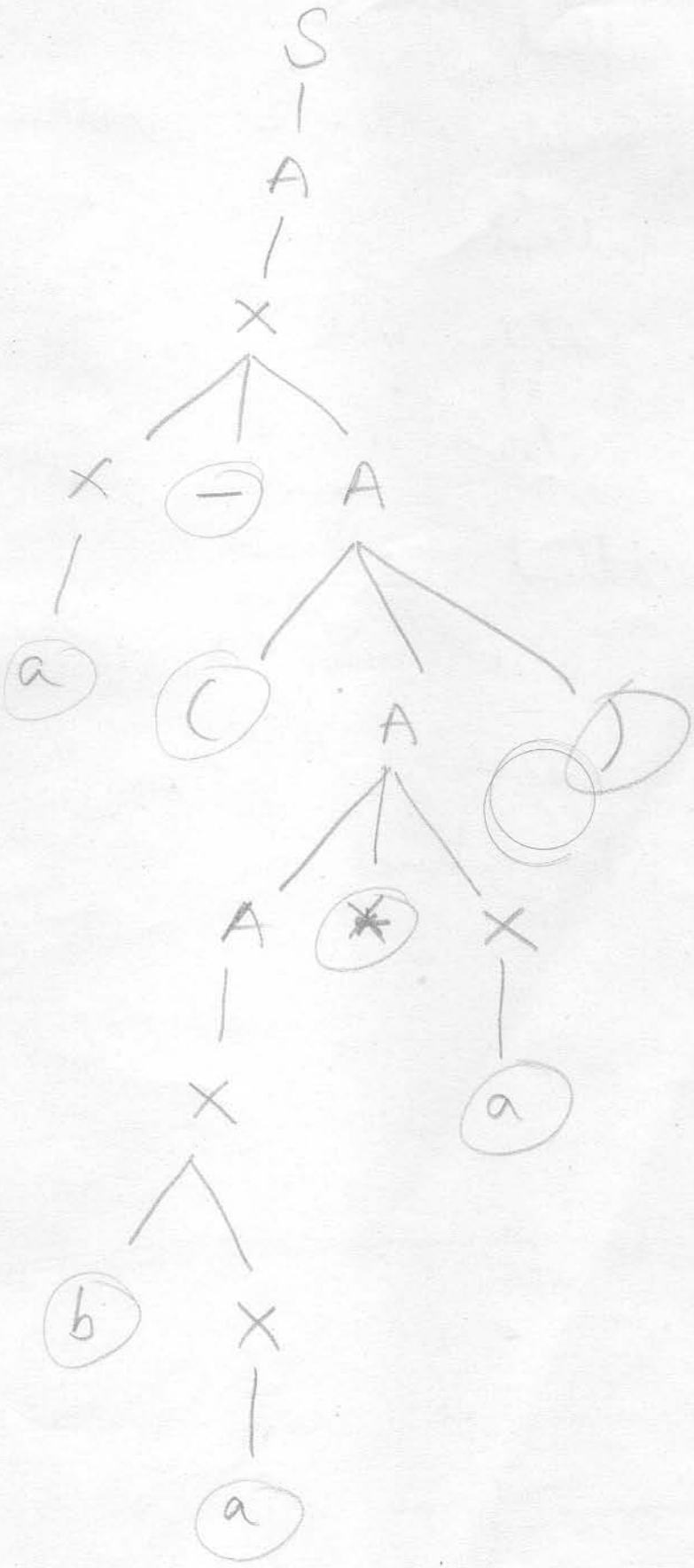
token match with), " "

A return true

S " "

(b)

a − (ba * a).



