Midterm Review

CSCE 322

Name: ____________________

Instructions

Please solve the problems presented below. **Show your work to receive full credit; just an answer is not enough**. **No Approximations**.

Question 1 (12 points)

(a) Describe in English the language defined by the regular expression `^([1-9][0-9]*)?[13579]$`.

> **Solution:**
> Positive, odd integers, without leading zeros

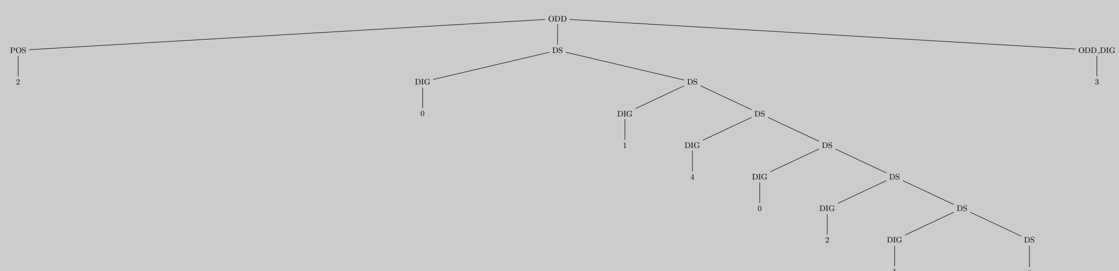
(b) Write an unambiguous context-free grammar that generates the same language.

> **Solution:**

$$
\begin{array}{rl}
\text{ODD} & \rightarrow \text{ODD\_DIG} \\
\text{ODD} & \rightarrow \text{POS DS ODD\_DIG} \\
\text{ODD\_DIG} & \rightarrow 1 \mid 3 \mid 5 \mid 7 \mid 9 \\
\text{POS} & \rightarrow \text{ODD\_DIG} \mid 2 \mid 4 \mid 6 \mid 8 \\
\text{DS} & \rightarrow \text{DIG DS} \\
\text{DS} & \rightarrow \epsilon \\
\text{DIG} & \rightarrow 0 \mid \text{POS}
\end{array}
$$

(c) Using your grammar from part (b), give a derivation of the string `20140213`.

> **Solution:**



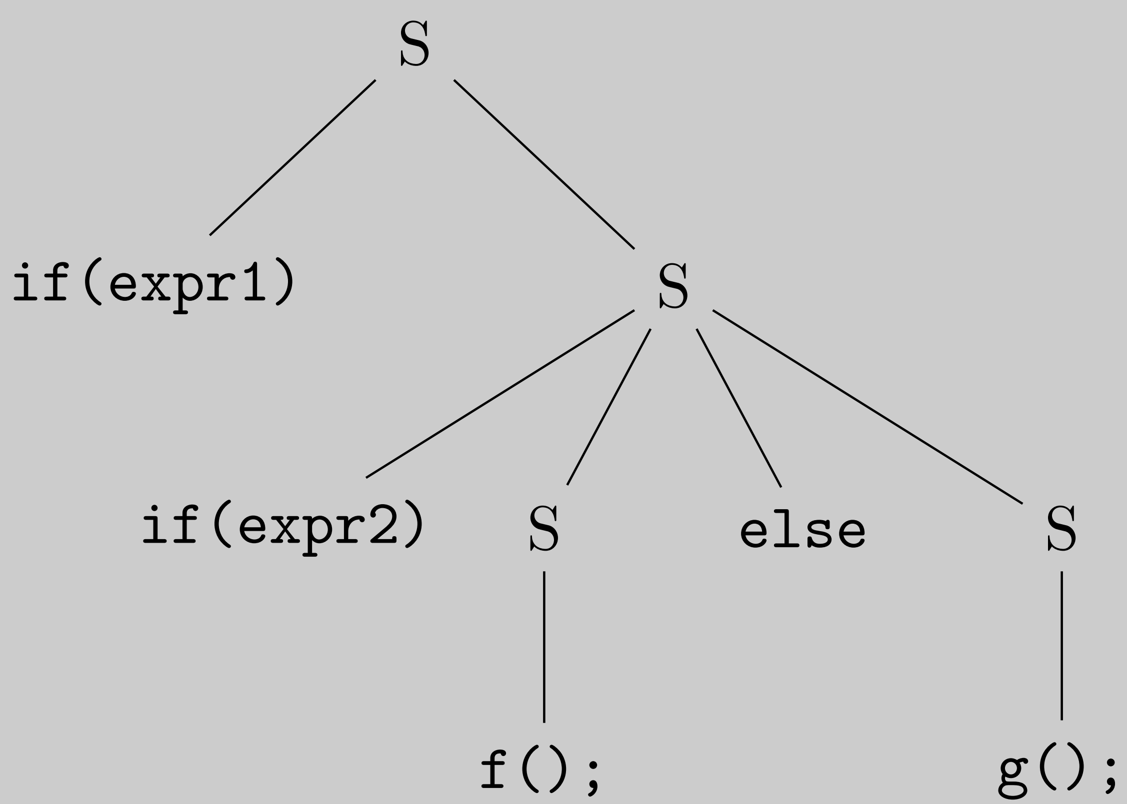
Question 2 (10 points)

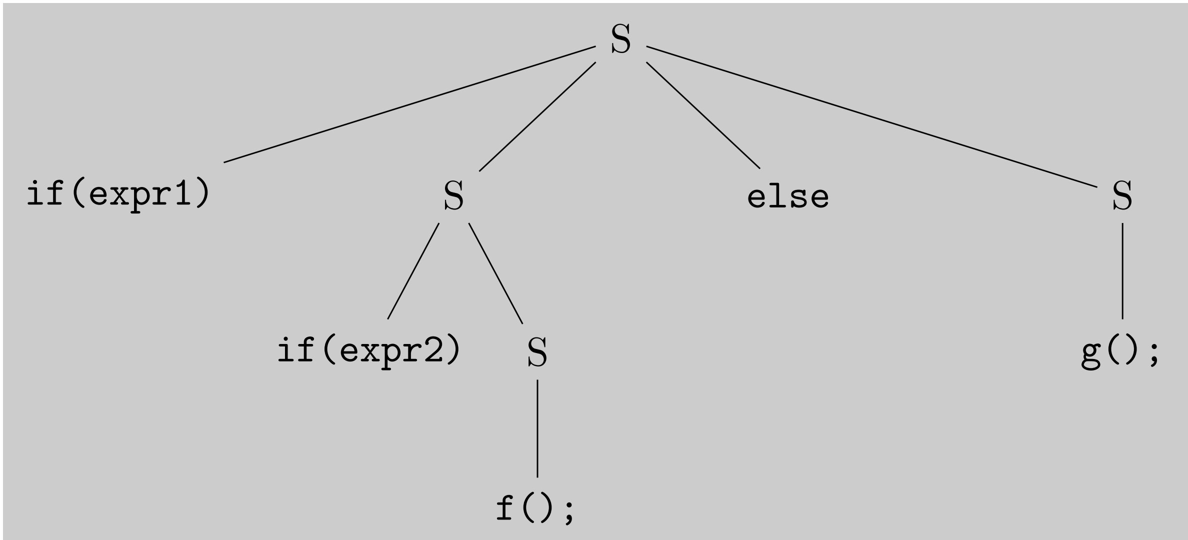
Consider this top-down grammar for `if` statements:

