When a grammar produces 2 or more parse trees for a sequence it is called ambiguous. We need to design unambiguous grammars, but sometimes this is impossible. Here is how we do it for our expressions with generation of numbers thrown in as well.

\[ NT = \{ <N>, <D>, <E>, <F>, <T> \} \]

Start Symbol is \( <E> \)

\[ \Sigma = \{ +, *, (, ), 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \} \]

Now we try to generate \( 3 + 4 * 5 \)

This is the correct grouping.
Let us try to create the "wrong" grouping. We want to make the * appear at the top of the tree with the + nested inside it.

The --- lines indicate the "obvious" steps that I have left out.

Notice how we were forced to put parentheses around $3 + 4$ when we tried to do this.