C

<ARITH_EXPR> ⟶ <EXPR> <ARITH_OPR> <EXPR>

| <LEFT UNARY_OPR> <IDENT> |

<IDENT> <RIGHT_UNARY_OPR> |

<TERM>

<ARITH_OPR> ⟶ + | - | * | / | %

<LEFT_UNARY_OPR> ⟶ - | -- | + | ++ | ~

<RIGHT_UNARY_OPR> ⟶ ++ | --

<EXPR> ⟶ <ARITH_EXPR> | <LOGICAL_EXPR>

| <RELAT_EXPR> | (<EXPR>)

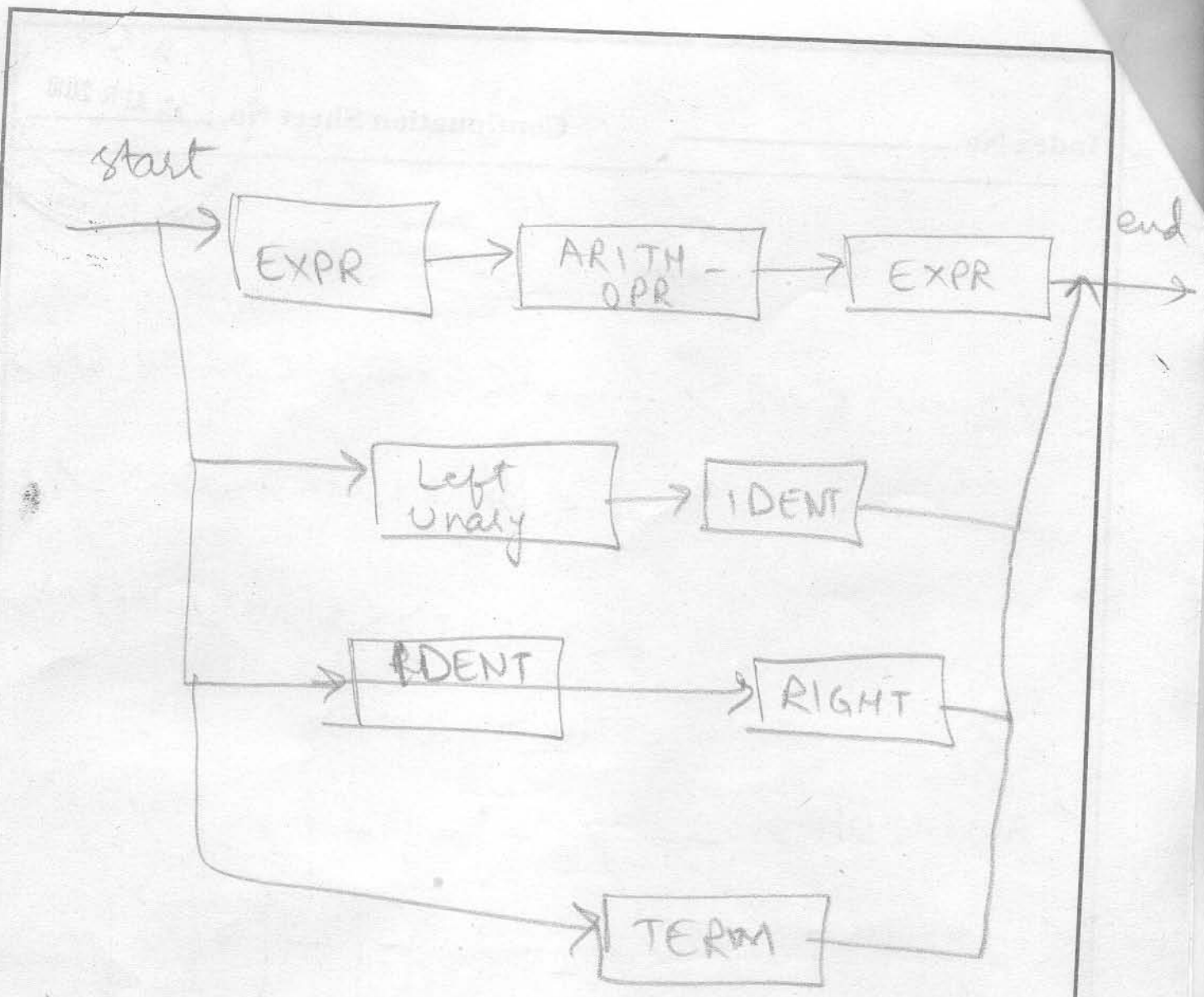
<TERM> ⟶ <LITERAL> | <FUN_CALL>

<LITERAL> ⟶ <INT_CONST> | <CHAR_CONST> |

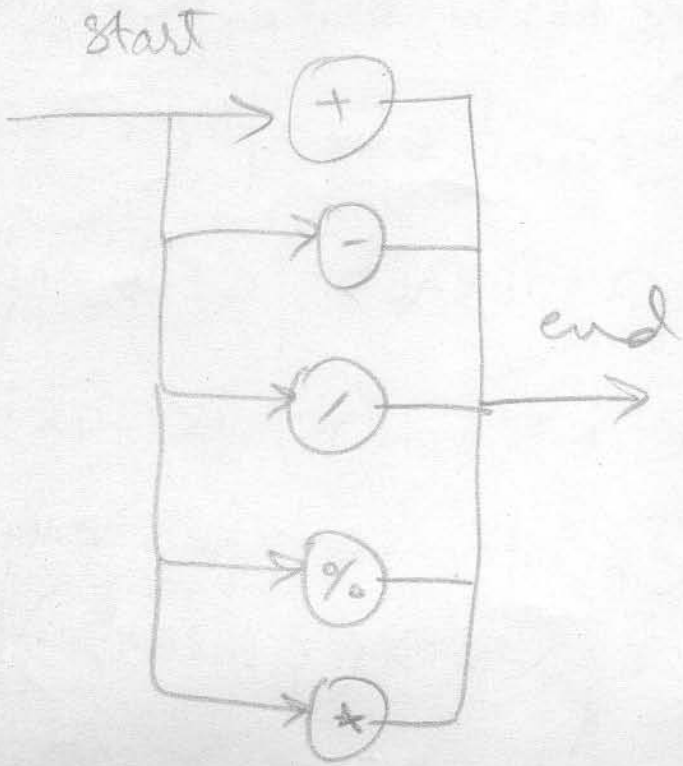
<FLOAT_CONST> | <STRING_CONST>

<FUN_CALL> ⟶ <IDENT> (<PARAM>)

