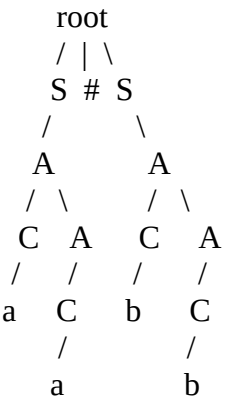
Olaolu Emmanuel
RUID: 159-00-3321
314:02
Assignment 2

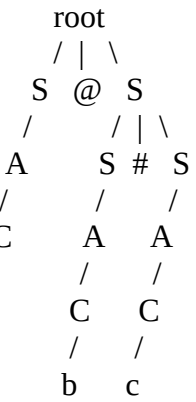
1A. parse tree:

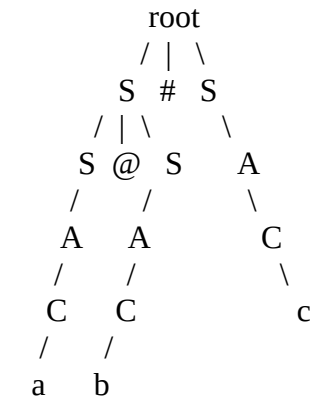


derivation:

S
S # S
A # S
C A # S
a A # S
a C # S
a a # S
a a # A
a a # C A
a a # b A
a a # b C
= aa # bb

1B. parse tree (it is ambiguous):





derivation:

<u>S</u>
<u>S</u> @ S
<u>A</u> @ S
<u>C</u> @ S
a @ <u>S</u>
a @ <u>S</u> # <u>S</u>
a @ <u>A</u> # <u>A</u>
a @ <u>C</u> # <u>C</u>
= a @ b # c

1C. parse tree

derivation:

<u>S</u>
<u>A</u>
<u>C</u> A
a <u>A</u>
a <u>C</u>
= ab

2. new grammar:

S → A | S @ T | T
T → T # A | A
A → C | C A
C → a | b | c

3A. Yes. Statement → Assignment → <Var> = <Value> [, <Value>]; → a = <Value> [, <Value>];
→ a = 0 [, <Value>]; → a = 0 , b;

3B. No. The only way to have “, <value>” is through the Assignment rule and in that rule the “, <value>” is optional meaning you can either put it once or not at all. We cannot add multiple commas so the expression is not in the grammar.

3C. Yes. Statement → <While> → while(<Value>) { {<Statement>} }
→ while(a) { {<Statement>} } → while(a) { <Statement> <Statement> }
→ while(a) { <Assignment> <While> }
→ while(a) { (<Var> = <Value> [, <Value>];) (while(<Value>) { {<Statement>} }) }
→ while(a) { b = 0; while (b){ } }

3D. No. We would have to make the statement an Assignment and you cannot make a while expression from that therefore it is not possible.

3E. <Statement> → <If>
<If> → if(<Value>) { {<Statement} } [else { {<Statement>} }]