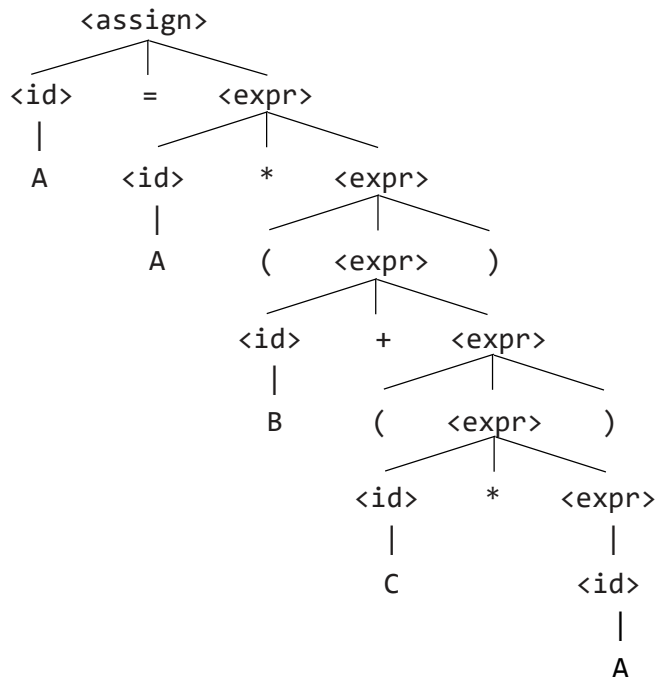
## CHAPTER 3 SOLUTIONS

3.
<assign> → <id> = <expr> <id> → A | B | C
<expr> → <expr> * <term> | <term>
<term> → <factor> + <term> | <factor>
<factor> → ( <expr> )
| <id>

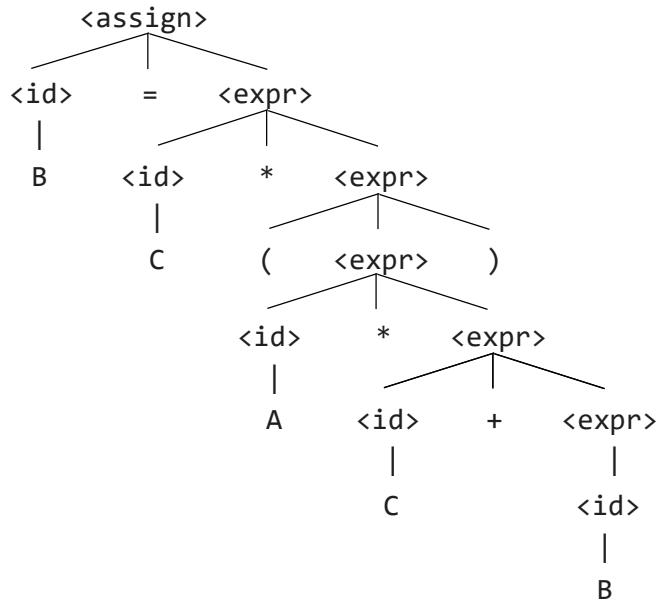
4.
<assign> -> <id> = <expr>
<id> -> A | B | C
<expr> -> <expr> + <term>
| <term>
<term> -> <term> * <factor>
| <factor>
<factor> -> ( <expr> )
| <id>
| <id> ++
| <id> - -

6.
a) A = A * (B + (C * A))
<assign> => <id> = <expr>
=> A = <expr>
=> A = <id> * <expr>
=> A = A * <expr>
=> A = A * ( <expr> )
=> A = A * ( <id> + <expr> )
=> A = A * ( B + <expr>)
=> A = A * ( B + ( <expr> ) )
=> A = A * ( B + ( <id> * <expr> ) )
=> A = A * ( B + ( C * <expr> ) )
=> A = A * ( B + ( C * <id> ) )
=> A = A * ( B + ( C * A ) )