$$\begin{aligned} S &\rightarrow \text{if (expression) } S \\ S &\rightarrow \text{if (expression) } S \text{ else } S \\ S &\rightarrow \text{other} \end{aligned}$$

Give two parse trees for the expression `if ( expr1 ) if( expr2 ) f(); else g();` that prove this grammar is ambiguous.

Solution:




S
if(expr1)
S
else
S
if(expr2)
S
g();
f();

Question 3 (18 points)

Consider the following CFG for octal numbers.

$$
\begin{aligned}
O &\rightarrow N\ O \\
O &\rightarrow \epsilon \\
N &\rightarrow \mathtt{0|1|2|3|4|5|6|7}
\end{aligned}
$$

Augment this grammar with attribute rules that will accumulate the value of the number into a `val` attribute of the root of the parse tree.

Solution:

$O_1 \rightarrow N\ O_2$

$\triangleright$ O_1.length = N.length + O_2.length

$\triangleright$ O_1.val = N.value $\times 8^{\mathrm{O_2.length}}+ O_2$.val

$O \rightarrow \epsilon$

$\triangleright$ O.length = 0

$\triangleright$ O.value = 0

$N \rightarrow \mathtt{0|1|2|3|4|5|6|7}$

$\triangleright$ N.value = parseInt(N) (or whatever will extract the value from a single character)

$\triangleright$ N.length = 1

Question 4 (24 points)

For the regular expression `^(-)?[^0][0-9]*('.'[0-9]+)?$`, determine which of the following inputs will match

(a) `3.14159`

Solution:
Yes

(b) `-2`

Solution:
Yes

(c) `F.75`

Solution:
Yes

(d) `0.7071`

Solution:
No

(e) `.5`

Solution:
Yes

(f)

Solution:
No

(g) `->`

Solution:
Yes

(h) `F80000`

Solution:
Yes

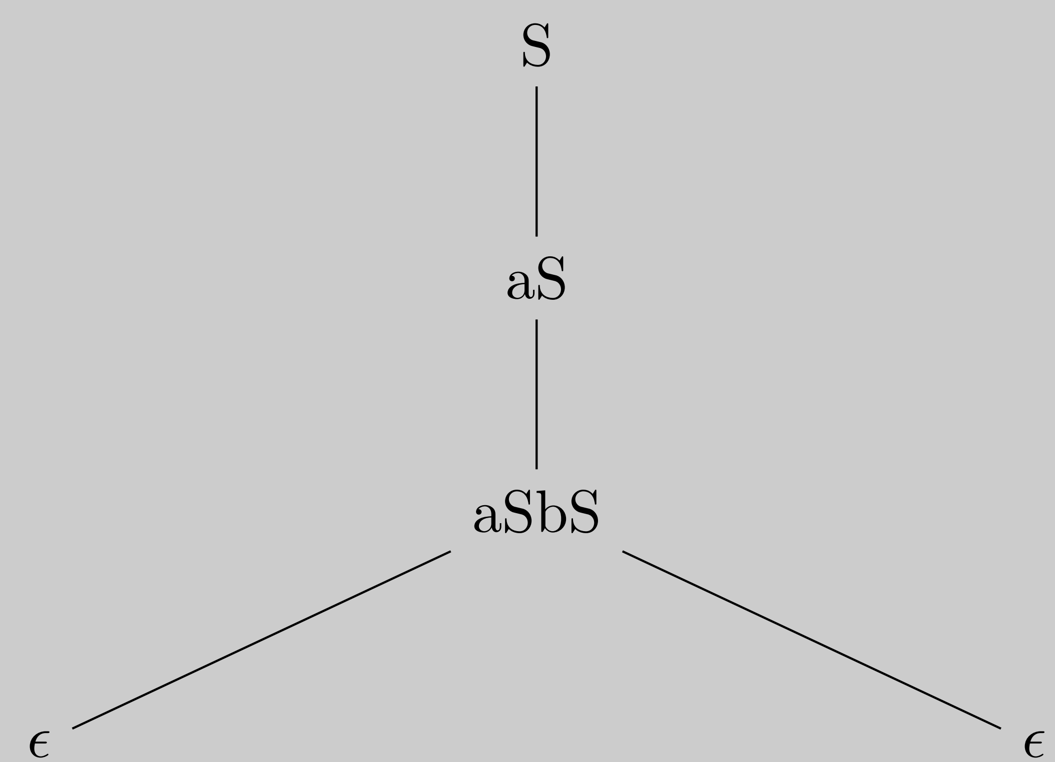
Question 5 (24 points)

Consider this top-down grammar

$$\begin{aligned} S &\rightarrow aS \\ S &\rightarrow aSbS \\ S &\rightarrow \epsilon \end{aligned}$$

(a) Provide the parse tree for the input `aab`

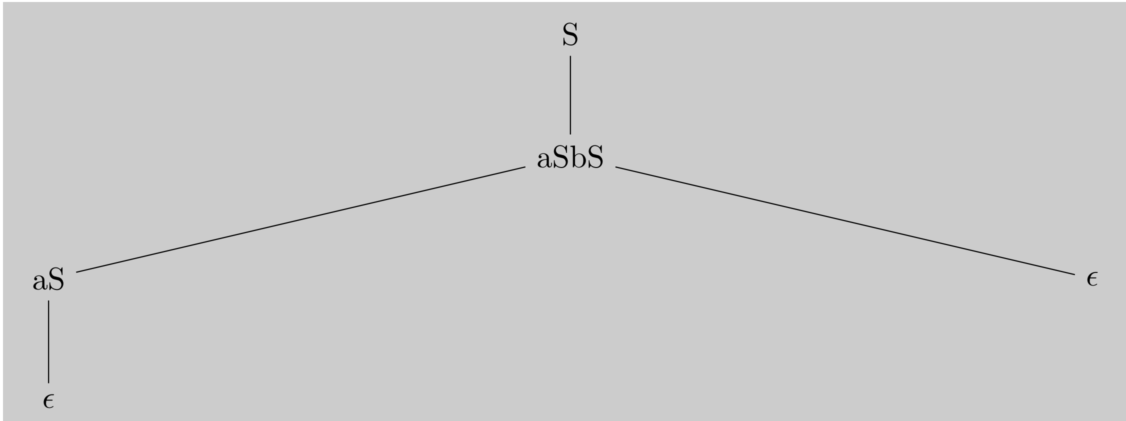
Solution:



(b) Is this language ambiguous? If so, provide an alternate parse tree for `aab` to prove it. If not, why not?

