Bruer Kelly
Chapter 3 HW

1. EBNF descriptions

a.) Java switch statement

<switch_stmt> => switch(<logic_expr>){<switch_block>}
<switch_block> => case <literal>: <stmt_list>
{case <literal>: <stmt_list>}
[default: <stmt_list>]

b.) Python while statement

<while_stmt> => while <logic_expr>:
<stmt_list>

2. Give '+' precedence over '*', force '+' right associative

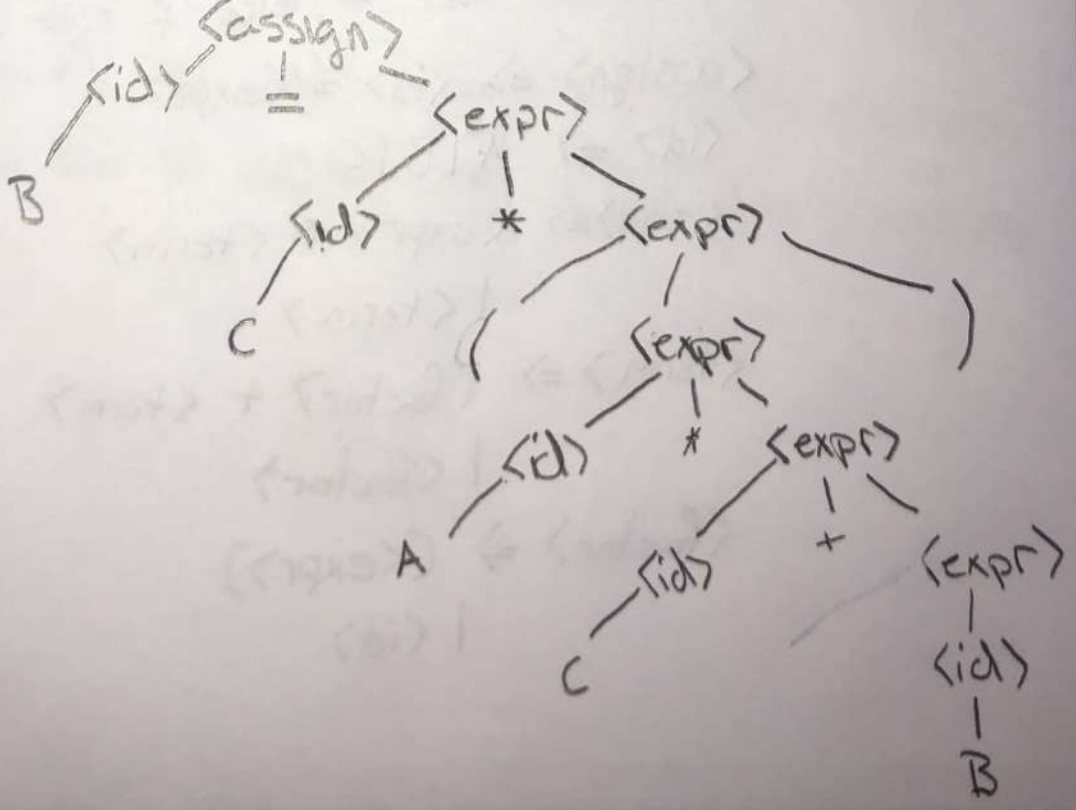
<assign> => <id> = <expr>
<id> => A | B | C
<expr> => <expr> * <term>
| <term>
<term> => <factor> + <term>
| <factor>
<factor> => (<expr>)
| <id>

3. B = C * (A * C + B)

Leftmost derivation:

<assign> → <id> = <expr>
→ B = <expr>
→ B = <id> * <expr>
→ B = C * <expr>
→ B = C * (<expr>)
→ B = C * (<id> * <expr>)
→ B = C * (A * <expr>)
→ B = C * (A * <id> + <expr>)
→ B = C * (A * C + <expr>)
→ B = C * (A * C + <id>)
→ B = C * (A * C + B)

Parse tree:



4. Ambiguous Proof:

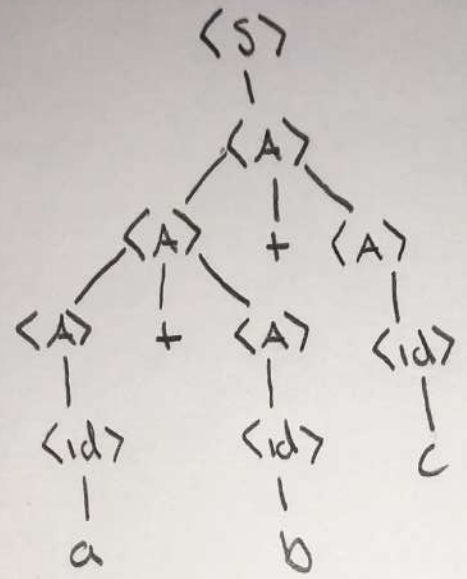
<S> → <A>

<A> → <A> + <A> | <id>

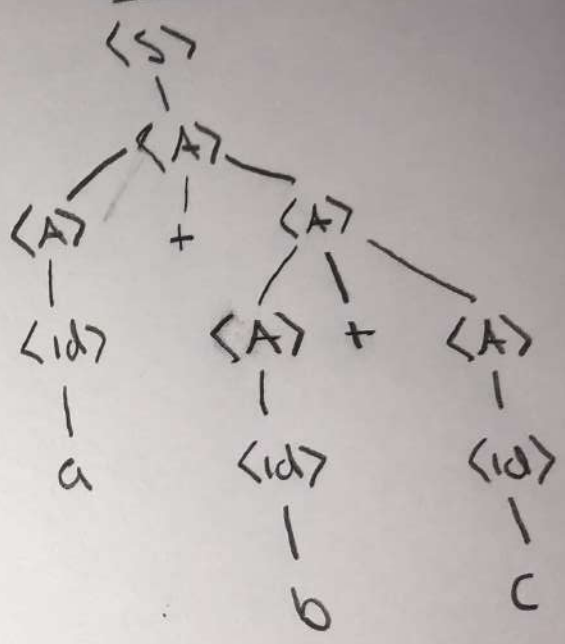
<id> → a | b | c

a + b + c

Tree #1



Tree #2



Note: In tree #1, the addition of a + b has precedence.

In tree #2, the addition of b + c has precedence.