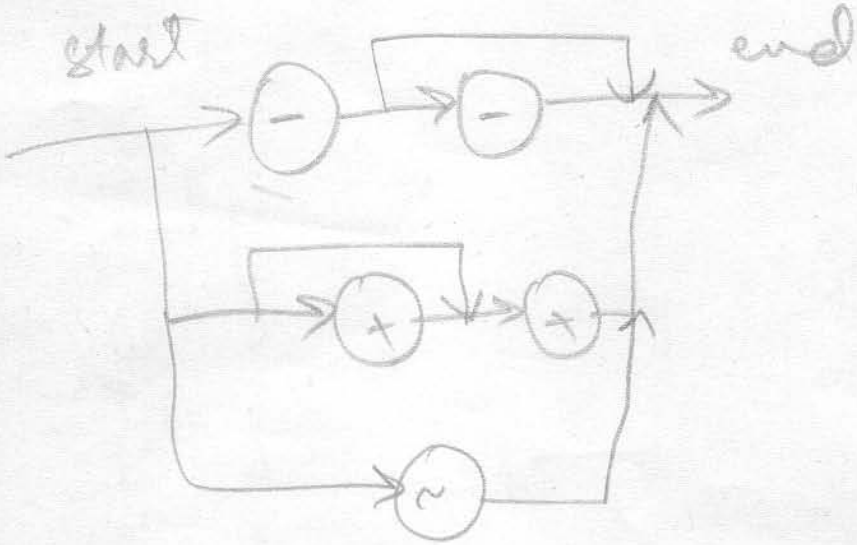
<PARAM> ⟶ ε | <LITERAL> <MORE> |


start
EXPR
ARITH - OPR
EXPR
end
Left Unary
IDENT
IDENT
RIGHT
TERM

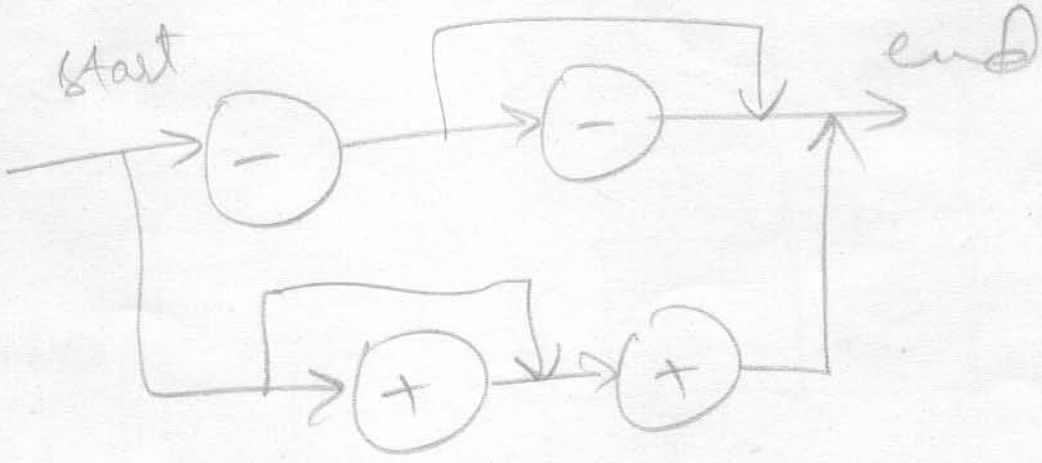
ARITH_OPR


start
+
-
/
%
*
end

Left_Unary_Opr



Right_Unary_Opr



and so on

Q: 3

(a)

1. $S \rightarrow xyA$
2. $A \rightarrow Bx$
3. $A \rightarrow y$
4. $B \rightarrow C\delta$
5. $C \rightarrow S$
6. $C \rightarrow \delta$

First

$\{x\}$

$\{x, \delta\}$

$\{y\}$

$\{x, \delta\}$

$\{x\}$

$\{\delta\}$

	x	y	δ	$
S	1			
A	2	3	2	
B	4		4	
C	5		6	

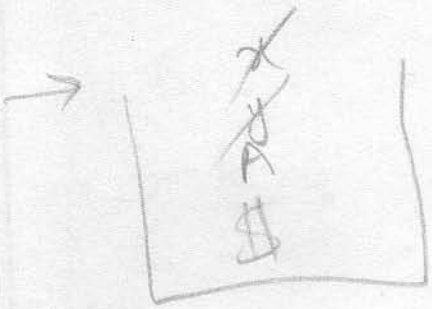
Yes it is LL(1).

Index No. _____ Continuation Sheet No. 9

input =

$xyxyyzx\$$

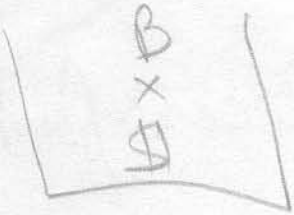
prod = 1.



~~$xyxyyzx\$$~~

$xyyzx\$$

prod - 2.



$xyyzx\$$

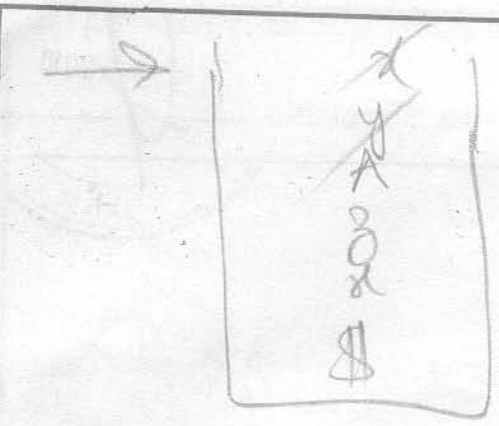
prod - 4

$xyyzx$

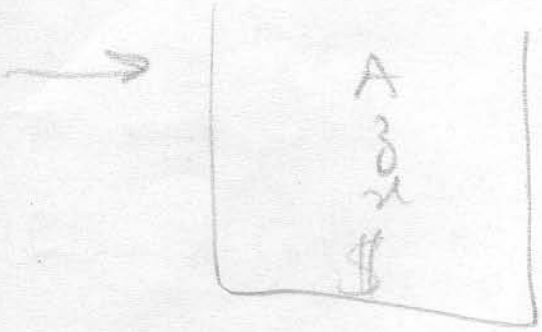
prod - 5

$xyyzx$

prod - 1.