Solution:

Yes.


S
aSbS
aS
ϵ
ϵ

Question 6 (26 points)

Consider the following CFG for binary numbers.

$$
\begin{aligned}
B &\rightarrow Z \\
&\triangleright B\texttt{.twice = false} \\
B &\rightarrow N\ M \\
Z &\rightarrow \texttt{0} \\
N &\rightarrow \texttt{1} \\
M &\rightarrow Z\ M \\
M &\rightarrow N\ M \\
M &\rightarrow \epsilon
\end{aligned}
$$

(a) Augment this grammar with attribute rules that will accumulate `true` into a `twice` attribute of the root of the parse tree if the string contains at least twice as many 1s as 0s, and `false` otherwise.

Solution:

$$
\begin{aligned}
B &\rightarrow Z \\
&\triangleright B\texttt{.twice = false} \\
B &\rightarrow N\ M \\
&\triangleright B\texttt{.twice =}\ \tfrac{M.ones+1}{2} \geq M.zeros \\
Z &\rightarrow \texttt{0} \\
N &\rightarrow \texttt{1} \\
M_1 &\rightarrow Z\ M_2 \\
&\triangleright M_1\texttt{.zeros =}\ M_2\texttt{.zeros + 1} \\
M_1 &\rightarrow N\ M_2 \\
&\triangleright M_1\texttt{.ones =}\ M_2\texttt{.ones + 1} \\
M &\rightarrow \epsilon \\
&\triangleright M\texttt{.zeros = 0} \\
&\triangleright M\texttt{.ones = 0}
\end{aligned}
$$

(b) Is your attribute grammar S-attributed?

Solution:

Yes.

Question 7 (15 points)

(a) What is the input to a scanner?

Solution:
Character stream

(b) What is the input to a parser?

Solution:
Token stream

(c) What is the input to a semantic analyzer?

Solution:
Parse tree

Question 8 (18 points)

Consider the following CFG for hexidecimal numbers

$$
\begin{aligned}
H &\rightarrow N\ NS \\
NS &\rightarrow N\ NS \\
NS &\rightarrow \epsilon \\
N &\rightarrow \texttt{0} \\
&\triangleright N\texttt{.value = 0} \\
N &\rightarrow \texttt{1} \\
&\triangleright N\texttt{.value = 1} \\
N &\rightarrow \texttt{2} \\
&\triangleright N\texttt{.value = 2} \\
N &\rightarrow \texttt{3} \\
&\triangleright N\texttt{.value = 3} \\
N &\rightarrow \texttt{4} \\
&\triangleright N\texttt{.value = 4} \\
N &\rightarrow \texttt{5} \\
&\triangleright N\texttt{.value = 5} \\
N &\rightarrow \texttt{6} \\
&\triangleright N\texttt{.value = 6} \\
N &\rightarrow \texttt{7} \\
&\triangleright N\texttt{.value = 7} \\
N &\rightarrow \texttt{8} \\
&\triangleright N\texttt{.value = 8} \\
N &\rightarrow \texttt{9} \\
&\triangleright N\texttt{.value = 9} \\
N &\rightarrow \texttt{a} \\
&\triangleright N\texttt{.value = 10} \\
N &\rightarrow \texttt{b} \\
&\triangleright N\texttt{.value = 11} \\
N &\rightarrow \texttt{c} \\
&\triangleright N\texttt{.value = 12} \\
N &\rightarrow \texttt{d} \\
&\triangleright N\texttt{.value = 13} \\
N &\rightarrow \texttt{e} \\
&\triangleright N\texttt{.value = 14} \\
N &\rightarrow \texttt{f} \\
&\triangleright N\texttt{.value = 15}
\end{aligned}
$$

Augment this grammar with attribute rules that will accumulate the Decimal representation `dec` into the root of the parse tree.

Solution:

$$
\begin{array}{rl}
H & \rightarrow N\ NS \\
 & \triangleright\ H\texttt{.value = } NS\texttt{.value + } N\texttt{.value } \times 16^{NS\texttt{.length}} \\
NS_1 & \rightarrow N\ NS_2 \\
 & \triangleright\ NS_1\texttt{.value = } NS_2\texttt{.value + } N\texttt{.value } \times 16^{NS_2\texttt{.length}} \\
 & \triangleright\ NS_1\texttt{.length = } NS_2\texttt{.length + 1} \\
NS & \rightarrow \epsilon \\
 & \triangleright\ NS\texttt{.length = 0} \\
 & \triangleright\ NS\texttt{.value = 0} \\
N & \rightarrow \texttt{0} \\
 & \triangleright\ N\texttt{.value = 0} \\
N & \rightarrow \texttt{1} \\
 & \triangleright\ N\texttt{.value = 1} \\
N & \rightarrow \texttt{2} \\
 & \triangleright\ N\texttt{.value = 2} \\
N & \rightarrow \texttt{3} \\
 & \triangleright\ N\texttt{.value = 3} \\
N & \rightarrow \texttt{4} \\
 & \triangleright\ N\texttt{.value = 4} \\
N & \rightarrow \texttt{5} \\
 & \triangleright\ N\texttt{.value = 5} \\
N & \rightarrow \texttt{6} \\
 & \triangleright\ N\texttt{.value = 6} \\
N & \rightarrow \texttt{7} \\
 & \triangleright\ N\texttt{.value = 7} \\
N & \rightarrow \texttt{8} \\
 & \triangleright\ N\texttt{.value = 8} \\
N & \rightarrow \texttt{9} \\
 & \triangleright\ N\texttt{.value = 9} \\
N & \rightarrow \texttt{a} \\
 & \triangleright\ N\texttt{.value = 10} \\
N & \rightarrow \texttt{b} \\
 & \triangleright\ N\texttt{.value = 11} \\
N & \rightarrow \texttt{c} \\
 & \triangleright\ N\texttt{.value = 12} \\
N & \rightarrow \texttt{d} \\
 & \triangleright\ N\texttt{.value = 13} \\
N & \rightarrow \texttt{e} \\
 & \triangleright\ N\texttt{.value = 14} \\
N & \rightarrow \texttt{f} \\
 & \triangleright\ N\texttt{.value = 15}
\end{array}
$$