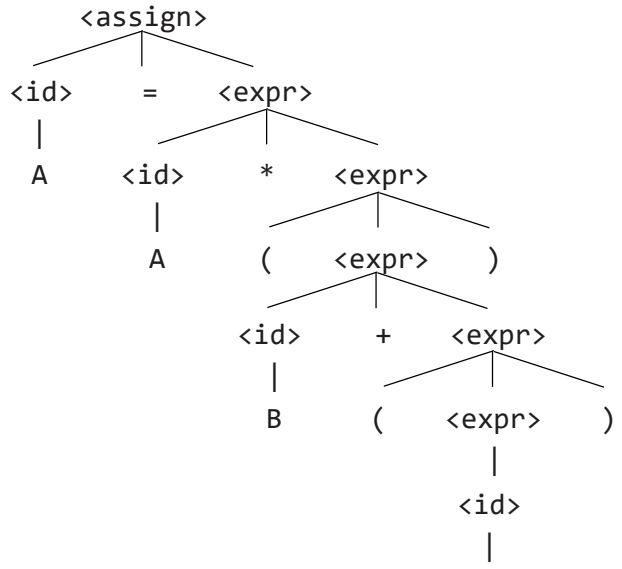

b) B = C * (A * C + B)

<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 )




c) A = A * (B + (C))

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



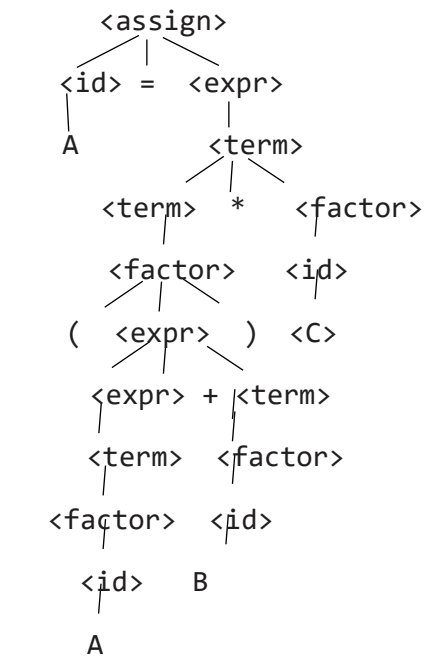
C

7.

a. A = ( A + B ) * C

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

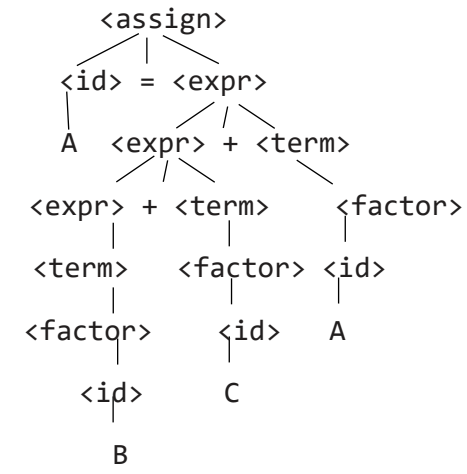
Parse Tree



b. A = B + C + A

<assign> => <id> =<expr>
=> A = <expr>
=> A = <expr> + <term>
=> A = <expr> + <term> + <term>
=> A = <term> + <term> + <term>
=> A = <factor> + <term> + <term>
=> A = <id> + <term> + <term>
=> A = B + <term> + <term>
=> A = B + <factor> + <term>
=> A = B + <id> + <term>
=> A = B + C + <term>
=> A = B + C + <factor>
=> A = B + C + <id>
=> A = B + C + A

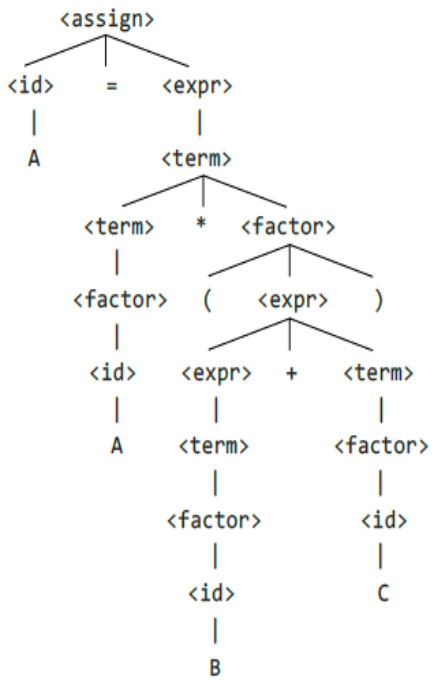
Parse Tree




c. A = A * (B + C)

