Shefali Emmanuel

CSCI 320

HW 6

1.

1. ([0-9])
2. (\b[0-9]+\b)
3. (\b([0-9]+ | [0-9]+ x[0-9]+)\b)
   1. [^abc] not abc
   2. \D not digit
   3. \B not word boundary

2.

1. ((0))

expr => term

=> factor

=>(expr)

=>(term)

=>(factor)

=>((expr))

=>((term))

=>((factor))

=>((number))

=>((digit))

=>((0))

b. 2 \* 3 + 5 \* 6 + 9

expr => expr + term

=> expr + term + term

=> term + term + term

=> term \* factor + term + term

=> term \* factor + term \* factor + term

=> factor \* factor + term \* factor + term

=> factor \* factor + factor \* factor + term

=> factor \* factor + factor \* factor + factor

=> number \* factor + factor \* factor + factor

=> number \* number + factor \* factor + factor

=> number \* number + number \* factor + factor

=> number \* number + number \* number + factor

=> number \* number + number \* number + number

=> digit \* number + number \* number + number

=> digit \* digit + number \* number + number

=> digit \* digit + digit \* number + number

=> digit \* digit + digit \* digit + number

=> digit \* digit + digit \* digit + digit

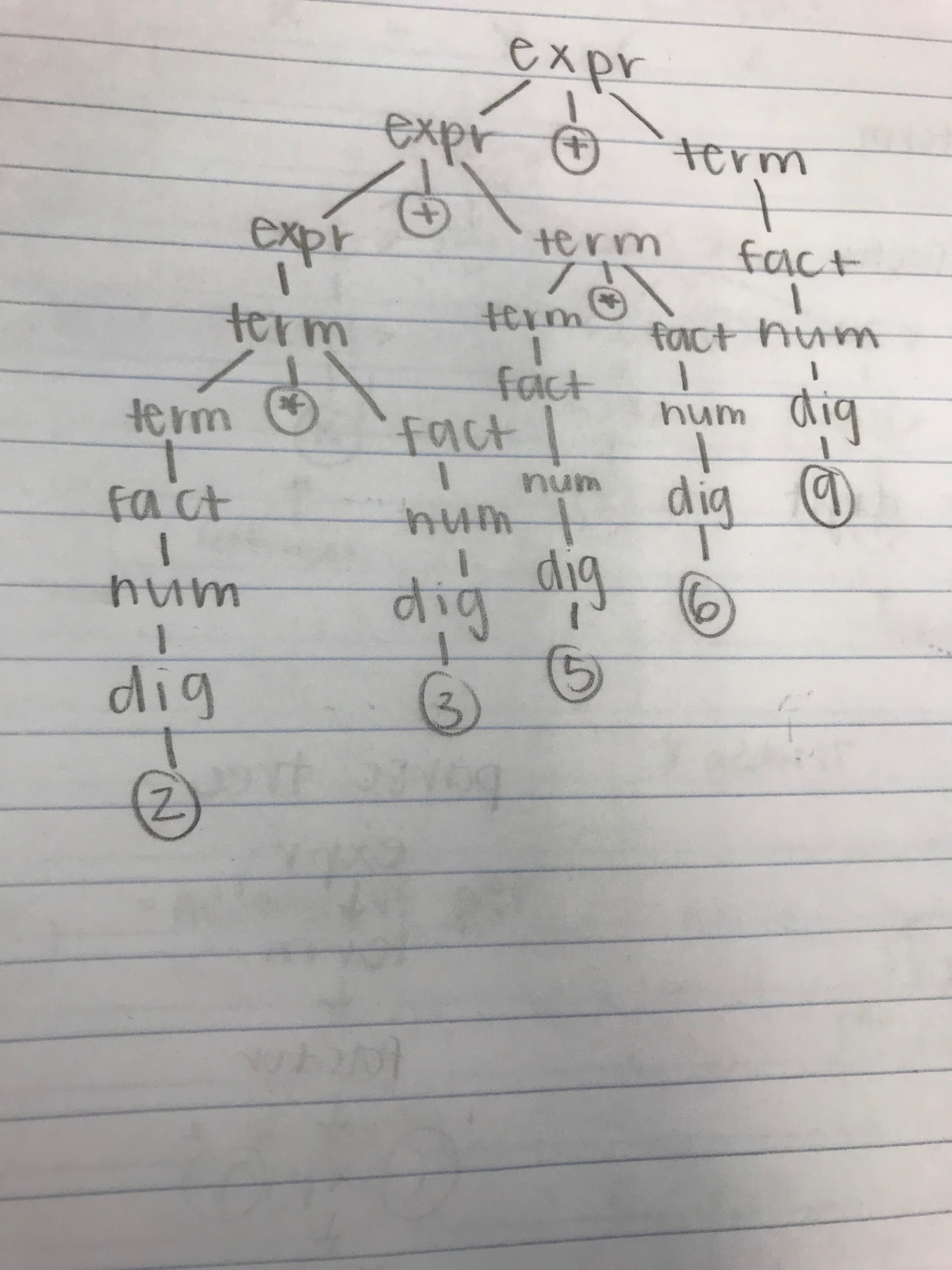
=> 2 \* digit + digit \* digit + digit

=> 2 \* 3 + digit \* digit + digit

=> 2 \* 3 + 5 \* digit + digit

=> 2 \* 3 + 5 \* 6 + digit

=> 2 \* 3 + 5 \* 6 + 9



c. 2 \* (3 + 5) \* (6 + 9)

expr => term

=> term \* factor \* factor

=> factor \* factor \* factor

=> number \* factor \* factor

=> factor \* (expr) \* factor

=> factor \* (expr) \* (expr)

=> number \* (expr) \* (expr)

=> number \* (expr + term) \* (expr)

=> number \* (expr + term) \* (expr + term)

=> 2 \* (expr + term) \* (expr + term)

=> 2 \* (term + term) \* (expr + term)

=> 2 \* (factor + term) \* (expr + term)

=> 2 \* (factor + factor) \* (expr + term)

=> 2 \* (number + factor) \* (expr + term)

=> 2 \* (number + number) \* (expr + term)

=> 2 \* (digit + number) \* (expr + term)

=> 2 \* (digit + digit) \* (expr + term)

=> 2 \* (3 + digit) \* (expr + term)

=> 2 \* (3 + 5) \* (expr + term)

=> 2 \* (3 + 5) \* (term + term)

=> 2 \* (3 + 5) \* (factor + term)

=> 2 \* (3 + 5) \* (factor + factor)

=> 2 \* (3 + 5) \* (number + factor)