Question 9 (12 points)

For the regular expression `^/'*'[^*/]*'*'/$`, determine which of the following inputs will match

(a) `/* This is a Java comment */`

Solution:
Yes

(b) `// This is also a Java comment`

Solution:
No

(c) `-- This is a Haskell comment`

Solution:
No

(d) `% This is a Prolog comment`

Solution:
No

(e) `/* Is *this* a Java comment? */`

Solution:
No

(f)

Solution:
No

(g) `/* What is the value of array[0]? */`

Solution:
Yes

(h) `/* Is this a /* comment */ within a comment? */`

Solution:
No

(i) `/* What is the value of x^2? */`

Solution:
Yes

(j) `/* How much is $Texas ? */`

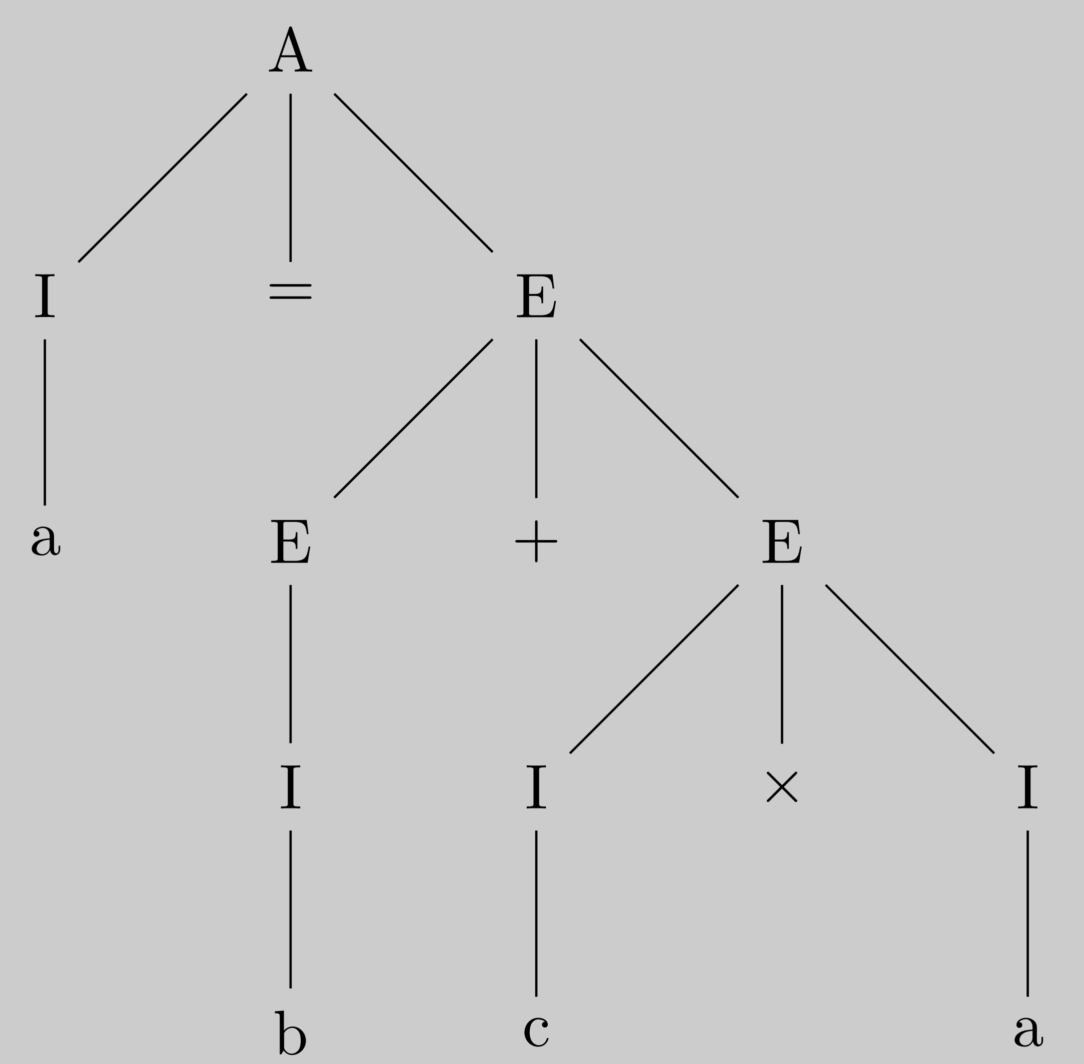
Solution:
Yes

Question 10 (12 points)
Consider this grammar

$$
\begin{aligned}
A &\rightarrow I \texttt{ = } E \\
I &\rightarrow \texttt{a} \mid \texttt{b} \mid \texttt{c} \\
E &\rightarrow E \texttt{ + } E \\
E &\rightarrow E \texttt{ * } E \\
E &\rightarrow \texttt{(} E \texttt{)} \\
E &\rightarrow I
\end{aligned}
$$