<assign>=><id>=<expr>
=> A = <expr>
=> A = <term>
=> A = <term> * <factor>
=> A = <factor> * <factor>
=> A = <id> * <factor>

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

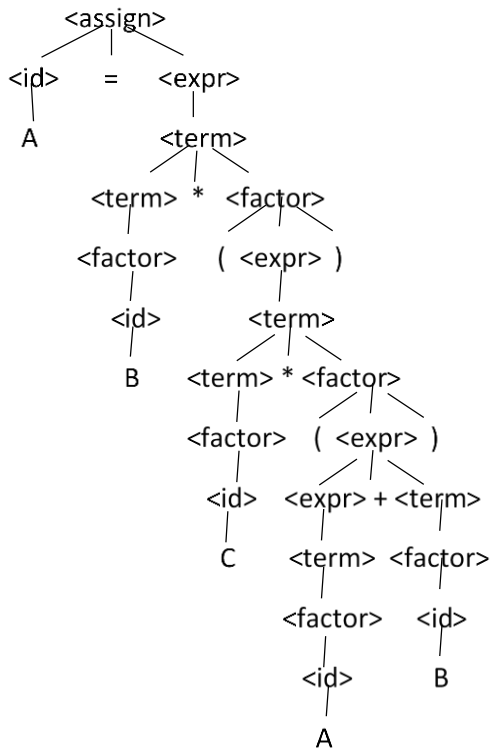



d. A = B * (C * (A + B))

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

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

Parse Tree



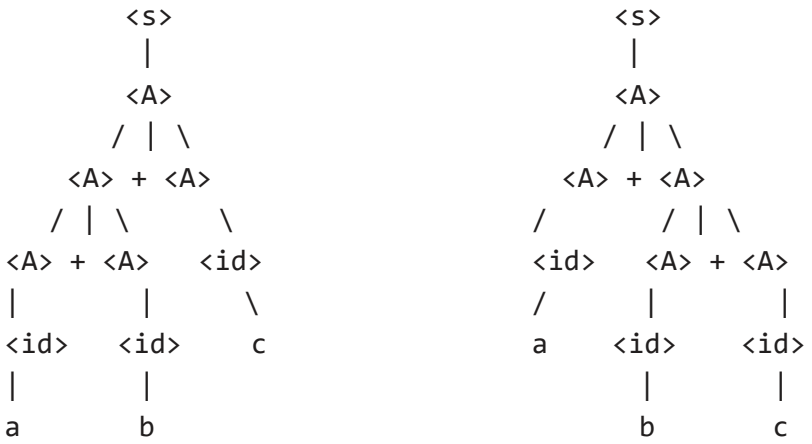

8.

*<S> → <A>*

*<A> → <A> + <A> | <id>*

*<id> → a | b | c*

Take "a+b+c" as example, it has two possible parse trees:




which will cause the ambiguity.

11.
*<S> → <A> a <B> b*
*<A> → <A> b | b*
*<B> → a <B> | a*
This BNF can be represented in EBNF:
*S = "b", {"b"}, "a", "a", {"a"}, "b";*

Thus, "*baab*" and "*bbaab*" are generated by this grammar; "*bbbab*" and "*bbaaaaa*" are not.
The answer is (a) (d).

12.
*<S> → a <S> c <B> | <A> | b*
*<A> → c <A> | c*
*<B> → d | <A>*
This BNF can be represented in EBNF:
*S = "a", S, "c", ("d" | "c", {"c"}) | "c", {"c"} | "b";*

Thus, "*abcd*" and "*accc*" are generated by this grammar; the others are not.
The answer is (a) (e).

13.
Derivation –

<S>=>a b |a <S> b

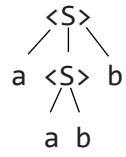
14.

For sentence aabb:

Derivation –

<S> => a <S> b
=> a a b b

Parse Tree




For sentence aaaabbbb:

Derivation –

<S> => a <S> b
=> a a <S> b b
=> a a a <S> b b b
=> a a a a b b b b

Parse Tree