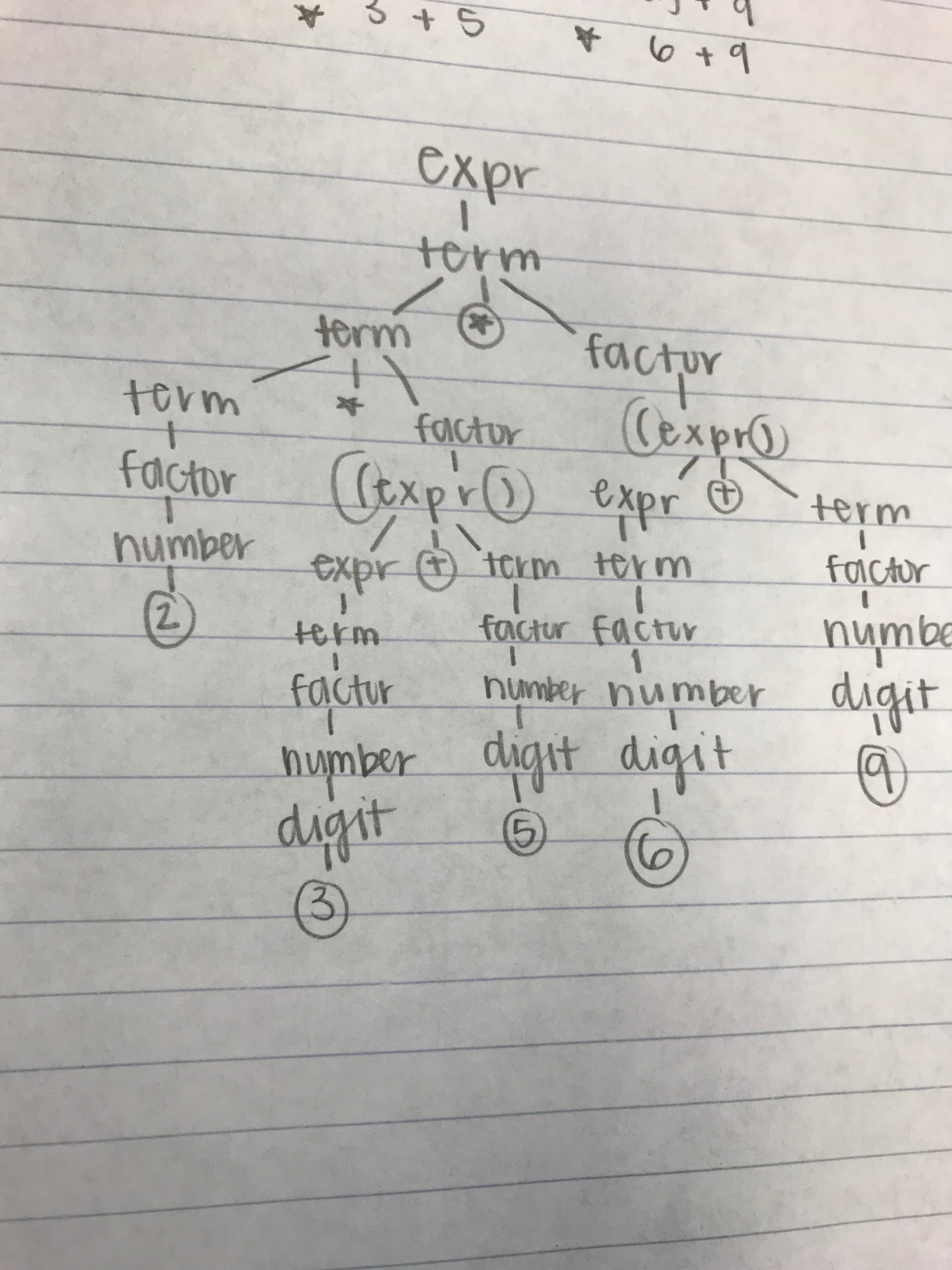
=> 2 \* (3 + 5) \* (number + number)

=> 2 \* (3 + 5) \* (digit + number)

=> 2 \* (3 + 5) \* (digit + digit)

=> 2 \* (3 + 5) \* (6 + digit)

=> 2 \* (3 + 5) \* (6 + 9)



3. <expr> -> \*<let>\*

<let> -> <as> | <bs>

<as> -> <as> A| A

<bs> -> <bs> B | B

4.