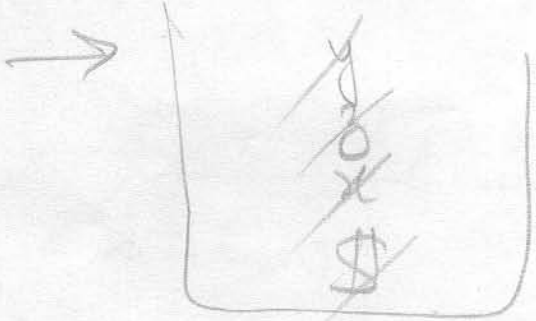


$xyyz x \$$



$yz x \$$

Prod - 3



$yz x \$$

Parsing Successful!

(6)

Index No. _____ Continuation Sheet No. _____

b

$S \rightarrow a = AB;$ LL(0)

$A \rightarrow B \mid C$

$B \rightarrow aC \mid bA$ — LL(1)

$C \rightarrow \varepsilon \mid BC$ — LL(1)

$A \rightarrow aC \mid bA \mid \varepsilon \mid BC$

↓

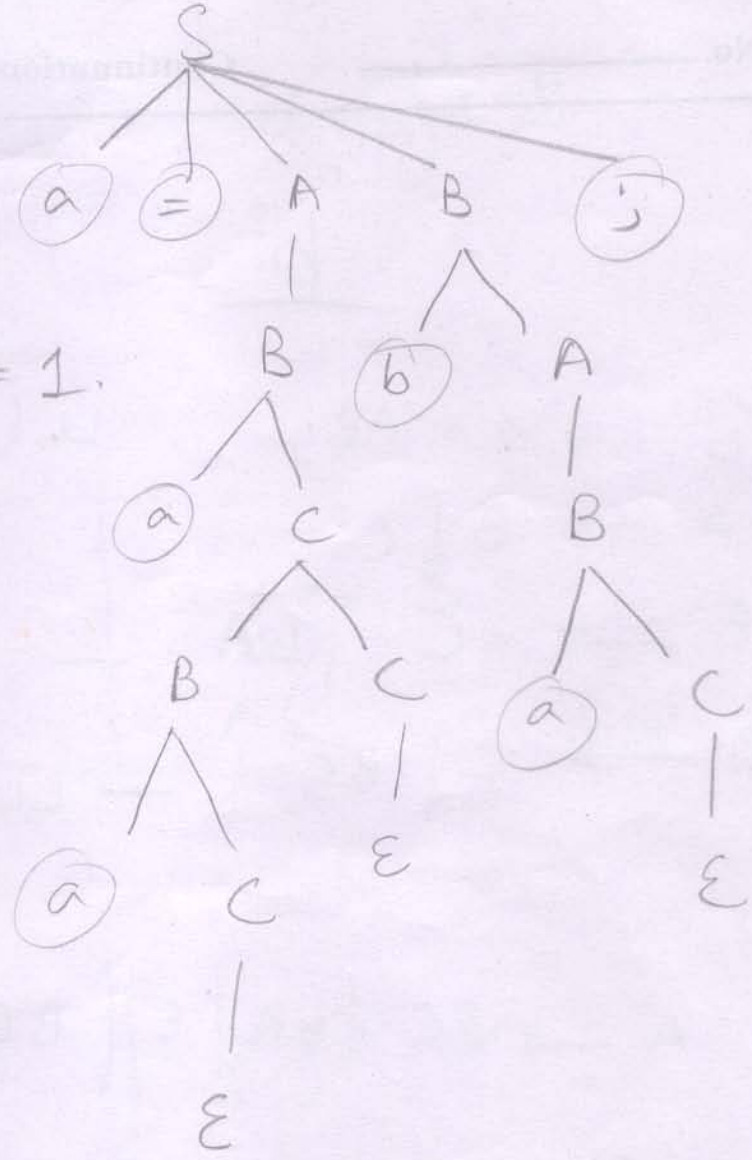
$A \rightarrow aC \mid bA \mid \varepsilon \mid aCC \mid bAC$

not LL(1).