(a) Provide the parse tree for the input `a = b + c * a`

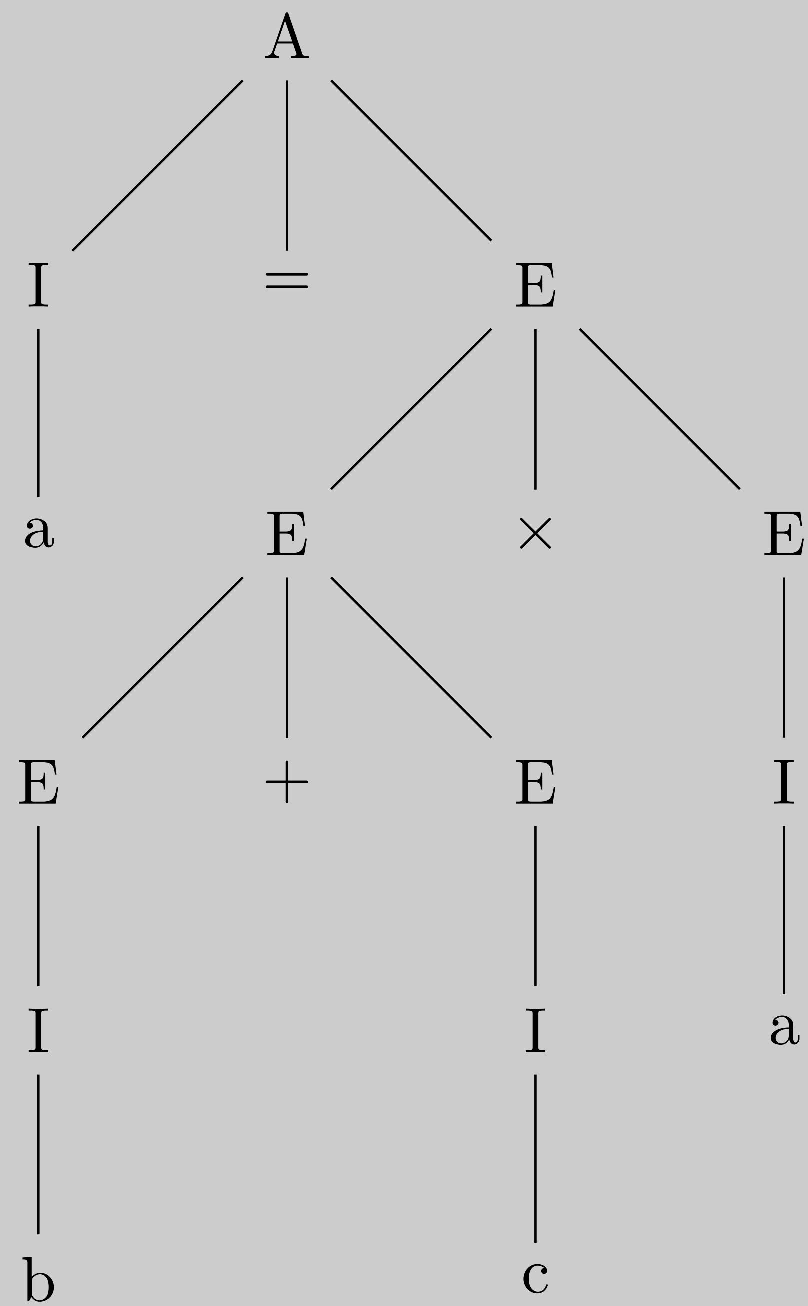
Solution:



(b) Is this language ambiguous? If so, provide an alternate parse tree for `a = b + c * a` to prove it. If not, why not?

Solution:

Yes



Question 11 (12 points)

Consider the following CFG for a list of numerals.

$$
\begin{array}{rl}
L & \rightarrow \epsilon \\
 & \triangleright\ L\texttt{.avg = 0} \\
L & \rightarrow N\ L \\
N & \rightarrow \texttt{0} \\
 & \triangleright\ N\texttt{.value = 0} \\
N & \rightarrow \texttt{1} \\
 & \triangleright\ N\texttt{.value = 1} \\
N & \rightarrow \texttt{2} \\
 & \triangleright\ N\texttt{.value = 2} \\
N & \rightarrow \texttt{3} \\
 & \triangleright\ N\texttt{.value = 3} \\
N & \rightarrow \texttt{4} \\
 & \triangleright\ N\texttt{.value = 4} \\
N & \rightarrow \texttt{5} \\
 & \triangleright\ N\texttt{.value = 5} \\
N & \rightarrow \texttt{6} \\
 & \triangleright\ N\texttt{.value = 6} \\
N & \rightarrow \texttt{7} \\
 & \triangleright\ N\texttt{.value = 7} \\
N & \rightarrow \texttt{8} \\
 & \triangleright\ N\texttt{.value = 8} \\
N & \rightarrow \texttt{9} \\
 & \triangleright\ N\texttt{.value = 9}
\end{array}
$$

(a) Augment this grammar with attribute rules that will accumulate the average of the list into an `avg` attribute at the root of the parse tree. Hint: The first number in a list of five numbers contributes 20% of its value to the average; the average of the last four numbers accounts for the other 80% of the overall average.

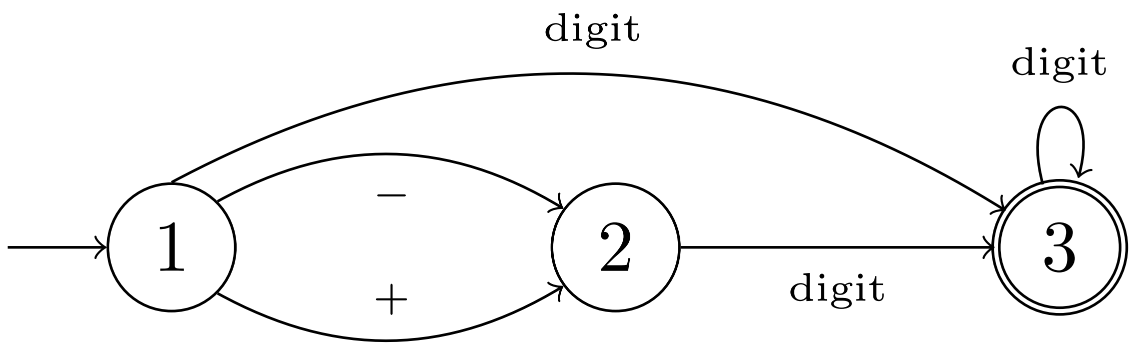
Solution:

$$
\begin{array}{ll}
L & \to \epsilon \\
 & \triangleright\ L.\texttt{avg = 0} \\
 & \triangleright\ L.\texttt{len = 0} \\
L_1 & \to N\ L_2 \\
 & \triangleright\ L_1.\texttt{len = } L_2.\texttt{len + 1} \\
 & \triangleright\ L_1.\texttt{avg = } \frac{L_2.len}{L_1.len} \times L_2.\texttt{avg } + \frac{1}{L_1.len} \times N.\texttt{value} \\
N & \to \texttt{0} \\
 & \triangleright\ N.\texttt{value = 0} \\
N & \to \texttt{1} \\
 & \triangleright\ N.\texttt{value = 1} \\
N & \to \texttt{2} \\
 & \triangleright\ N.\texttt{value = 2} \\
N & \to \texttt{3} \\
 & \triangleright\ N.\texttt{value = 3} \\
N & \to \texttt{4} \\
 & \triangleright\ N.\texttt{value = 4} \\
N & \to \texttt{5} \\
 & \triangleright\ N.\texttt{value = 5} \\
N & \to \texttt{6} \\
 & \triangleright\ N.\texttt{value = 6} \\
N & \to \texttt{7} \\
 & \triangleright\ N.\texttt{value = 7} \\
N & \to \texttt{8} \\
 & \triangleright\ N.\texttt{value = 8} \\
N & \to \texttt{9} \\
 & \triangleright\ N.\texttt{value = 9}
\end{array}
$$

(b) Is your attribute grammar S-attributed?