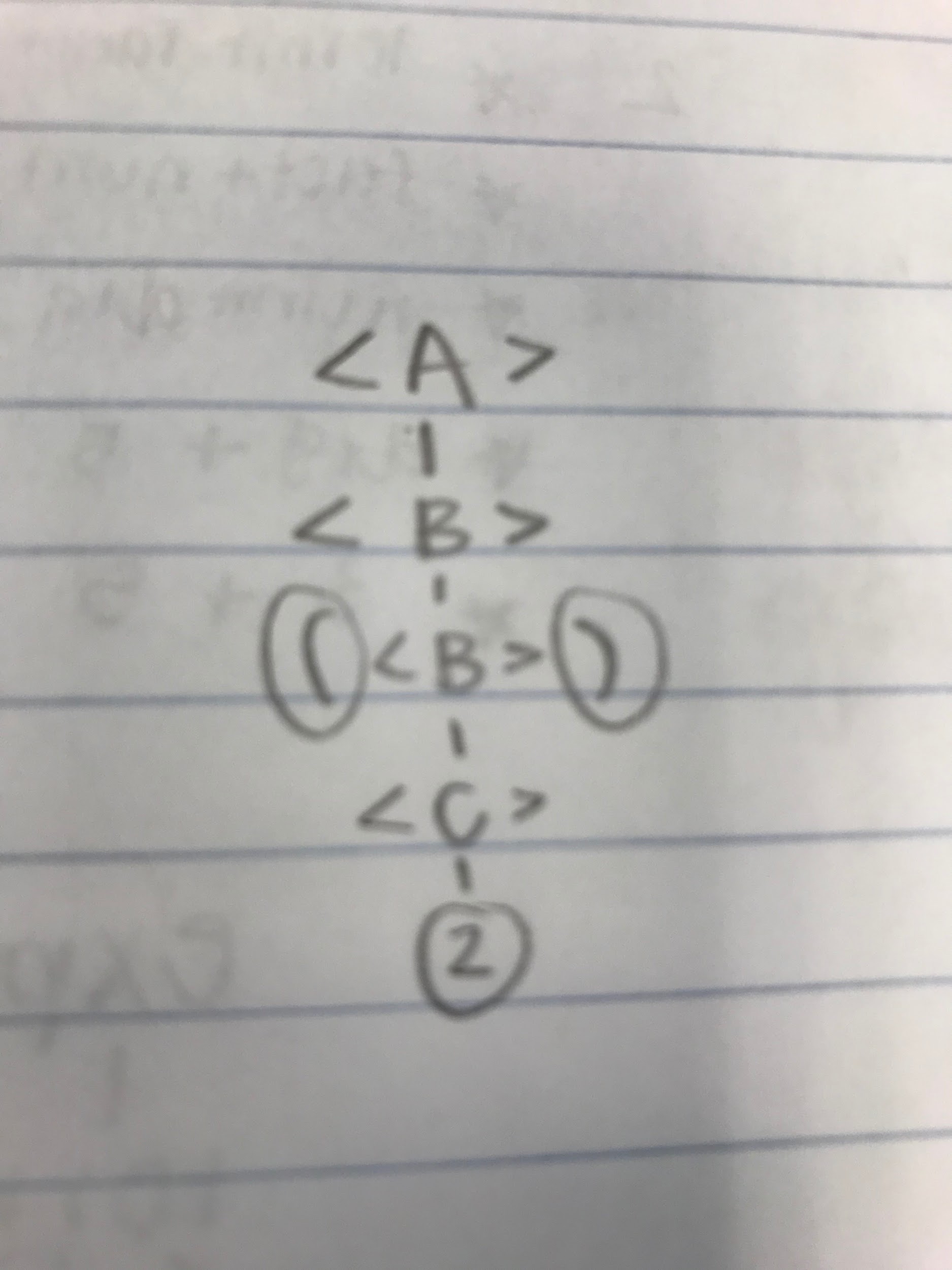
1. (2)

<A> => <B>

=> (<B>)

=> (<C>)

=> (2)



1. [2]

NOT POSSIBLE

1. [(1),2]

<A> => [<B>, <A>]

=> [(<B>), <A>]