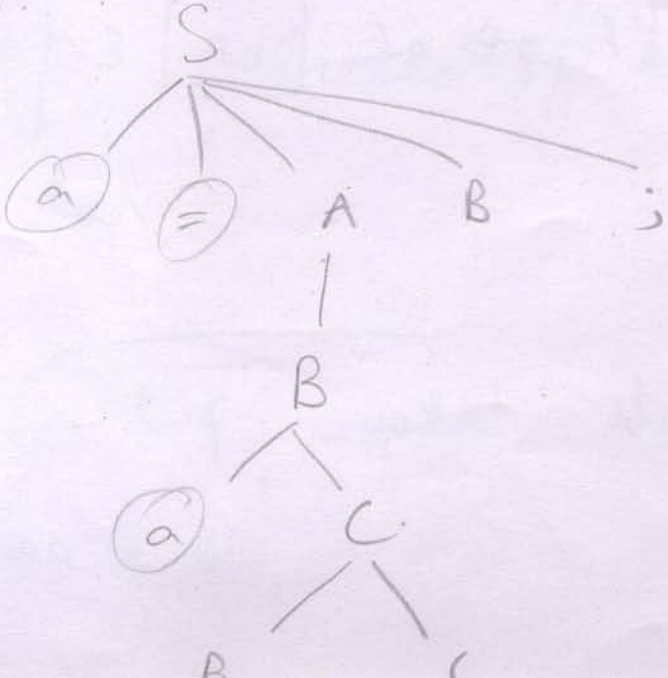
Lets taken input as

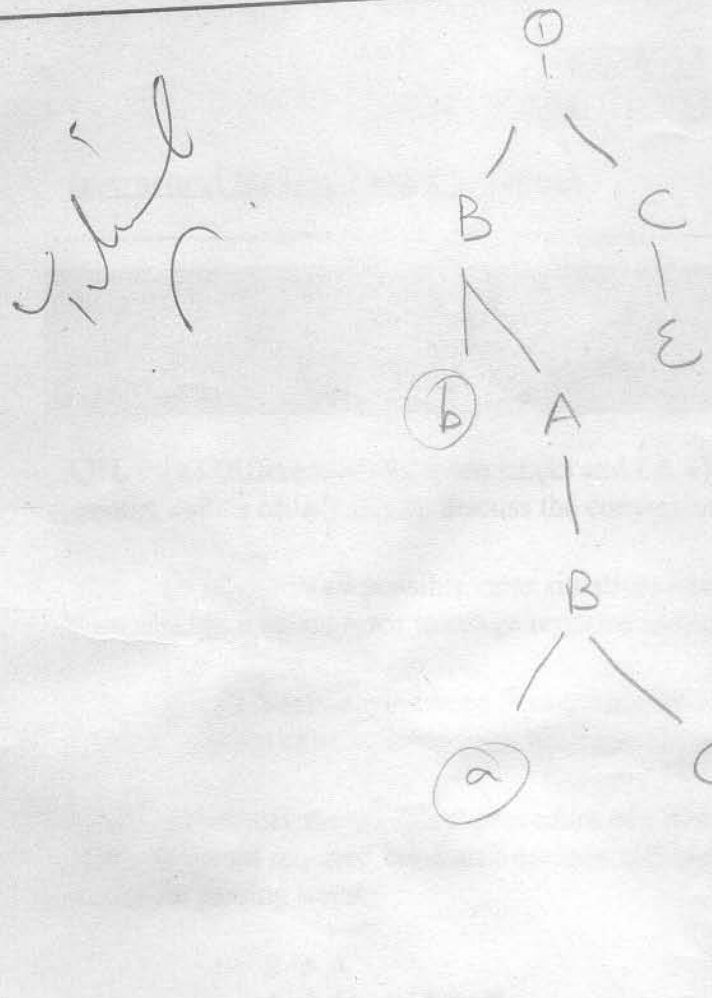
$a = aaba;$

parse tree # 1.



parse tree # 2.





Yes it is ambigous!

Not left-recursive

since no production has the same symbol on L.H.S and R.H.S.

$S \rightarrow a = ABs$ ×

$A \rightarrow bA|\varepsilon|BC$ ×