Solution:
Yes

Question 12 (12 points)
Create the scanner table for this finite state automata that describes optionally signed integers,



Solution:

State	Symbol		
	digit	+	-
1	3	2	2
2	3	Error	Error
3	3	Error	Error

Question 13 (12 points)

(a) Provide two examples of imperative languages

Solution:
Java, C++

(b) Provide two examples of functional languages

Solution:
Haskell, Scheme

(c) Provide one example of logic languages

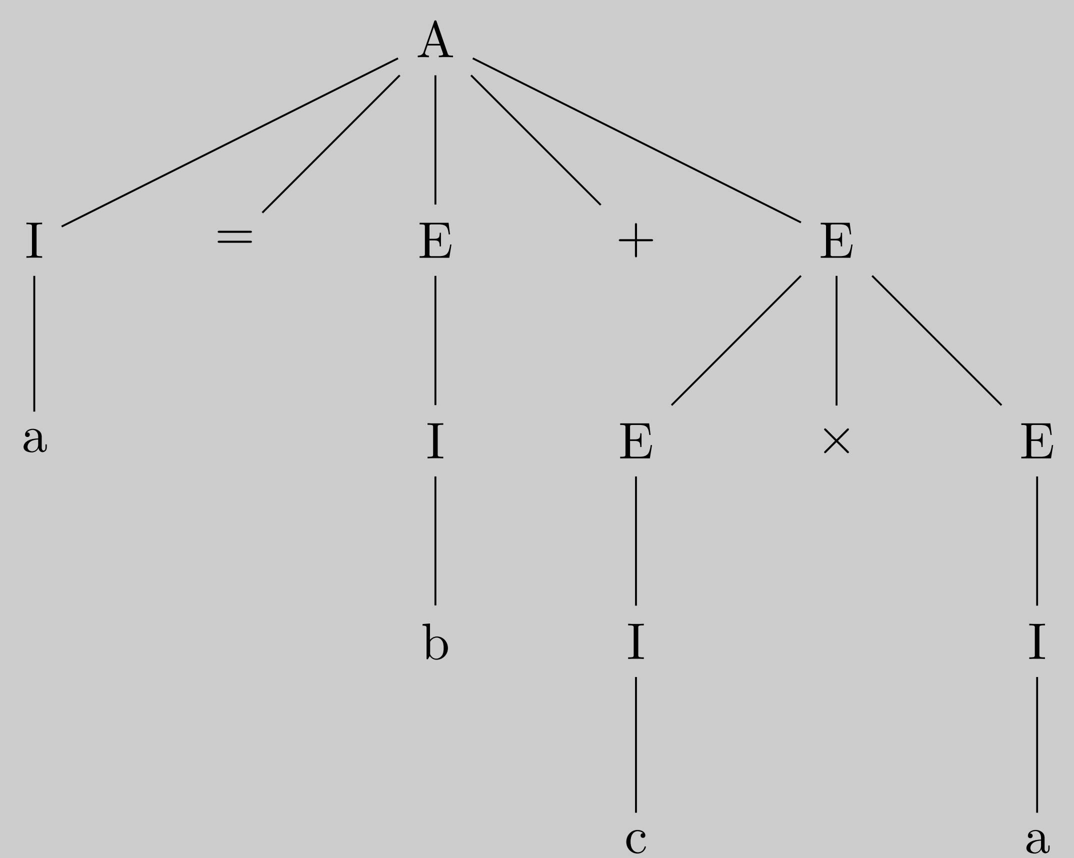
Solution:
Prolog

Question 14 (12 points)
Consider this grammar

$$
\begin{aligned}
A &\rightarrow I \texttt{ = } E \mid I \texttt{ = } E \texttt{ + } E \\
I &\rightarrow \texttt{a} \mid \texttt{b} \mid \texttt{c} \\
E &\rightarrow E \texttt{ * } E \\
E &\rightarrow \texttt{ (} E \texttt{) } \\
E &\rightarrow I
\end{aligned}
$$

(a) Provide the parse tree for the input `a = b + c * a`

Solution:



(b) Is this language ambiguous? If so, provide an alternate parse tree for `a = b + c * a` to prove it. If not, why not?

Solution:
No. Addition is only done at top level

Question 15 (12 points)

Consider the following CFG for a list of numerals.

$$
\begin{aligned}
L &\rightarrow N\ LT \\
LT &\rightarrow \epsilon \\
LT &\rightarrow \texttt{,}\ N\ LT \\
N &\rightarrow \texttt{0} \\
&\quad \triangleright N\texttt{.value = 0} \\
N &\rightarrow \texttt{1} \\
&\quad \triangleright N\texttt{.value = 1} \\
N &\rightarrow \texttt{2} \\
&\quad \triangleright N\texttt{.value = 2} \\
N &\rightarrow \texttt{3} \\
&\quad \triangleright N\texttt{.value = 3} \\
N &\rightarrow \texttt{4} \\
&\quad \triangleright N\texttt{.value = 4} \\
N &\rightarrow \texttt{5} \\
&\quad \triangleright N\texttt{.value = 5} \\
N &\rightarrow \texttt{6} \\
&\quad \triangleright N\texttt{.value = 6} \\
N &\rightarrow \texttt{7} \\
&\quad \triangleright N\texttt{.value = 7} \\
N &\rightarrow \texttt{8} \\
&\quad \triangleright N\texttt{.value = 8} \\
N &\rightarrow \texttt{9} \\
&\quad \triangleright N\texttt{.value = 9}
\end{aligned}
$$

(a) Augment this grammar with attribute rules that will accumulate the maximum of the list into a `max` attribute at the root of the parse tree.