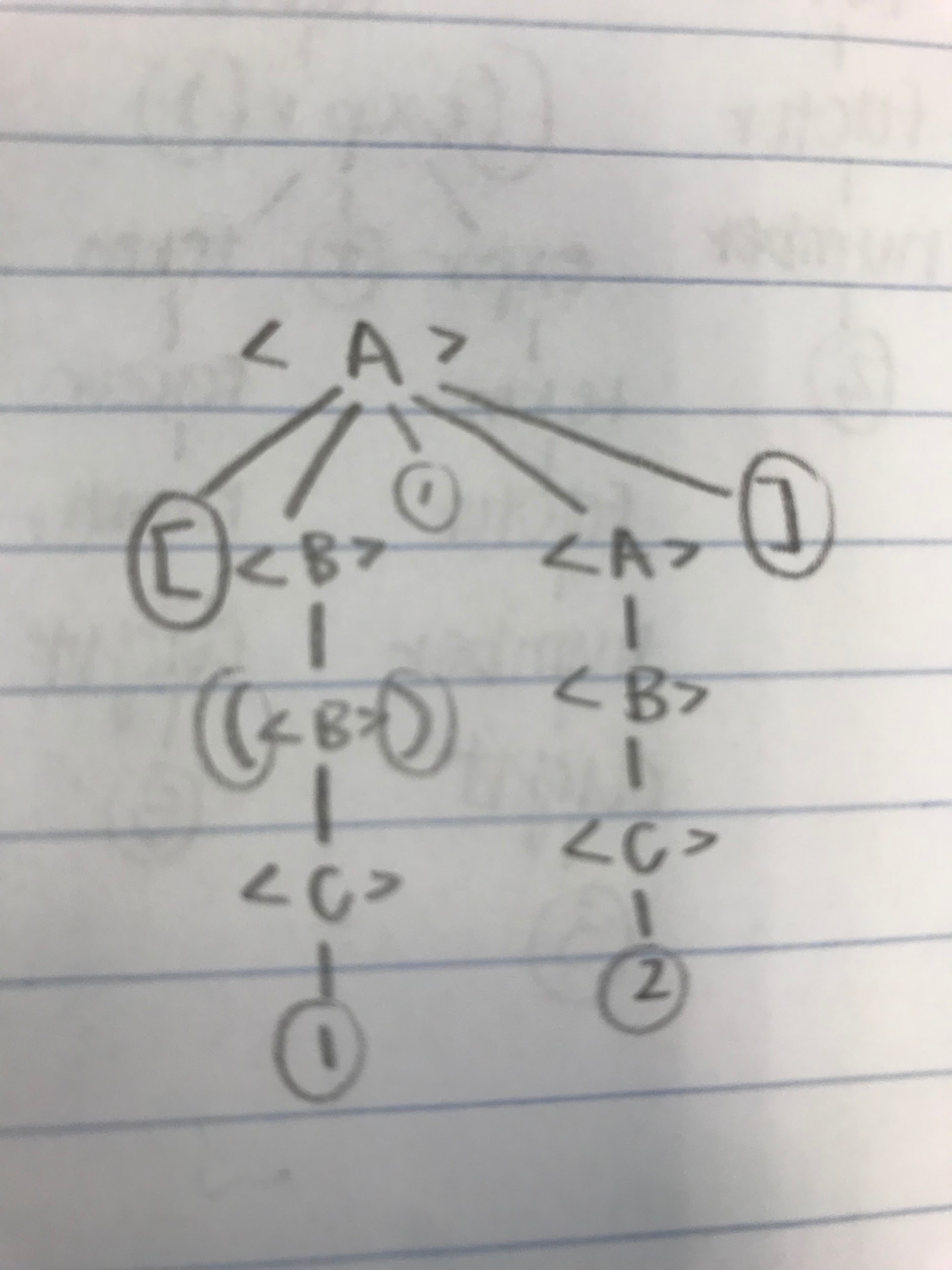
=> [(<B>), <B>]

=> [(<C>), <B>]

=> [(<C>), <C>]

=> [(<1>), <C>]

=> [(<1>), <2>]



1. [(1), [3,1], ([3])]

NOT POSSIBLE

5. YACC: Yet Another Compiler Compiler, program developed for the Unix System by Stephen Johnson