Solution:

$$
\begin{aligned}
L &\rightarrow N\ LT \\
&\triangleright\ LT.\texttt{temp = } N.\texttt{value} \\
&\triangleright\ L.\texttt{max = } LT.\texttt{max} \\
LT &\rightarrow \epsilon \\
&\triangleright\ LT.\texttt{max = } LT.\texttt{temp} \\
LT &\rightarrow \texttt{,}\ N\ LT \\
&\triangleright\ LT.\texttt{temp = max(}N.\texttt{value,}LT.\texttt{temp)} \\
N &\rightarrow \texttt{0} \\
&\triangleright\ N.\texttt{value = 0} \\
N &\rightarrow \texttt{1} \\
&\triangleright\ N.\texttt{value = 1} \\
N &\rightarrow \texttt{2} \\
&\triangleright\ N.\texttt{value = 2} \\
N &\rightarrow \texttt{3} \\
&\triangleright\ N.\texttt{value = 3} \\
N &\rightarrow \texttt{4} \\
&\triangleright\ N.\texttt{value = 4} \\
N &\rightarrow \texttt{5} \\
&\triangleright\ N.\texttt{value = 5} \\
N &\rightarrow \texttt{6} \\
&\triangleright\ N.\texttt{value = 6} \\
N &\rightarrow \texttt{7} \\
&\triangleright\ N.\texttt{value = 7} \\
N &\rightarrow \texttt{8} \\
&\triangleright\ N.\texttt{value = 8} \\
N &\rightarrow \texttt{9} \\
&\triangleright\ N.\texttt{value = 9}
\end{aligned}
$$

(b) Is your attribute grammar S-attributed?

Solution:
No

Question 16 (20 points)

For the regular expression ^(<[^>]*>[A-Za-z0-9+=/]*</[^>]*>)+$, determine which of the following inputs will match

(a) HELLO

> **Solution:**
> Yes

(b) <pre>HELLO

> **Solution:**
> No

(c) 5>4

> **Solution:**
> No

(d) 1+1=2

> **Solution:**
> Yes

(e) 5-2=3

> **Solution:**
> No

(f)

> **Solution:**
> No

(g) 1</pre>

> **Solution:**
> Yes

(h) <></>

> **Solution:**
> Yes

(i) OK</pre>

> **Solution:**
> No

(j) <pre>OK</pre>

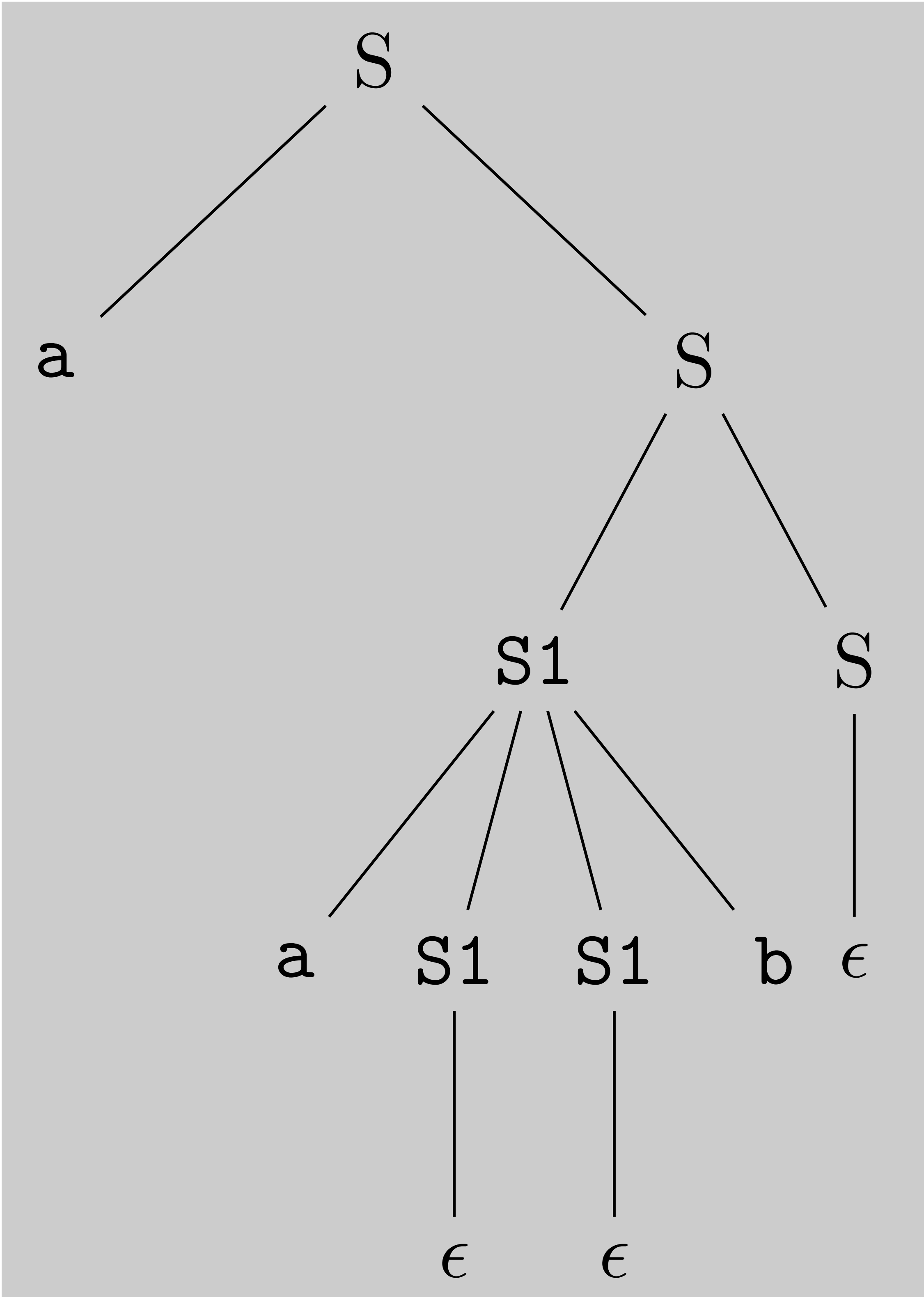
> **Solution:**
> No

Question 17 (12 points)
Consider this grammar

$$\begin{aligned} S &\to \texttt{a}\ S \\ S &\to S1\ S \\ S &\to \epsilon \\ S1 &\to \texttt{a}\ S1\ S1\ \texttt{b} \\ S1 &\to \epsilon \end{aligned}$$

(a) Provide the parse tree for the input `aab`

Solution:



(b) Is this language ambiguous? If so, provide an alternate parse tree for `aab` to prove it. If not, why not?

Solution:
No. $S1$ forces `a` to match `b`, so there is no other way to match a `b`

Question 18 (24 points)

Consider the following CFG for a list of vowels.

$$
\begin{aligned}
L &\rightarrow \epsilon \\
 &\quad \triangleright L.\texttt{all = false} \\
L &\rightarrow N\ L \\
N &\rightarrow \texttt{A} \\
N &\rightarrow \texttt{E} \\
N &\rightarrow \texttt{I} \\
N &\rightarrow \texttt{O} \\
N &\rightarrow \texttt{U}
\end{aligned}
$$

(a) Augment this grammar with attribute rules that will accumulate the value `true` into an `all` attribute at the root of the parse tree if the list contains at least one of each of the possible vowels (A,E,I,O,U), and `false` otherwise.

Solution:

$L \rightarrow \epsilon$

$\triangleright$ L.all = false
$\triangleright$ L.A = false
$\triangleright$ L.E = false
$\triangleright$ L.I = false
$\triangleright$ L.O = false
$\triangleright$ L.U = false

$L \rightarrow N\ L$

$\triangleright$ L_1.A = L_2.A OR N.A
$\triangleright$ L_1.E = L_2.E OR N.E
$\triangleright$ L_1.I = L_2.I OR N.I
$\triangleright$ L_1.O = L_2.O OR N.O
$\triangleright$ L_1.U = L_2.U OR N.U
$\triangleright$ L_1.all = L_1.A AND L_1.E AND L_1.I AND L_1.O AND L_1.U

$N \rightarrow$ A

$\triangleright$ N.A = true
$\triangleright$ N.E = false
$\triangleright$ N.I = false
$\triangleright$ N.O = false
$\triangleright$ N.U = false

$N \rightarrow$ E

$\triangleright$ N.E = true
$\triangleright$ N.A = false
$\triangleright$ N.I = false
$\triangleright$ N.O = false
$\triangleright$ N.U = false

$N \rightarrow$ I

$\triangleright$ N.I = true
$\triangleright$ N.A = false
$\triangleright$ N.E = false
$\triangleright$ N.O = false
$\triangleright$ N.U = false

$N \rightarrow$ O

$\triangleright$ N.O = true
$\triangleright$ N.A = false
$\triangleright$ N.E = false
$\triangleright$ N.I = false
$\triangleright$ N.U = false

$N \rightarrow$ U

$\triangleright$ N.U = true
$\triangleright$ N.A = false
$\triangleright$ N.E = false
$\triangleright$ N.I = false
$\triangleright$ N.O = false

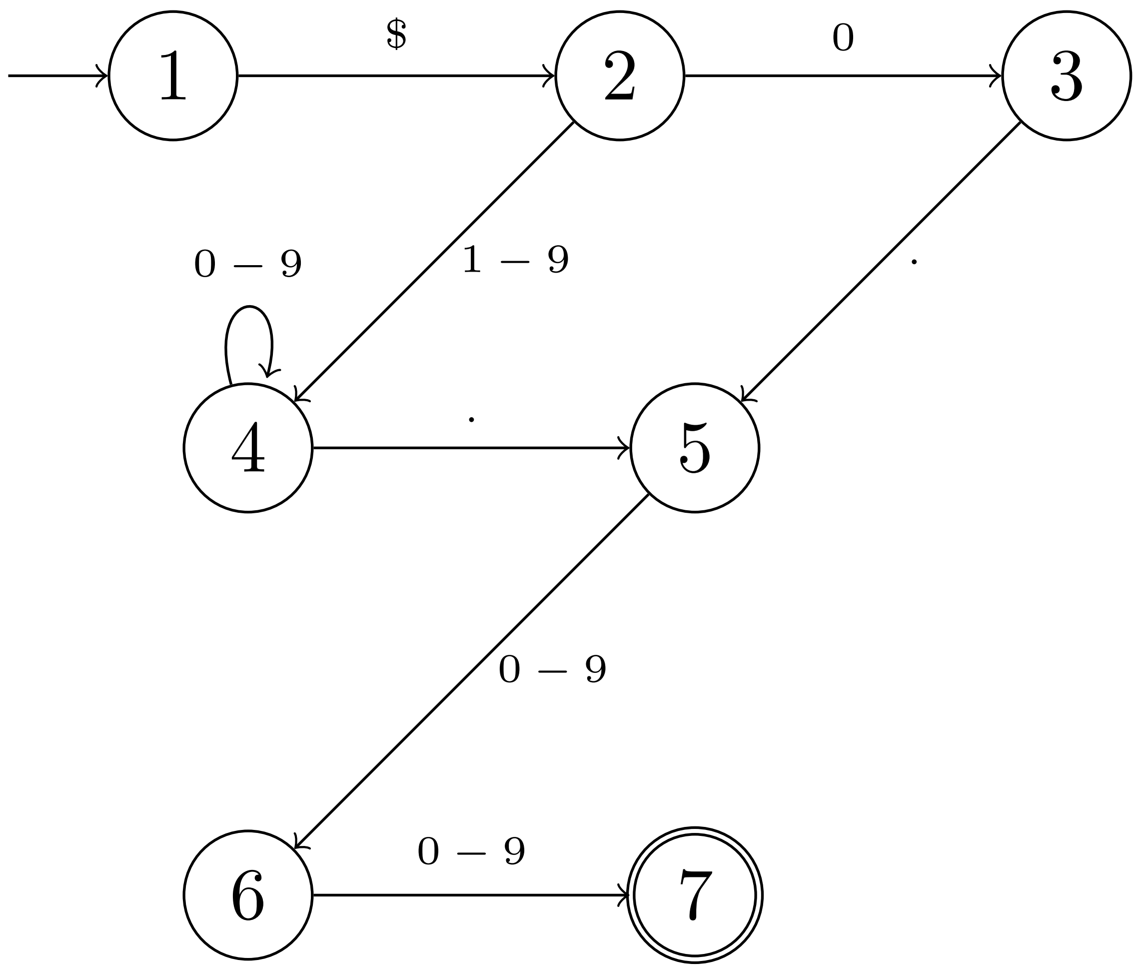
(b) Is your attribute grammar S-attributed?

Solution:

Yes

Question 19 (24 points)

Create the scanner table for this finite state automata that describes amounts of US dollars.



Solution:

State	Symbol			
	`$`	`0`	`1-9`	`.`
1	2	-	-	-
2	-	3	4	-
3	-	-	-	5
4	-	4	4	5
5	-	6	6	-
6	-	7	7	-
7	-	-	-	-